Certified Polyhedral Decompositions of Collision-Free Configuration Space

International Journal of Robotics Research
XX(X):1–45
©The Author(s) 0000
Reprints and permission:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Hongkai Dai[*2], Alexandre Amice[*1], Peter Werner[1], Annan Zhang[1], Russ Tedrake[1,2]

Abstract
Understanding the geometry of collision-free configuration space (C-free) in the presence of task-space obstacles is an essential ingredient for collision-free motion planning. While it is possible to check for collisions at a point using standard algorithms, to date no practical method exists for computing C-free *regions* with rigorous certificates due to the complexity of mapping task-space obstacles through the kinematics. In this work, we present the first to our knowledge rigorous method for approximately decomposing a rational parametrization of C-free into certified polyhedral regions. Our method, called C-Iris (C-space Iterative Regional Inflation by Semidefinite programming), generates large, convex polytopes in a rational parameterization of the configuration space which are rigorously certified to be collision-free. Such regions have been shown to be useful for both optimization-based and randomized motion planning. Based on convex optimization, our method works in arbitrary dimensions, only makes assumptions about the convexity of the obstacles in the *task* space, and is fast enough to scale to realistic problems in manipulation. We demonstrate our algorithm's ability to fill a non-trivial amount of collision-free C-space in several 2-DOF examples where the C-space can be visualized, as well as the scalability of our algorithm on a 7-DOF KUKA iiwa, a 6-DOF UR3e and 12-DOF bimanual manipulators. An implementation of our algorithm is open-sourced in Drake. We furthermore provide examples of our algorithm in interactive Python notebooks.

1 Introduction

The notion of configuration space (C-space) has played a foundational role in robot motion planning since its proposal in the seminal work (Lozano-Perez 1983). In the presence of obstacles in the Cartesian

[1]Massachusetts Institute of Technology (MIT), [2]Toyota Research Institute, [*]equal contribution

task space, a fundamental challenge is describing the collision-free C-space (C-free): the full range of configurations for which a robot is not in collision. Prior work has taken two complementary approaches to this problem.

The first approach attempts to find an explicit description of the C-space obstacles from their task-space description and the inverse kinematics (IK). We refer to this approach as the negative approach, as C-free is described as the complement of the set of C-space obstacles. In its full generality, the problem of describing C-space obstacles is intractable (Canny 1988), and so limiting assumptions on the robot are often made. For example, (Kavraki 1995) develops a method for computing C-space obstacles based on the Fast Fourier Transform under the assumption that the robot can only translate in the workspace. In (Branicky and Newman 1990), explicit descriptions of C-space obstacle due to the presence of point, line, and planar task-space obstacles are presented for two and three degree of freedom (DOF) robots. A thorough review of describing C-space obstacles can be found in (Latombe 2012, Chapter 3). There it is shown that if all the task-space obstacles are described as semi-algebraic sets (i.e. as the intersection and union of polynomial inequalities) then C-space obstacles are also semi-algebraic. This is an important result from a complexity-theoretic standpoint as it shows that describing the C-space obstacles is at least decidable, though still very hard.

We refer to the second approach as the positive approach, as it seeks to directly describe C-free as a union of simpler sets. This description is attractive as a variety of optimization-based motion planning methods can efficiently leverage such descriptions, particularly when the simpler sets are convex (Deits and Tedrake 2015b; Schouwenaars et al. 2001; Marcucci et al. 2021, 2022).

Rapidly-exploring Random Trees (RRT) (LaValle 1998), Probabilistic Roadmaps (PRM) (Kavraki et al. 1996), and their variants can all be considered examples of this approach, describing C-free using piecewise-linear paths. Frequently, these methods provide probabilistic guarantees that the paths contain no collisions via sampling along the paths. To avoid false positive claims of non-collision, rigorous certification procedures such as (Schwarzer et al. 2004) can be used. Works such as (Verghese et al. 2022; Han et al. 2019; Wong et al. 2014) all seek to describe non-zero volume subsets of C-free. Similar to RRTs and PRMs, these methods have the advantage of working in arbitrary configuration spaces, make no assumptions on the C-space obstacles, and proceed via sampling. Therefore, they are typically relatively simple to implement and quite fast in low dimensions. Unfortunately, these methods only provide probabilistic guarantees of non-collision.

When the C-space obstacles are assumed to be convex, rigorous descriptions of C-free may be possible, though hardness results exist. For example, in two and three dimensions with polyhedral C-space obstacles, it is known that finding a minimal decomposition is NP-hard (Lingas 1982) to solve exactly and even APX-hard (Eidenbenz and Widmayer 2003) to approximate[*]. Works such as (Lien and Amato 2007) and (Ghosh et al. 2013) overcome these hardness results by finding decompositions that are unions of approximately convex sets.

In arbitrary dimensions and under the assumption of known, convex C-space obstacles, C-free can be decomposed into convex polyhedra by using the IRIS algorithm of (Deits and Tedrake 2015a). As it is based on convex programming, IRIS is relatively fast, and is also able to generate *rigorous certificates* of

[*]A problem is said to be APX-hard if no polynomial time algorithm can achieve an approximation ratio of $1+\delta$ for some $\delta > 0$ unless $P = NP$.

non-collision. Unfortunately, it is often the case that obstacles are naturally described as convex sets in *task space*, which are rarely convex in C-space.

In this work, we similarly provide a method for describing C-free using convex polyhedra in a bijective, rational parametrization of C-space known as the tangent configuration space (TC-space). Our primary technical contributions are two convex (specifically Sums-of-Squares (SOS)) programs which can certify that a polyhedron in TC-space contains no collision when the obstacles are specified as convex sets in *task space*. Similar to (Deits and Tedrake 2015a), we then construct certified, collision-free polytopic regions by alternating between a pair of convex programs. Our method works in arbitrary dimensions and is the first to our knowledge to provide rigorous certificates for non-zero volume sets in this setting. Moreover, we provide a fast, mature implementation technique in the open-source robotics toolbox Drake[†].

A conference version of this paper is published in (Amice et al. 2022), which assumes a robotic manipulator composed of revolute joints operating in a scene where all task-space obstacles are decomposed as a union of vertex representation (V-rep) polytopes. This journal version extends these results in many ways.

First, we demonstrate how our approach can be extended to handle other common, non-polytopic geometries such as spheres, capsules, and cylinders. Moreover, we describe how to extend our approach to handle a robot composed of any of the algebraic joints: revolute, prismatic, spherical, planar, and cylindrical. Our second technical contribution introduces a second method for certifying non-collision inspired by the dual of the separating hyperplane approach used in the conference paper. This approach takes the form of certifying the emptiness of a set of polynomial equations and inequalities which can also be written as an optimization program. The third technical contribution of this work is to show that feasibility of the optimization programs we use for certification is not only sufficient, but also necessary for a TC-space region to be collision free provided the degree of certain polynomials are chosen sufficiently large. Finally, we provide new examples of our algorithm deployed on various robots including 2-DOF robots to visualize the TC-space, a robot containing a prismatic joint, and a UR3e robot with collision geometries approximate by cylinders.

We begin in Section 2 by formally introducing our problem and our assumptions. We proceed in Section 3 by introducing necessary mathematical background for describing our technical approach. In Section 4, we present our most technical results: two convex programs which can certify whether a region of TC-space is collision-free. We also state the conditions under which feasibility of these programs are guaranteed when a proposed region is collision-free. We describe how to leverage the certification programs to generate convex decompositions of TC-free in Section 5. We conclude in Section 6 with examples of our algorithm deployed on various robots. We will first illustrate the algorithm on two simple 2-DOF systems where both the task and configuration spaces can be visualized and the entire configuration space can be quickly covered. We next demonstrate the ability of our algorithm to certify a wide range of postures for two realistic, 7-DOF manipulators interacting with a shelf. We conclude by showing our algorithm's ability to scale by exploring two 12-DOF, bimanual manipulators.

Notation: Throughout the paper, we will use calligraphic letters ($\mathcal{S}$) to denote sets, Roman capitals (X) to denote matrices, and Roman lower case (x) to denote vectors. We use $[N] = \{1, \dots, N\}$, denote the

[†]https://drake.mit.edu/

set of all multivariate polynomials in the vector of variables x as $\mathbb{R}[x]$, and denote the cone of Sums-of-Squares (SOS) polynomials as $\mathbf{\Sigma}$. Additionally, we will adopt the monogram notation of (Tedrake 2021) for rigid transforms.

2 Problem Statement

We consider a known, task-space environment where our robot and all obstacles have been decomposed as a union of compact, convex bodies[‡] for example cylinders, capsule, spheres, or vertex representation (V-rep) polytopes. Such collision geometries of our task space are readily available through standard tools such as V-HACD (Mamou and Ghorbel 2009) and are often a required step for simulating any given environment.

Our robot is a mechanism composed of $N+1$ links connect via either revolute or prismatic joints (Wampler and Sommese 2011):

- Revolute (R): a 1-DOF joint permitting revolution about an axis of symmetry. An example is a door handle.
- Prismatic (P): a 1-DOF joint permitting translation along an axis. An example is a linear rail.

We will assume that all revolute joints are constrained from undergoing complete rotations and all prismatic joints have bounded translation. Formally, if θ is the configuration-space variable associated to revolute joint, then:

$$-\pi < \theta_l \leq \theta \leq \theta_u < \pi, \tag{1}$$

and if z is the configuration-space variable associated to a displacement then:

$$z_l \leq z \leq z_u. \tag{2}$$

where the bounds θ_l, θ_u, z_l, and z_u are fixed constants.

Our objective is to find large, convex regions of TC-free regardless of the dimension of the configuration space. This objective is beyond the scope of current decomposition for non-convex spaces/objects such as V-HACD due to the dimensionality of the problem for interesting robots and the complexity of the non-linear kinematics.

Remark 1. *Our approach can handle a robot composed of any of the five algebraic joints: revolute, prismatic, planar, cylindrical, planar, and spherical (Wampler and Sommese 2011). We restrict ourselves to R and P joints as the other joints can be seen as a composition of these two (see appendix A for details).*

3 Background

This section introduces key notions from convex analysis and algebraic geometry that will be essential for our approach presented in Section 4. We begin by recalling some classic theorems pertaining to

[‡]For technical reasons, we formally assume that the bodies are compact, convex sets expressible as a Archimedean, basic semi-algebraic sets. See Appendix B for the definition of Archimedean

the separation of convex bodies. We next review the Positivstellensatz, a central theorem from algebraic geometry that forms the basis for many applications of the Sums-of-Squares method that we will leverage. We conclude by recalling a parameterization of a robot's forward kinematics using rational functions.

3.1 Separating Convex Bodies

In this section, we review two dual ways to check whether two compact, convex sets $\mathcal{A}$ and $\mathcal{B}$ intersect by using convex optimization. Our certification programs in Section 4 will rely on generalizations of the programs introduced in this section.

A well-known result from convex optimization theory is the Separating Hyperplane Theorem (Boyd et al. 2004, Section 2.5) which states that $\mathcal{A}$ and $\mathcal{B}$ do not intersect, if and only if there exists a hyperplane $\mathcal{H}(a, b) = \{x \mid a^T x + b = 0, (a, b) \neq (0, 0)\}$ which strictly separates the two bodies. The hyperplane $\mathcal{H}(a, b)$ serves as a *certificate* of non-intersection. Such a hyperplane is visualized in Figure 1a and is described by the solution to program (3). Many previous works (Brossette and Wieber 2017; Lin et al. 2022) have applied the Separating Hyperplane Theorem to find a *single* collision-free posture; in this paper we apply the theorem to find a convex set of collision-free postures.

Conversely, if $\mathcal{A}$ and $\mathcal{B}$ do intersect, then it is possible to certify this by finding a point in $\mathcal{A} \cap \mathcal{B}$. Such a point can be found by solving the convex optimization program (4). A certificate of the *infeasibility* of (4) proves that $\mathcal{A}$ and $\mathcal{B}$ do not intersect. Finding a certificate of infeasibility can be obtained by considering the dual of (4) and is a standard notion in convex optimization (Boyd et al. 2004, Section 5.8).

A solution to program (3) has the advantage of being able to quantify the magnitude of separation between the two bodies. Therefore, in Section 5 we will prefer to base our algorithm on a generalization of (3). However, we will see that certain results will be easier to show by considering the *infeasibility* of program (4).

We conclude by noting that programs (3) and (4) are *strong alternatives*; exactly one of the two programs is feasible. The key to solving either program is to find a finite parameterization of conditions (3a), (3b), and (4a). In Table 1, we provide a convenient reference for some common geometries.

Remark 2. *Problem (3) is frequently written with non-strict inequalities (3a) and (3b) to make it compatible with modern solvers. Such a formulation requires excluding the trivial solution $(a, b) = (0, 0)$ via extra constraints as well as planes which are not strictly separating. The conditions given in Table 1 accomplish both with the constraint $a^T x + b \geq 1$ with $x = v_i$ for polytopic geometries and $x = o$ for sphere, cylinder, and capsules.*

3.2 Certificates of Positivity and Infeasibility

In Section 4, we will show how to generalize programs (3) and (4) to be able to certify non-collision for a range of robot configurations. Both generalizations will reduce to well-studied polynomial problems. Specifically, given the set

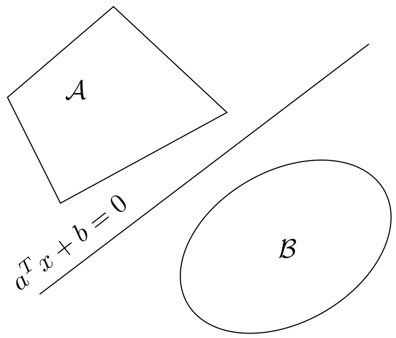
$$\mathcal{S}_{g,h} = \{x \mid g_i(x) \geq 0, h_j(x) = 0, i \in [n], j \in [m]\},$$

[§]Strictly speaking, the formulation (3a) given for the sphere, capsule, and cylinder only enforce non-strict separation i.e. $a^T x + b \geq 0$. This can be remedied by replacing r with $r + \varepsilon$ for any $\varepsilon > 0$.

Find $a,\ b$

$$a^T x + b > 0,\ \forall\, x \in \mathcal{A} \tag{3a}$$

$$a^T y + b < 0,\ \forall\, y \in \mathcal{B} \tag{3b}$$

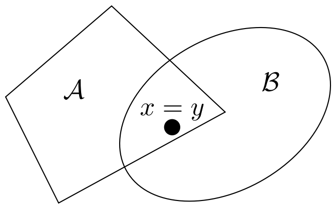


(a) If $\mathcal{A} \cap \mathcal{B} = \emptyset$ then there exists a hyperplane $a^T x + b = 0$ which separates the two bodies.

Find $x,\ y$ **subject to**

$$x \in \mathcal{A},\ y \in \mathcal{B} \tag{4a}$$

$$x = y \tag{4b}$$



(b) If $\mathcal{A} \cap \mathcal{B} \neq \emptyset$ then there exists $x \in \mathcal{A}$ and $y \in \mathcal{B}$ such that $x = y$

Figure 1. Program (3) searches for a hyperplane which separates $\mathcal{A}$ and $\mathcal{B}$ while program (4) searches for a point in $\mathcal{A} \cap \mathcal{B}$. Both of these are convex optimization programs, and exactly one of these programs is feasible.

where $g_i(x)$ and $h_j(x)$ are all given polynomial functions of x, then certifying the separating hyperplane conditions (3a) and (3b) will be akin to a certifying a polynomial implication of the form

$$x \in \mathcal{S}_{g,h} \implies p(x) \geq 0 \tag{5}$$

where $p(x)$ is again a polynomial.

Moreover, certifying the infeasibility of (4) will be akin to certifying that

$$\mathcal{S}_{g,h} = \emptyset. \tag{6}$$

Both of these polynomial problems are tractable. In particular, a class of results known as Positivstellensatz Theorems (Psatz) can be used to reduce both problems to a convex optimization program (Parrilo 2000; Blekherman et al. 2012). In this section, we review the Psatz results that we will use.

Our assumption (1) and (2) that our robot has joint limits implies that the subsets of TC-free we wish to certify will be Archimedean sets, a property slightly stronger than compactness formally defined in Appendix B. This will enable us to use a very strong Psatz Theorem for proving implications of the form (5) known as Putinar's Positivstellensatz.

Theorem 1. Positivstellensatz (Putinar 1993). *Suppose $\mathcal{S}_{g,h}$ is Archimedean and suppose that $p(x) > 0$ for all $x \in \mathcal{S}_{g,h}$. Then there exists polynomials $\phi_j(x),\ j = 0, \dots, m$ and SOS polynomials $\lambda_i(x),\ i =$*

Body	$a^Tx + b > 0, \forall x \in \mathcal{A}$	$x \in \mathcal{A}$
V-rep Polytope with vertices $\{v_1, \ldots, v_m\}$.	$a^Tv_i + b \geq 1,\ \forall\, i \in \{1, \ldots, m\}$	$x = \sum_{i=1}^m \mu_i v_i,\ \sum_{i=1} \mu_i = 1,$ $\mu_i \geq 0$
Sphere with center o and radius r.	$a^To + b \geq r\,\|a\|$ $a^To + b \geq 1$	$\|x - o\|^2 \leq r^2$
Capsule, the convex hull of two spheres with centers o_1 and o_2 and radii r_1 and r_2.	$a^To_1 + b \geq r_1\,\|a\|$ $a^To_2 + b \geq r_2\,\|a\|$ $a^To_1 + b \geq 1$	$o_\mu = \mu o_1 + (1-\mu)o_2$ $\|x - o_\mu\| \leq \mu r_1 + (1-\mu)r_2$ $0 \leq \mu \leq 1$
Cylinder, the convex hull of two circles with centers o_1 and o_2, and with radii r_1 and r_2.	$\frac{a_z\,\|o_1 - o_2\|}{2} + b \geq r_1 \left\| \begin{bmatrix} a_x & a_y \end{bmatrix} \right\|$ $\frac{-a_z\,\|o_1 - o_2\|}{2} + b \geq r_2 \left\| \begin{bmatrix} a_x & a_y \end{bmatrix} \right\|$ $a^T\left(\frac{o_1 + o_2}{2}\right) + b \geq 1$	$o_\mu = \mu o_1 + (1-\mu)o_2$ $v^T(o_1 - o_2) = 0$ $x = o_\mu + v$ $\|v\| \leq \mu r_1 + (1-\mu)r_2$ $0 \leq \mu \leq 1$

Table 1. Parameterizations of conditions (3a) and (4a) respectively for particular convex bodies. §

$0, \ldots, n$ *such that:*

$$p(x) = \lambda_0(x) + \sum_{i=1}^n \lambda_i(x)g_i(x) + \sum_{j=1}^m \phi_j(x)h_j(x). \tag{7}$$

Moreover, if $p(x)$ *is any polynomial that can be expressed as in* (7)*, then*

$$x \in \mathcal{S}_{g,h} \implies p(x) \geq 0 \tag{8}$$

As an immediate corollary, the previous theorem can be used to prove that $\mathcal{S}_{g,h}$ is empty.

Theorem 2. (Parrilo 2004). *Suppose $\mathcal{S}_{g,h}$ is Archimedean. Then $\mathcal{S}_{g,h} = \emptyset$ if and only if there exists polynomials $\phi_j(x)$ and SOS polynomials $\lambda_i(x)$ such that*

$$-1 = \lambda_0(x) + \sum_i \lambda_i(x) g_i(x) + \sum_j \phi_j(x) h_j(x). \tag{9}$$

In both cases, the multiplier polynomials λ and ϕ serve as *certificates* that the conditions (5) or (6) hold. These certificates can be searched for using a convex optimization technique known as Sums-of-Squares (SOS) programming, a subset of semidefinite programming (SDP) (Parrilo 2000). The SOS technique has been widely used in robotics, for example in stability verification (Tedrake et al. 2010; Majumdar and Tedrake 2017; Shen and Tedrake 2020), reachability analysis (Jarvis-Wloszek et al. 2003; Yin et al. 2021) and geometric modeling (Ahmadi et al. 2016). In this paper, we will use SOS programming to generate certificates that subsets of TC-space are contained in TC-free.

3.3 Rational Forward Kinematics

Our method in Section 4 will rely critically on parameterizing the forward kinematics of our robot using polynomials. Many robots contain rotational joints and so their forward kinematics are naturally specified as trigonometric functions. In this section, we review a standard change of variables of our robot kinematics which will enable us to parameterize the forward kinematics as a rational function.

The forward kinematics of a rigid-body robot with N joints can be written by composing rigid transforms (Craig 2005; Tedrake 2021). Written in homogeneous coordinates, and using the monogram notation Tedrake (2021)[¶], the pose of a frame A, expressed in the reference frame F, as a function of the robot configuration q assumes the form:

$$^{F}X^{A} = \begin{bmatrix} ^{F}R^{A}(q) & ^{F}p^{A}(q) \\ 0_{1\times 3} & 1 \end{bmatrix} = \prod_{i \in \mathcal{I}_{F,A}} {}^{P_i}X^{C_i}(q_i)\, {}^{C_i}X^{P_{i+1}} \tag{10}$$

In equation (10), $\mathcal{I}_{F,A} = \{i_1, \dots, i_n\} \subseteq [N]$ is the set of joints lying on the kinematic chain between F and A. We attach two frames to each joint, with P_i rigidly fixed to the parent link of the i^{th} joint, and C_i rigidly fixed to the child link of the same joint. The two frames P_i and C_i coincide when the joint configuration $q_i = 0$. The subset of configuration variables q_i defines the degrees of freedom at the i^{th} joint, ${}^{P_i}X^{C_i}(q_i)$ is the relative transform of the joint after the joint moves by q_i. The rigid transform ${}^{C_i}X^{P_{i+1}}$ describes the physical properties of the i^{th} link such as its length. We assume that the reference frame F is the P_{i_1}, the parent frame of the first joint i_1; while the frame A is C_{i_n}, the child frame of the last joint i_n. [‖] We choose to be explicit about the reference frame F at the risk of being pedantic, as the choice of reference frame F will have important consequences for the scalability of the approach described in Section 5 (see Appendix F.1 for a detailed discussion).

[¶]In monogram notation, the pose of a frame A expressed in a frame F is denoted as ${}^{F}X^{A}$.

[‖]Since joint i_n is the last joint on this chain $\mathcal{I}_{F,A}$, we assume ${}^{C_{i_n}}X^{P_{i_{n+1}}} = I$.

The matrices ${}^{P_i}X^{C_i}(q_i)$ assume the following forms (Wampler and Sommese 2011)

$$
{}^{P_i}X^{C_i}(q_i) = \begin{cases} \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) & 0 & 0 \\ \sin(\theta_i) & \cos(\theta_i) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} & \text{if } i^{\text{th}} \text{ joint is Revolute} \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & z_i \\ 0 & 0 & 0 & 1 \end{bmatrix} & \text{if } i^{\text{th}} \text{ joint is Prismatic} \end{cases} \tag{11}
$$

Expression (10) expresses the position of our robot as an *multilinear trigonometric polynomial function*. Concretely, the w^{th} component (where $w \in \{x, y, z\}$) of the position of A relative to F and expressed in F is an expression of the form:

$$
{}^{F}p_w^A(q) = \sum_j c_{jw} \prod_{i \in \mathcal{I}_{F,A}} \xi_{ij,w}(q_i) \tag{12}
$$

with $\xi_{ij,w}(q_i) \in \{\cos(\theta_i), \sin(\theta_i), z_i\}$. The scalar constants c_{jw} are determined by the robot kinematic parameters (link length, joint axis, etc). Therefore, our configuration-space variables are

$$
q = \bigcup_i \{\theta_i, z_i\}.
$$

Multilinear trigonometric functions have many fortunate algebraic properties which we exploit throughout this paper, the first of which will be a change of variables enabling us to write (12) as a rational function.

Specifically, we will introduce the substitution:

$$
t_i \coloneqq \tan\left(\frac{\theta_i}{2}\right), \tag{13}
$$

which allows us to write

$$
\cos(\theta_i) = \frac{1 - t_i^2}{1 + t_i^2}, \quad \sin(\theta_i) = \frac{2t_i}{1 + t_i^2}.
$$

This substitution is known as the stereographic projection (Spivak 1994) and is bijective if $\theta_i \in (-\pi, \pi)$ which we have assumed is the case for our robotic system[**]. After performing this change of variables,

[**]An alternative approach is to write the forward kinematics $p_w(q)$ as a multilinear polynomial of indeterminates $c_i = \cos(\theta_i)$ and $s_i = \sin(\theta_i)$, with the additional constraints $c_i^2 + s_i^2 = 1$. We don't choose this parameterization as it is hard to integrate the volume on the quotient ring $c_i^2 + s_i^2 = 1, \ \forall i$. Also this parameterization requires introducing two variables c_i, s_i for each revolute joint, rather than one variable t_i.

our forward kinematics variables are

$$s = \bigcup_i \{t_i, z_i\}.$$

We refer to the configuration-space variable s as the *tangent-configuration-space* (TC-space) variable.

In the TC-space variable, our forward kinematics are a *rational function* with a polynomial numerator and positive, polynomial denominator. This is an expression of the form

$$ {}^F p^A_w(s) = \sum_j c_{jw} \prod_{i \in \mathcal{I}_{F,A}} \frac{{}^F f^A_{ij,w}(s_i)}{{}^F g^A_{ij,w}(s_i)} = \frac{{}^F f^A_w(s)}{{}^F g^A_w(s)}, \quad w \in \{x, y, z\}, \tag{14}$$

where

$$\frac{{}^F f^A_{ij,w}(s_i)}{{}^F g^A_{ij,w}(s_i)} \in \left\{ \frac{1-t_i^2}{1+t_i^2}, \frac{2t_i}{1+t_i^2}, \frac{z_i}{1} \right\}.$$

We will abbreviate the vector quantity:

$${}^F p^A(s) = \frac{{}^F f^A(s)}{{}^F g^A(s)} \tag{15}$$

where ${}^F f^A(s)$ is a *vector of polynomials* and ${}^F g^A(s)$ is a single, positive polynomial. Notice that ${}^F g^A(s) > 0$ since each denominator ${}^F g^A_{ij,w}(s_i) = 1 + t_i^2$ or 1, which is strictly positive.

We emphasize again that we have assumed:

$$-\pi < \theta_{l,i} \leq \theta_i \leq \theta_{u,i} < \pi,$$
$$z_{l,i} \leq z_i \leq z_{u,i}.$$

and therefore generically $s_l \leq s \leq s_u$ component-wise.

Therefore, our substitution between q and s is bijective and so trajectories in TC-space correspond unambiguously to trajectories in C-space. Moreover, this assumption on boundedness of our configuration space allows us to seek collision-free regions $\mathcal{P}$ that are contained within $\mathcal{P}_{lim}$, a polytope encoding our joint limit: $\mathcal{P} \subseteq \mathcal{P}_{lim} = \{s \mid s_l \leq s \leq s_u\}$.

Example 1. *As an example, we consider the double pendulum (Tedrake 2022).*

The pose of the tip of the second pendulum can be written as:

$$\left[\begin{array}{c|c} R(\theta) & \begin{matrix} p_x(\theta) \\ p_y(\theta) \end{matrix} \\ \hline 0 & 1 \end{array}\right] = \begin{bmatrix} \cos(\theta_1) & \sin(\theta_1) & 0 \\ \sin(\theta_1) & -\cos(\theta_1) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & l_1 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} \cos(\theta_2) & \sin(\theta_2) & 0 \\ \sin(\theta_2) & -\cos(\theta_2) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & l_2 \\ 0 & 0 & 1 \end{bmatrix}$$

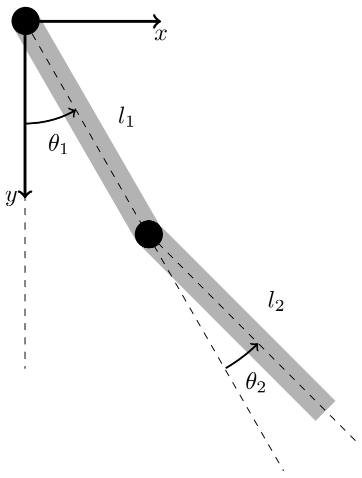


Figure 2. The forward kinematics of the double pendulum described in (Tedrake 2022) can be described in the form (10).

The difference in the sign of the trigonometric part ensures that the y-axis is pointing down. Expanding out this product enables us to write the x coordinate of the tip of the system as:

$$p_x(\theta_1, \theta_2) = l_2(\sin(\theta_2)\cos(\theta_1) - \sin(\theta_1)\cos(\theta_2)) + l_1 \sin(\theta_1)$$
$$p_y(\theta_1, \theta_2) = l_2(\sin(\theta_1)\sin(\theta_2) + \cos(\theta_1)\cos(\theta_2)) - l_1 \cos(\theta_1)$$

Notice that these are multilinear trigonometric polynomials, i.e. no term contains $\cos(\theta_i)\sin(\theta_i)$. We can perform the substitution given in (13) to express the position as a rational function:

$$p_x(t_1, t_2) = \frac{2l_2(t_2(1-t_1)^2 - t_1(1-t_2)^2) + 2l_1 t_1(1+t_2)^2}{(1+t_1^2)(1+t_2^2)}$$
$$p_y(t_1, t_2) = \frac{l_2(4t_1t_2 + (1-t_1)^2(1-t_2)^2) - l_1(1-t_1)^2(1+t_2)^2}{(1+t_1)^2(1+t_2)^2}$$

.

4 Certification of Set-Membership in TC-Free

In this section, we will consider the problem of certifying the non-collision of two convex bodies $\mathcal{A}$ and $\mathcal{B}$ whose poses in task space are a function of the configuration of our robot. While programs (3) and (4) can be used to certify non-collision between $\mathcal{A}$ and $\mathcal{B}$ for any fixed configuration, they are insufficient to certify $\mathcal{A}$ and $\mathcal{B}$ do not intersect for all configurations in an entire region $\mathcal{P}$ of the configuration space. Therefore, in Sections 4.1 and 4.2, we will show how to combine the ingredients of Section 3 to generalize programs (3) and (4).

The presence of trigonometric functions when the forward kinematics are expressed in the variable q precludes using SOS programming, our tool of choice. Therefore, we will assume that $\mathcal{A}(s)$ and $\mathcal{B}(s)$

are convex sets in task space with their poses expressed as *rational functions* in the TC-space variable s. This can be achieved using the developments in Section 3.3. Our objective will be to certify that $\mathcal{A}(s)$ and $\mathcal{B}(s)$ do not intersect for all $s \in \mathcal{P} = \{s \mid Cs \leq d\} \subseteq \mathcal{P}_{lim} = \{s \mid s_l \leq s \leq s_u\}$.

Under these assumptions, the generalizations of (3) and (4) will respectively take the form of certifying a polynomial implication and certifying the emptiness of a basic-semialgebraic set. We give a formulation of each as a SOS program. We will conclude in Section 4.3 by proving that feasibility of our convex optimization programs is both necessary and sufficient for $\mathcal{P}$ to be collision-free.

4.1 Parametrized Hyperplane Certificates of Non-Collision

In this section, we generalize (3) and use SOS to search for a polynomial family of hyperplanes parametrized by the TC-space variable s which will certify the non-collision of $\mathcal{A}(s)$ and $\mathcal{B}(s)$ for all $s \in \mathcal{P} = \{s \mid Cs \leq d\}$.

We begin by remarking that even if $\mathcal{A}(s)$ and $\mathcal{B}(s)$ do not collide for all $s \in \mathcal{P}$, there may not be a single, static hyperplane $\mathcal{H} = (a, b)$ which certifies this fact. An example of this can be seen in Figure 3.

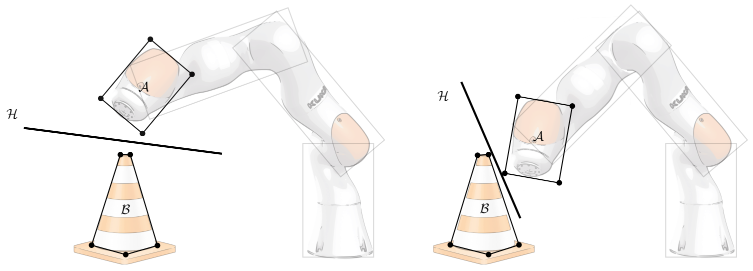


Figure 3. The convex collision geometries $\mathcal{A}(s)$ and $\mathcal{B}(s)$ are collision-free if and only if there exists a family of hyperplanes $\mathcal{H}(s)$ separating the two for each configuration s_0. The planes act as a certificate of non-collision.

We therefore will look for a *polynomial family* of hyperplanes $\mathcal{H}(s) = \{x \mid a(s)^T x + b(s) = 0\}$ parametrized by our TC-space variable s. Inspection of Table 1 shows that we must generalize

$$s \in \mathcal{P} \implies a^T(s)\, {}^F p^v(s) + b(s) \geq 1, \tag{16}$$

for particular points v specific to each of the geometries, and

$$s \in \mathcal{P} \implies a^T(s)\, {}^F p^o(s) + b(s) \geq r \left\| a(s) \right\|, \tag{17}$$

for center o if $\mathcal{A}(s)$ is either a sphere or capsule. The generalization of the conditions for the cylinder are similar to those of the sphere and capsule, and so we defer its complete derivation to Appendix D.

To generalize (16) and (17), we recall that the position of any point $A \in \mathcal{A}(s)$ (and similarly $\mathcal{B}(s)$) can be expressed as a rational function ${}^F p^A(s) = \frac{{}^F f^A(s)}{{}^F g^A(s)}$ where ${}^F g^A(s) > 0$.

Therefore, we can express (16) as:

$$s \in \mathcal{P} \implies a^T(s)\, {}^F f^v(s) + (b(s) - 1)\, {}^F g^v(s) \geq 0 \tag{18}$$

This is an polynomial implication of the form (8). As $\mathcal{P} \subseteq \mathcal{P}_{lim}$ is compact polytope, $\mathcal{P}$ is Archimedean (Marshall 2008, Theorem 7.1.3) and so we can use Theorem 1 to express condition (16) as:

$$a^T(s)\ {}^F f^v(s) + (b(s) - 1)\ {}^F g^v(s) = \lambda_{01}(s) + \sum_{j=1}^{m} \lambda_{j1}(s)(d_j - c_j^T s) \tag{19}$$

where $\lambda_{j1}, j = 0, \dots, m$ are all SOS polynomials.

The condition (17), can be expressed as a polynomial, matrix inequality using the Schur complement[††] (Boyd et al. 2004)

$$s \in \mathcal{P} \implies \begin{bmatrix} \left((a(s))^T\ {}^F f^o(s) + b(s)\ {}^F g^o(s)\right) I_3 & ra(s)\ {}^F g^o(s) \\ r(a(s))^T\ {}^F g^o(s) & (a(s))^T\ {}^F f^o(s) + b(s)\ {}^F g^o(s) \end{bmatrix} \succeq 0. \tag{20}$$

This is known as a *matrix SOS* condition which can be represented as a set of semidefinite constraints (Nie 2011). Specifically, by introducing a vector auxillary variable u, we can write (20) as:

$$s \in \mathcal{P}, u^T u = 1 \implies$$
$$u^T \begin{bmatrix} \left(a^T(s)\ {}^F f^o(s) + b(s)\ {}^F g^o(s)\right) I_3 & ra(s)\ {}^F g^o(s) \\ r(a(s))^T\ {}^F g^o(s) & (a(s))^T\ {}^F f^o(s) + b(s)\ {}^F g^o(s) \end{bmatrix} u \geq 0 \tag{21}$$

which can be expressed as the SOS condition:

$$u^T \begin{bmatrix} \left((a(s))^T\ {}^F f^o(s) + b(s)\ {}^F g^o(s)\right) I_3 & ra(s)\ {}^F g^o(s) \\ r(a(s))^T\ {}^F g^o(s) & (a(s))^T\ {}^F f^o(s) + b(s)\ {}^F g^o(s) \end{bmatrix} u =$$
$$\lambda_{02}(u, s) + \sum_{j=1}^{m} \lambda_{j2}(u, s)(d_j - c_j^T s) + \phi(u, s)(1 - u^T u) \tag{22}$$

where λ_{j2} are all SOS polynomials, and $\phi \in \mathbb{R}[u, s]$. We introduce the additional equality $u^T u = 1$ to make the set $\{(u, s) | s \in \mathcal{P}, u^T u = 1\}$ an Archimedean set.

We are now ready to describe our convex program certifying that $\mathcal{P}$ is a region of TC-space containing no collision. For each pair of bodies $\mathcal{A}(s)$ and $\mathcal{B}(s)$ which can collide in the scene, we search for a polynomial hyperplane via the optimization program:

$$\forall \text{ pairs } \mathcal{A}, \mathcal{B} \textbf{ Find } a_{\mathcal{A},\mathcal{B}}, b_{\mathcal{A},\mathcal{B}} \textbf{ subject to} \tag{23a}$$
$$\forall\ s \in \mathcal{P},\ a^T_{\mathcal{A},\mathcal{B}}(s)x + b_{\mathcal{A},\mathcal{B}}(s) > 0,\ \forall x \in \mathcal{A}(s) \tag{23b}$$
$$\forall\ s \in \mathcal{P},\ a^T_{\mathcal{A},\mathcal{B}}(s)y + b_{\mathcal{A},\mathcal{B}}(s) < 0\ \forall y \in \mathcal{B}(s) \tag{23c}$$
$$\lambda^{\mathcal{A},\mathcal{B}}_{ij}(u, s),\ \mu^{\mathcal{A},\mathcal{B}}_{ij}(u, s) \in \mathbf{\Sigma},\ \phi^{\mathcal{A},\mathcal{B}}(u, s),\ \chi^{\mathcal{A},\mathcal{B}}(u, s) \in \mathbb{R}[u, s] \tag{23d}$$

[††]We have that $\gamma \geq r \|a\|$ if and only if the Schur complement $\begin{bmatrix} \gamma I_3 & ra \\ ra^T & \gamma \end{bmatrix} \succeq 0$.

where $(a_{\mathcal{A},\mathcal{B}}(s), b_{\mathcal{A},\mathcal{B}}(s))$ are the parameters of the polynomial hyperplane separating $\mathcal{A}$ and $\mathcal{B}$, the polynomials $\lambda_{ij}^{\mathcal{A},\mathcal{B}}(s)$ and $\phi^{\mathcal{A},\mathcal{B}}(s)$ collect all the multiplier polynomials for enforcing (23b), and $\mu_{ij}^{\mathcal{A},\mathcal{B}}(s)$ and $\chi^{\mathcal{A},\mathcal{B}}(s)$ collect all the multiplier polynomials for enforcing (23c) by using (19) and (22) depending on the geometry of $\mathcal{A}$ and $\mathcal{B}$. We stress in the above program that the decision variables are the *coefficients* of the polynomials $a_{\mathcal{A},\mathcal{B}}$, $b_{\mathcal{A},\mathcal{B}}$, and the multiplier polynomials. The symbols u and s are known as *indeterminates* and are not explicitly searched over.

In Table 2, we summarize the conditions for enforcing (23b) and (23c) for common families of sets. We call a feasible solution to (23) a *certificate* for the polytope $\mathcal{P}$ which we denote:

$$\mathcal{C}_\mathcal{P} = \bigcup_{(\mathcal{A},\mathcal{B})} \{a_{\mathcal{A},\mathcal{B}}(s),\ b_{\mathcal{A},\mathcal{B}}(s),\ \lambda_{ij}^{\mathcal{A},\mathcal{B}}(u,s),\ \phi^{\mathcal{A},\mathcal{B}}(u,s),\ \mu_{ij}^{\mathcal{A},\mathcal{B}}(u,s),\ \chi^{\mathcal{A},\mathcal{B}}(u,s)\} \tag{24}$$

Body	Psatz Condition for (23b)
V-rep Polytope with m vertices v_i at position ${}^F p^{v_i}(s) = \frac{{}^F f^{v_i}(s)}{{}^F g^{v_i}(s)}$	Enforce (19) for each vertex v_i.
Sphere with center o at position ${}^F p^{o}(s) = \frac{{}^F f^{o}(s)}{{}^F g^{o}(s)}$ and radius r	Enforce (22) for the center o with radius r. Also enforce (19) for the center o.
Capsule, the convex hull of two spheres with centers o_1 and o_2 at positions ${}^F p^{o_i}(s) = \frac{{}^F f^{o_i}(s)}{{}^F g^{o_i}(s)}$ and radii r_1, r_2	For $i \in \{1,2\}$ enforce (22) for center o_i with radius r_i. Also enforce (19) for o_i.
Cylinder, the convex hull of two circles with centers o_1 and o_2, at position ${}^F p^{o_i}(s) = \frac{{}^F f^{o_i}(s)}{{}^F g^{o_i}(s)}$, lying in the plane normal to ${}^F p^{o_1}(s) - {}^F p^{o_2}(s)$, and with radii r_1 and r_2.	See Appendix D.

Table 2. SOS conditions for the constraint (23b) and (23c) depending on the geometry of bodies $\mathcal{A}$ and $\mathcal{B}$.

4.2 Polynomial Infeasibility Certificates

As we remarked in section 3.1, non-collision of two convex shapes $\mathcal{A}$ and $\mathcal{B}$ can be checked by certifying the *infeasibility* of (4). The infeasibility of (4) can be extended to the case when the locations of $\mathcal{A}(s)$

and $\mathcal{B}(s)$ are a function of s.

$$\textbf{Certify that } \nexists\, s \in \mathcal{P},\ x, y \in \mathbb{R}^3 \textbf{ such that} \tag{25a}$$
$$x \in \mathcal{A}(s), y \in \mathcal{B}(s) \tag{25b}$$
$$x = y \tag{25c}$$

An equivalent, and perhaps more instructive, way of expressing (25) is to consider the set

$$\mathcal{S}_{\mathcal{P},\mathcal{A},\mathcal{B}} = \{x, s \mid s \in \mathcal{P},\ x \in \mathcal{A}(s),\ x \in \mathcal{B}(s)\} \tag{26}$$
$$= \left\{ x,\ s,\ u_{\mathcal{A}},\ u_{\mathcal{B}} \;\middle|\; \begin{array}{c} Cs \le d, \\ \gamma_i^{\mathcal{A}}(s, x, u_{\mathcal{A}}) \ge 0,\ h_j^{\mathcal{A}}(s, x, u_{\mathcal{A}}) = 0 \\ \gamma_k^{\mathcal{B}}(s, x, u_{\mathcal{B}}) \ge 0, h_l^{\mathcal{B}}(s, x, u_{\mathcal{B}}) = 0, \\ i \in [n_{\mathcal{A}}],\ j \in [m_{\mathcal{A}}],\ k \in [n_{\mathcal{B}}],\ l \in [m_{\mathcal{B}}] \end{array} \right\} \tag{27}$$

and to consider the problem

$$\textbf{Certify that } \mathcal{S}_{\mathcal{P},\mathcal{A},\mathcal{B}} = \emptyset, \tag{28}$$

In (27), $\gamma_i^{\mathcal{A}}(s, x, u_{\mathcal{A}})$ and $h_j^{\mathcal{A}}(s, x, u_{\mathcal{A}})$ are the polynomials encoding the condition that $x \in \mathcal{A}(s)$ and $u_{\mathcal{A}}$ collects any extra variables needed to write this condition. Similarly, $u_{\mathcal{B}}$, $\gamma_k^{\mathcal{B}}(s, x, u_{\mathcal{B}})$, and $h_l^{\mathcal{B}}(s, x, u_{\mathcal{B}})$ encode that $x \in \mathcal{B}(s)$. We provide explicit expressions for $\gamma_i^{\mathcal{A}}, \gamma_k^{\mathcal{B}}$ and $h_j^{\mathcal{A}}, h_l^{\mathcal{B}}$ in Table 4 (given in Appendix C) for a few common geometries.

Example 2. *If $\mathcal{A}$ is a polytope with $n_{\mathcal{A}}$ vertices given by $v_{\mathcal{A}_i}$, and $\mathcal{B}$ is a sphere with center $o_{\mathcal{B}}$ and radius $r_{\mathcal{B}}$, then we can write*

$$\mathcal{S}_{\mathcal{P},\mathcal{A},\mathcal{B}} = \left\{ x,\ s,\ \mu_{\mathcal{A}_i} \;\middle|\; \begin{array}{c} Cs \le d, \\ \left(\prod_i {}^F g^{v_{\mathcal{A}_i}}\right)\left(x - \sum_{i=1}^{m} \mu_{\mathcal{A}_i}\left(\frac{{}^F f^{v_{\mathcal{A}_i}}(s)}{{}^F g^{v_{\mathcal{A}_i}}(s)}\right)\right) = 0, \\ 1 - \sum_{i=1}^{m} \mu_{\mathcal{A}_i} = 0, \\ \mu_{\mathcal{A}_i} \ge 0\, \forall\, i \in [n_{\mathcal{A}}], \\ \left({}^F g^{o_{\mathcal{B}}}(s)\right)^2 \left(r_{\mathcal{B}}^2 - \left\| x - \frac{{}^F f^{o_{\mathcal{B}}}(s)}{{}^F g^{o_{\mathcal{B}}}(s)} \right\|^2 \right) \ge 0 \end{array} \right\}$$

Now, we note that $\mathcal{S}_{\mathcal{P},\mathcal{A},\mathcal{B}}$ is an Archimedean set. This implies that we can use Theorem 2 to write (28) as an optimization problem. Denoting $u = \{u_{\mathcal{A}}, u_{\mathcal{B}}\}$, this can be written explicitly as

$$\textbf{Find } \lambda_0,\ \lambda_j^{\mathcal{P}},\ \lambda_j^{\mathcal{A}},\ \lambda_j^{\mathcal{B}},\ \phi_k^{\mathcal{A}},\ \phi_k^{\mathcal{B}} \tag{29a}$$

$$\begin{aligned} -1 = \lambda_0(s,x,u) + \sum_{j=1}^{n} \lambda_j^{\mathcal{P}}(s,x,u)(d_j - c_j^T s) + \\ \sum_{i=1}^{n_{\mathcal{A}}} \lambda_i^{\mathcal{A}}(s,x,u)\gamma_i^{\mathcal{A}}(s,x,u_{\mathcal{A}}) + \sum_{j=1}^{m_{\mathcal{A}}} \phi_j^{\mathcal{A}}(s,x,u) h_j^{\mathcal{A}}(s,x,u_{\mathcal{A}}) + \\ \sum_{l=1}^{n_{\mathcal{B}}} \lambda_l^{\mathcal{B}}(s,x,u)\gamma_l^{\mathcal{B}}(s,x,u_{\mathcal{B}}) + \sum_{k=1}^{m_{\mathcal{B}}} \phi_k^{\mathcal{B}}(s,x,u) h_k^{\mathcal{B}}(s,x,u_{\mathcal{B}}) \end{aligned} \tag{29b}$$

$$\lambda_0,\ \lambda_j^{\mathcal{P}},\ \lambda_i^{\mathcal{A}},\ \lambda_l^{\mathcal{B}} \in \mathbf{\Sigma} \tag{29c}$$

$$\phi_j^{\mathcal{A}},\ \phi_k^{\mathcal{B}} \in \mathbb{R}[s,x,u] \tag{29d}$$

We again emphasize that in program (29) the decision variables are the coefficients of $\lambda_0,\ \lambda_j^{\mathcal{P}},\ \lambda_i^{\mathcal{A}},\ \lambda_l^{\mathcal{B}},\ \phi_j^{\mathcal{A}}$, and $\phi_k^{\mathcal{B}}$, while the symbols $\{x, s, u\}$ are not decision variables but rather polynomial indeterminates. Similar to the program in (23), a certificate of non-collision can be obtained by solving (29) for each pair $(\mathcal{A}, \mathcal{B})$ with the multipliers acting as the certificate.

4.3 Power of the Certification Programs

In this section, we consider the power of both certification programs. Specifically, in Sections 4.1 and 4.2 we argued that feasibility of (23) and (29) are sufficient to prove that $\mathcal{P}$ is collision-free. In this section, we present two theorems showing that the feasibility of these programs is also *necessary*.

Such a result is important given the fact that as stated, (23) and (29) are infinite dimensional and therefore in practice must be solved by selecting a basis of finite degree for the polynomials. Other subtleties about the power of our formulation are discussed in Appendix E. Fortunately, we can prove that there do exist finite degrees such that both programs become feasible when $\mathcal{P}$ is truly collision-free.

Theorem 3. *Let all multiplier polynomials from (23) have degree at least ρ and let all of the polynomials in the parameterization of the hyperplane have degree at least κ. Suppose $\mathcal{P} \subseteq \mathcal{P}_{lim}$ is a subset of TC-free.*

Then there exists finite κ and ρ sufficiently large such that (23) is feasible.

A similar theorem can be stated for the program in (29).

Theorem 4. *Let $\mathcal{P} \subseteq \mathcal{P}_{lim}$ be a compact, polytopic subset of TC-free and let all multiplier polynomials from (29) have degree at least ρ. There exists a finite ρ sufficiently large such that (29) is feasible.*

We delay the proofs and further discussion of these results to Appendix E. For now, we simply remark that Theorems 3 and 4 assert that the certification programs presented in this section are both complete in the sense that any collision-free polytope $\mathcal{P}$ can be certified with our technique.

5 Polyhedral Decomposition of TC-free

In this section, we describe our algorithm for rapidly generating certified, polyhedral decomposition of TC-free. Our algorithm can be seen as a generalization of the IRIS algorithm of (Deits and Tedrake 2015a) to non-convex TC-space obstacles and so we name it C-IRIS (Configuration-Space, Iterative Regional Inflation by Semidefinite programming). The key idea is to iteratively grow certified convex polytopes of increasing size around various important configurations in the TC-space. This is achieved by solving a series of convex optimization programs. The complete algorithm is summarized in Algorithm 1.

We begin by discussing how we will measure the size of our polytope $\mathcal{P} = \{s \mid Cs \leq d\}$. While it may be attractive to measure the size of a polytope by its volume, it is known that computing the volume of a half-space representation (H-Rep) polytope is #P-hard[‡‡] (Dyer and Frieze 1988) and therefore intractable as an objective. A useful surrogate for the volume of $\mathcal{P}$ used in (Deits and Tedrake 2015a) is the volume of the maximum volume inscribed ellipse of $\mathcal{P}$: the set $\mathcal{E}_{\mathcal{P}} = \{Qs + s_0 \mid \|s\|_2 \leq 1\}$ where Q is a positive-semidefinite matrix describing the shape of the ellipsoid and s_0 its center. The problem of finding the maximum volume inscribed ellipsoid in a polytope is a semidefinite program described in (Boyd et al. 2004, Section 8.4.2).

$$\max_{Q, s_0} \ \text{logdet} Q \ \textbf{subject to} \tag{30a}$$
$$\|Qc_i\|_2 \leq d_i - c_i^T s_0 \ \ \forall\, i \in [m] \tag{30b}$$
$$Q \succeq 0 \tag{30c}$$

As we wish our polytopes to cover diverse areas of TC-free, we will grow each polytope $\mathcal{P}$ around some nominal configuration s_s we call the seed point. New seed points are typically chosen using rejection sampling to obtain a point outside of the existing certified regions. The polytope $\mathcal{P}$ is required to contain s_s as it grows.

A maximal volume, certified polytope around s_s can be obtained by solving the following optimization program which combines the ellipsoidal program (30) with the certification program (23) from Section 4.1.

$$\max_{\substack{Q, s_0, C, d, \\ \forall(\mathcal{A},\mathcal{B}) \\ \lambda_{ij}^{\mathcal{A},\mathcal{B}},\ \phi^{\mathcal{A},\mathcal{B}}, \\ \mu_{ij}^{\mathcal{A},\mathcal{B}},\ \chi^{\mathcal{A},\mathcal{B}} \\ a_{\mathcal{A},\mathcal{B}},\ b_{\mathcal{A},\mathcal{B}}}} \ \text{logdet} Q \ \textbf{subject to} \tag{31a}$$
$$(30b), (30c) \tag{31b}$$
$$Cs_s \leq d \tag{31c}$$
$$\|c_i\|_2 \leq 1 \ \forall\, i \in [m] \tag{31d}$$
$$(23b),\ (23c),\ (23d) \tag{31e}$$

[‡‡]#P-hard problems are at least as hard as NP-complete problems (Provan and Ball 1983).

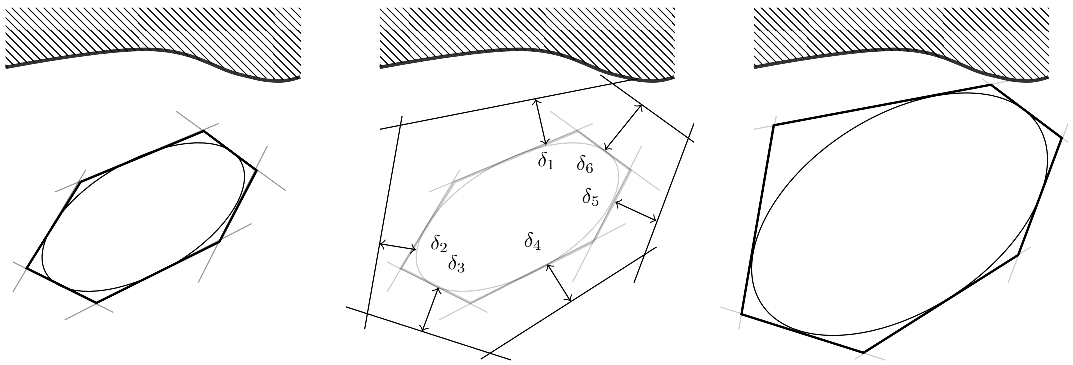


Figure 4. In (32) we search for the maximum amount the polytopes faces can be pushed away from the current inscribed ellipse without violating the certificate found in the previous step.

The condition $\mathcal{E}_\mathcal{P} \subset \mathcal{P}$ is given by the constraints (31b). Constraint (31c) enforces that $\mathcal{P}$ grows around s_s. The added constraint (31d) prevents numerically undesirable scaling. Finally, (31e) enforces that we search for hyperplanes $(a_{\mathcal{A},\mathcal{B}}(s), b_{\mathcal{A},\mathcal{B}}(s))$ which separate each collision pair $\mathcal{A}(s)$ and $\mathcal{B}(s)$.

While this program is attractive as a specification, it is not convex due to bilinearity between Q and c_i in (30b) and the bilinearity between the multipliers and the defining equations of $\mathcal{P}$ implicit in (31e) (see Section 4.1). This bilinearity precludes simultaneous search of the polytope $\mathcal{P}$, inscribed ellipsoid $\mathcal{E}_\mathcal{P}$, and the corresponding certificate $\mathcal{C}_\mathcal{P}$. Therefore, we will approximate the solution to (31) by alternating between two convex programs; one of which will generate certificates of non-collision and one which will improve our polytope without violating the previous certificate.

Remark 3. *It is possible to replace* (31e) *with the equivalent constraints from program* (29)*. We prefer to base our algorithm on* (23) *as it can be visualized (i.e. planes in the task space) and the polynomials contain fewer indeterminates and hence the optimization problem size is smaller. Also the separating planes approach produces separating certificates with quantifiable margins by measuring the distance from the collision geometries to the plane in task space.*

We begin by demonstrating how a certified polytopic region can be improved. Suppose that a convex polytope $\mathcal{P} = \{s | Cs \leq d\}$ has been certified with certificate $\mathcal{C}_\mathcal{P}$ and the maximum inscribed ellipse $\mathcal{E}_\mathcal{P}$ has been computed using (30). A new, larger polytope $\mathcal{P}'$ can be found by solving the convex optimization program (32) which pushes the faces of $\mathcal{P}'$ as far away from the surface of $\mathcal{E}_\mathcal{P}$ without violating the certificate $\mathcal{C}_\mathcal{P}$. This procedure is visualized in Figure 4.

This can be achieved with the following optimization program:

$$\max_{\substack{C,d,\delta,\\ \forall(\mathcal{A},\mathcal{B})\\ \lambda_{01}^{\mathcal{A},\mathcal{B}},\lambda_{02}^{\mathcal{A},\mathcal{B}},\phi^{\mathcal{A},\mathcal{B}}\\ \mu_{01}^{\mathcal{A},\mathcal{B}},\mu_{02}^{\mathcal{A},\mathcal{B}},\chi^{\mathcal{A},\mathcal{B}}\\ a_{\mathcal{A},\mathcal{B}},b_{\mathcal{A},\mathcal{B}}}} \prod_{i=1}^{m}(\delta_i+\varepsilon_0) \textbf{ subject to} \tag{32a}$$

$$\|Qc_i\|_2 \le d_i - \delta_i - c_i^T s_0,\ \delta_i \ge 0\ \forall\, i \in [m] \tag{32b}$$

$$(31c), (31d), (31e)\ \forall \text{pairs } (\mathcal{A}(s), \mathcal{B}(s)) \tag{32c}$$

where $\varepsilon_0 > 0$ is some positive constant ensuring that the objective is never 0. We recall that (32c) is either a constraint of the form (19) or (22). We emphasize that in (32), $\lambda_{i1}, \lambda_{i2}, \mu_{i1}, \mu_{i2}, i \ge 1$ are all fixed and it is the variables c_j and d_j which are searched over.

Algorithm 1: Given an initial polytopic region $\mathcal{P}_0$ and seed point $s_s \in \mathcal{P}_0$ for which (31) is feasible, return a new polytopic region $\mathcal{P}_i$ with a maximal inscribed ellipse $\mathcal{E}_{\mathcal{P}_i}$ with larger volume than $\mathcal{E}_{\mathcal{P}_0}$ and a collision-free certificate $\mathcal{C}_{\mathcal{P}_i}$.

```
1 $i \leftarrow 0$
2 do
3     $\mathcal{C}_{\mathcal{P}_i} \leftarrow$ Solution of (23) with data $\mathcal{P}_i$
4     $\mathcal{E}_{\mathcal{P}_i} \leftarrow$ Solution of (30) with data $\mathcal{P}_i$
5     $(\mathcal{P}_{i+1}, \mathcal{C}_{\mathcal{P}_{i+1}}) \leftarrow$ Solution of (32) with data $(\mathcal{E}_{\mathcal{P}_i}, \mathcal{C}_{\mathcal{P}_i})$
6     $i \leftarrow i+1$
7 while $\left(\textit{vol}(\mathcal{E}_{\mathcal{P}_i}) - \textit{vol}(\mathcal{E}_{\mathcal{P}_{i-1}})\right) / \textit{vol}(\mathcal{E}_{\mathcal{P}_{i-1}}) \ge \textit{tolerance}$;
8 return $(\mathcal{P}_i, \mathcal{C}_{\mathcal{P}_i})$
```

Our complete algorithm proceeds in three steps. First, an initial, collision-free polytope $\mathcal{P}_0$ containing a seed point s_s is certified using (24) to obtain $\mathcal{C}_{\mathcal{P}_0}$. Next, the maximum inscribed ellipsoid $\mathcal{E}_{\mathcal{P}_0}$ is computed using (30). Finally, $\mathcal{P}_0$ is improved using (32) to obtain a new polytope $\mathcal{P}_1$. This polytope $\mathcal{P}_1$ has the same number of defining inequalities as $\mathcal{P}_0$. We iterate this process until the volume of $\mathcal{E}_{\mathcal{P}}$ stops improving. This algorithm is formalized in Algorithm 1. Every step of this process involves solving an *convex program* for which very fast, commercial solvers exist (ApS 2019; Andersen and Andersen 2000).

Remark 4. *Some practical considerations for improving the runtime of Algorithm 1 are discussed in the appendices. Specifically, in Appendix F we expand on design choices which substantially impact the size of the optimization programs as well as which part of Algorithm 1 can be parallelized. Additionally, in Appendix G we discuss a heuristic strategy for proposing a large, initial regions $\mathcal{P}_0$.*

6 Results

We demonstrate the use of Algorithm 1 on systems of varying complexity. We begin with very simple robots where both the task and configuration space can be visualized and demonstrate that our algorithm can find very large portions of TC-space and achieve near-complete coverage for simple systems in reasonable time.

We then demonstrate the use of Algorithm 1 on various robots commonly found in industry. These include a KUKA iiwa reaching into a shelf, a bimanual KUKA iiwa, and similar setups for the Franka UR3. Our objective is show the scalability of our algorithm in realistic settings as well as demonstrate the diversity of shapes our approach can handle.

A mature implementation of our algorithm is available in the open-source robotics toolbox Drake (Tedrake and the Drake Development Team 2019). We furthermore provide examples of our algorithm in interactive Python notebooks. Animations of various figures in this section can also be found on this project's website.

The implementation details of all experiments in this section, such as the choice of reference frame for each plane, the degree of the polynomials parametrizing the hyperplanes, and the degree of the multipliers polynomials in each program are expounded on in Appendix F.

6.1 Simple Robots

In this section, we consider two simple robots each containing only two degrees of freedom. This enables us to visualize both the task space, as well as the configuration space. Though containing few degrees of freedom, each environment maintains rich, realistic collision geometries.

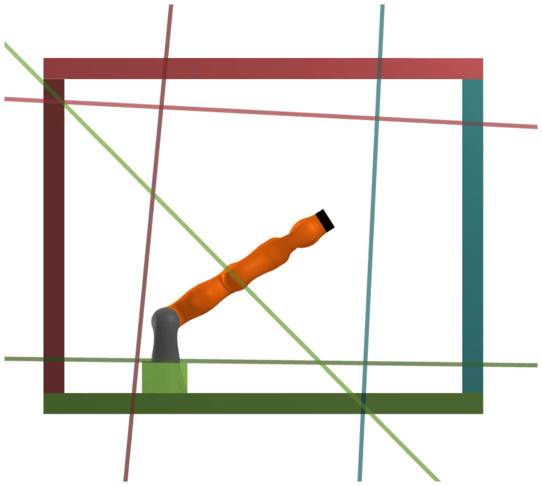
6.1.1 Pendulum on a Rail Our first robot shown in Figure 5a consists of a single arm, shown in orange, connected to a base via a revolute joint and placed within a box. The base of the robot is connected to the box via a prismatic joint. The collision geometries of the robot and box are approximated using polytopic boxes. A total of 42 pairs of geometries can collide in this scene (i.e. certifying non-collision requires solving 42 instances of either (23) or (29)). In Figure 5b, we visualize the two dimensional tangent configuration space of our robot with the TC-space obstacle shown in red. We emphasize the highly non-convex shape of TC-free.

We run Algorithm 1 starting with a regular octagon of side length 0.01 centered at the configuration $(0, 0)$, a configuration with the arm fully extended upwards and centered in the box. We obtain a sequence of certified polytopes of increasing size in the TC-space which are plotted in varying colors in Figure 5b.

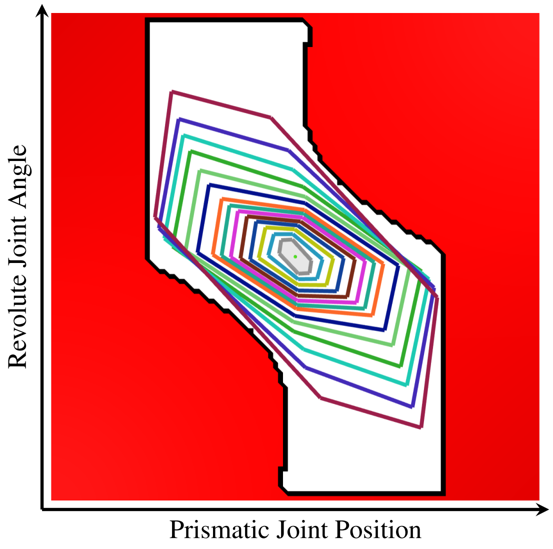
The algorithm terminates after 86 iterations of the while loop from Algorithm 1 taking a total of 314 seconds of wall time. During the course of the algorithm, the volume of the maximum inscribed ellipsoid improves by a factor of 83, from a starting value of 0.021 to 1.746. The improvement in the volume of the inscribed ellipsoid, as well as the average time to solve both the certification program (23) and (32) are reported in Figure 6a and Table 6b respectively.

After completion, we select a single random configuration within our final certified region. In Figure 5, we highlight the tip of the pendulum in black. Additionally, we color each collision body for which the tip can collide in a separate color and plot the separating plane certificate between the tip and the body in the same color.

6.1.2 Pinball Flipper We refer to our second system shown in Figure 7a as the pinball flipper. Each orange arm is connected to its gray base via a revolute joint. Each collision geometry in the scene is approximated with a box and a total of 130 collision pairs exist. We similarly plot the TC-space in Figure 7b with the TC-space obstacle highlighted in red. In this experiment, we attempt to almost completely cover TC-free with polytopic regions in order to enable a motion plan where the flippers



(a) The pendulum on a rail robot. Each hyperplane is a function of the TC-variable s and separates the collision body of the same color from the tip of the robot highlighted in black.



(b) The tangent configuration space of the pendulum on a rail robot. The tangent-configuration-space obstacle is in red. A sample of the polytopes obtained running Algorithm 1 around the configuration $(0, 0)$ are shown.

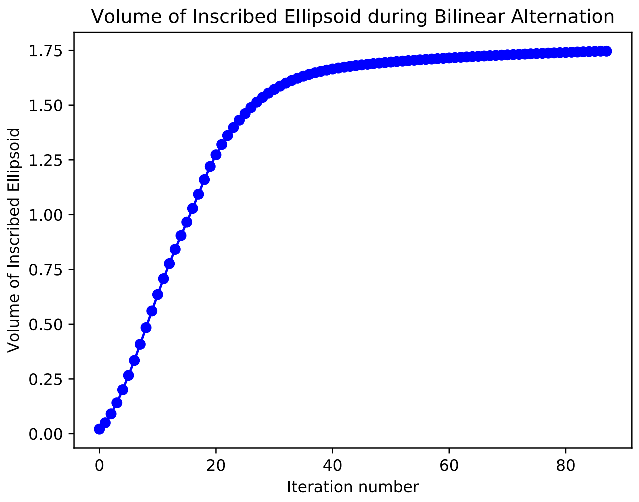
Figure 5. A 2-DOF robot consisting of a revolute joint at the base of the orange link and a prismatic joint between the base and the box.

exchange positions. Overall, this scene exhibits a much more complicated TC-space obstacle as well as substantially more collision pairs when compared to the system from Section 6.1.1.

We run Algorithm 1 seeded with octagonal regions of side length 0.01, each centered at one of 5 different configurations shown as the black dots in Figure 7b. The resulting regions are also plotted in Figure 7b and almost completely cover the space. Though each region was initially seeded with a polytope of the same shape, our algorithm successfully adapts the shape of each polytope to fill the space. Our algorithm also is not conservative; it successfully finding regions which are tight to the TC-space obstacle in all cases.

The change in volume of the maximum inscribed ellipsoid of each region is shown in Figure 8a. We remark that the volume of each region exhibits a diverse set of behaviors over the iterations. Each region was grown sequentially, with a total wall time to cover the space of 1439s. This wall time could easily be improved by growing each region in parallel.

In Figure 9, we demonstrate the behavior of our certificates for various poses of our robot. In the top panel, we highlight in black the two tips of each flipper. The current configuration is highlighted as the green dot in the bottom panel. For each configuration, we also plot the hyperplane that proves the separation between the two black tips. Notice that in Figures 9g, 9h, and 9i, the current configuration is



Number of collision pairs	42
Size of the largest PSD variable	2
Average time to solve (23)	0.191s
Average time to solve (32)	0.423s
Wall time to grow single region	314s

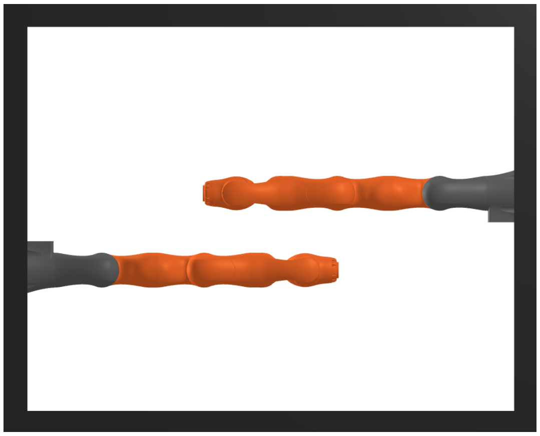
(a) The volume of the maximum inscribed ellipsoids of the TC-free regions shown in Figure 5b is plotted over iterations of Algorithm 1. This volume grows by a factor of 83 over the course of 86 iterations Algorithm 1.

(b) Statistics dominating the run time of Algorithm 1 for the pendulum on a rail system. The complexity scales with the number of collision geometries as well as the size of the largest PSD matrix variable for enforcing the Psatz conditions in Programs (23) and (32).

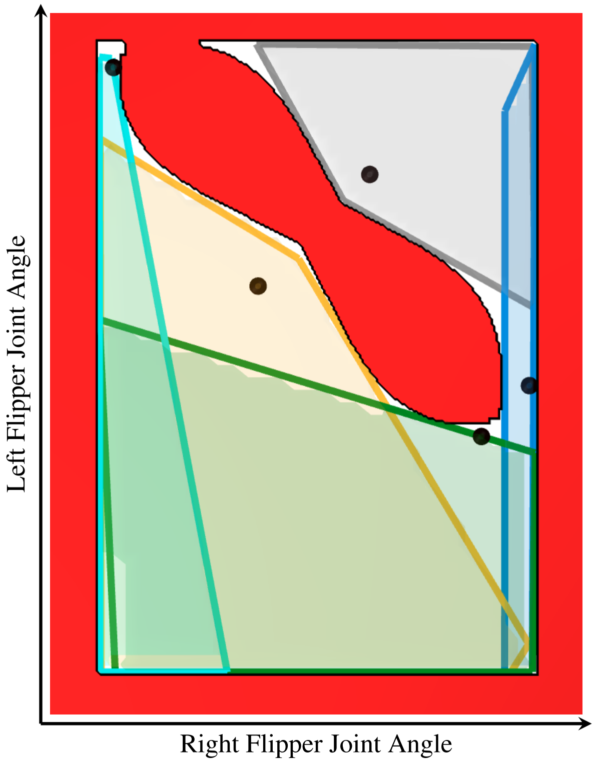
Figure 6. The progress of Algorithm 1 on the pendulum on a rail system for a single polytopic region is plotted. Statistics dominating the run time of the algorithm are also reported.

contained in multiple regions at once. Therefore, each hyperplane in Figure 9a - 9e is drawn in the same color as its associated TC-space region in Figures 9f - 9j.

We draw attention to the fact that at every configuration s_0 in TC-free, many different separating hyperplanes exist. The hyperplane obtained by evaluating the output of our certifier at s_0 is highly dependent on the region which is being certified. For example, in Figure 9h, the blue region corresponds largely to a change in the position of the left flipper, while the green region corresponds largely to a change in the right flipper. We see in Figure 9c, that the algorithm finds different separating planes for the blue and the green region, even for the same configuration, so as to accommodate the different range of robot motion in each region. For the blue region, which includes a large rotation of the left flipper, the blue plane would continue to separate the left flipper from the right flipper as the left flipper moves. Similarly, the green plane would continue to separate the right flipper from the left as the right flipper moves.



(a) The pinball flipper system consists of pendulums each with a revolute joint between the orange link and the gray base. All collision geometries in the scene are approximate using boxes.



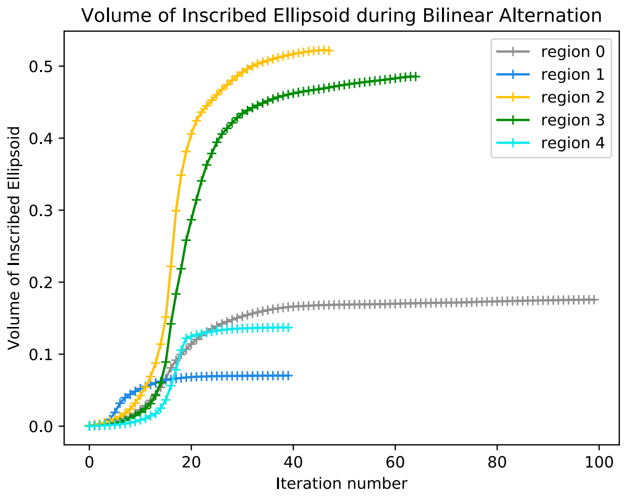
(b) The TC-space of the 2DOF pendulum flipper system. The TC-space obstacle is shown in red. Algorithm 1 is run for five different polytopes each initially centered around the black dots. The polytopes output by the algorithm are plotted in various colors. These polytopes almost fully cover TC-free and are guaranteed to be collision-free by construction.

Figure 7. The pinball flipper system and its TC-space. Algorithm 1 is successfully able to cover TC-free with polytopic regions. An animation of the regions growing to cover this space is available here

6.2 KUKA IIWA robot

In this section we demonstrate our algorithm deployed on the KUKA iiwa arm in two scenes relevant to robot manipulation. The collision geometry of the iiwa is approximated as a union of convex polytopes as are all obstacles in the scene. We begin by considering a single iiwa to demonstrate the practicality of our algorithm before considering a bimanual manipulator to demonstrate the scalability of our approach.

6.2.1 7-DOF IIWA With a Shelf We apply Algorithm 1 to the scene shown in Figure 10: a 7-DOF KUKA iiwa arm reaching into a shelf. Our approach successfully finds many collision-free configurations, and we plot in green the separating hyperplane certificate between the end-effector, highlighted in blue, and the top shelf highlighted in red.



(a) The volume of the maximum inscribed ellipsoid as the polytope is grown around various seedpoints is improved during Algorithm 1. The final polytopes associate to each color are shown in Figure 7b.

Number of collision pairs	130
Size of the largest PSD variable	2
Average time to solve (23)	0.638s
Average time to solve (32)	1.319s
Wall time to grow cover	1439s

(b) Statistics dominating the run time of Algorithm 1 for the pinball flipper system. The complexity scales with the number of collision geometries as well as the size of the largest PSD matrix variable for enforcing the Psatz conditions in Programs (23) and (32).

Figure 8. The progress of Algorithm 1 on the pinball flipper system for each polytopic region is plotted. Statistics dominating the run time of the algorithm are also reported.

The run time of Algorithm 1 is dominated by the certification of non-collision between the pairs with the longest kinematic chain, as this leads to the highest degree polynomials and hence semidefinite variables in programs (23) and (32). For this program, the largest positive semidefinite matrix variable has 16 rows. Overall, the largest certification program (23) takes 54s to solve, while the program (32) takes on average 8s to solve.

In Figure 11, we demonstrate the behavior of one certified region. In Figure 11a, we show that the configurations of one of our certified polytopic region of TC-space (with 24 faces in the polytope) corresponds to many task-space end-effector positions. The configurations from Figure 11a are drawn from a region which grows by a factor of $10,000$ using 11 iterations of Algorithm 1. This improvement in volume is reported in Figure 11b, where we also compare the volume of the maximum volume inscribed ellipsoid against the volume of the polytopic region.

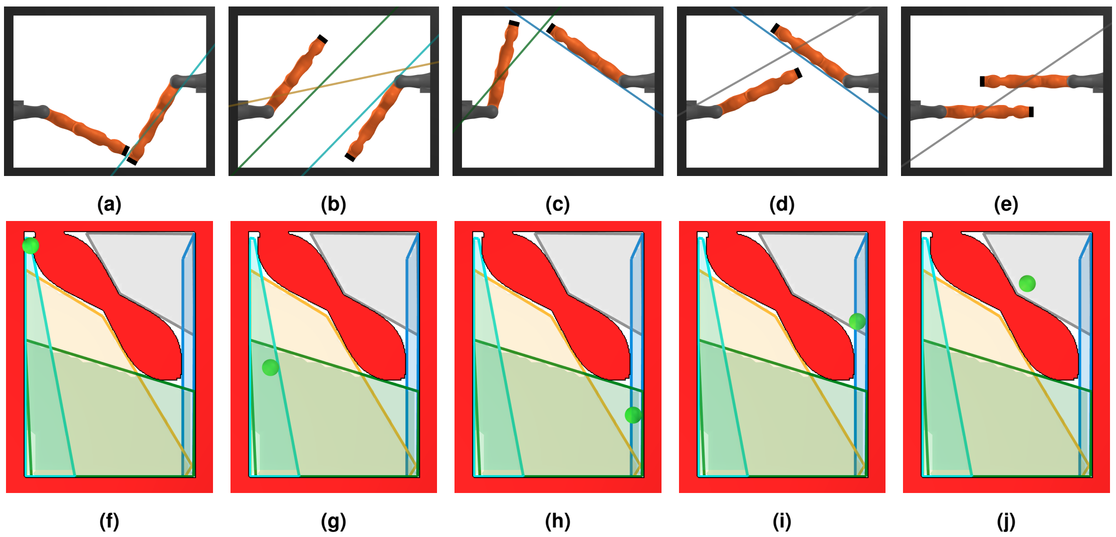


Figure 9. We approximate almost the entirety of TC-free for the robot flipper system using 5 polytopic regions. The top panel shows the hyperplanes certifying that the two black tips of the system do not collide. The bottom panel shows the configuration of the robot as a green dot. An example of this system undergoing a trajectory is available here.

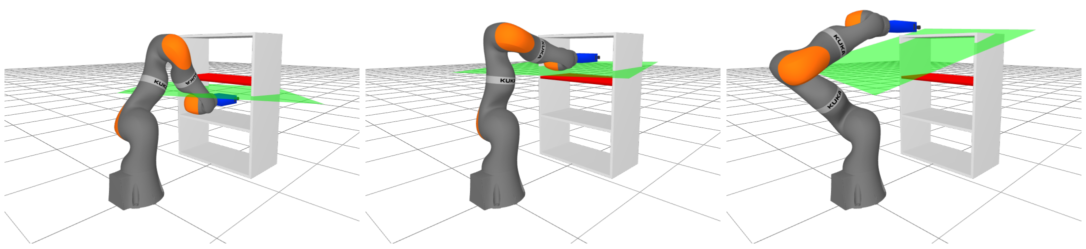
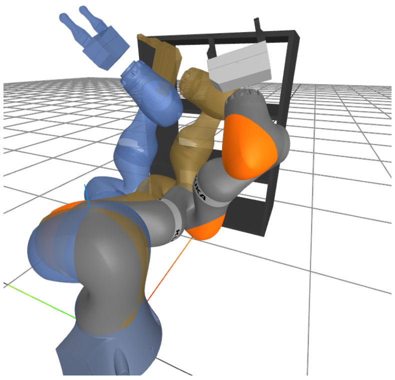


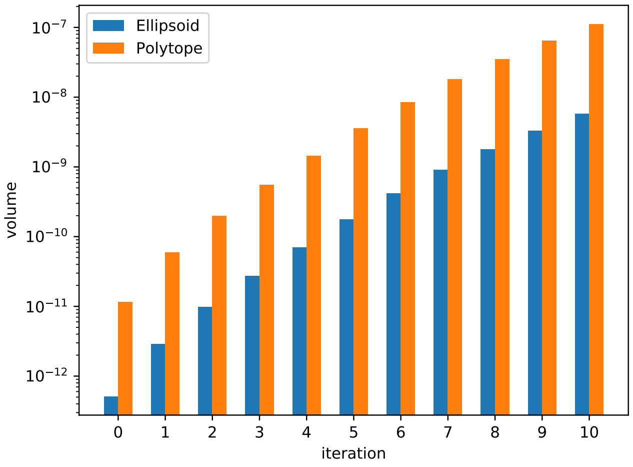
Figure 10. 7-DOF iiwa example. We highlight one pair of collision geometries (blue on robot gripper and red on the shelf), together with their separating plane (green).

6.2.2 12-DOF Bimanual KUKA IIWA Example We next consider designing regions to avoid self-collision for a robot consisting of two KUKA iiwa arms with the final joint welded (rotation of the final joint does not change the configuration of any geometry for this robot). This robot contains 12-DOF. This system tests the scalability of our algorithm due to the degree of the polynomials involved in the forward kinematics, as well as the complexity of the collision geometries.

Solving the largest certification program in (23) takes 105 minutes, while the program in (32) takes 4 minutes. The increase in solve times compared to the single iiwa environment from Section 6.2.1 is best attributed to the increase in the size of the semidefinite variables due to the larger DOF. The largest



(a) The configurations in our certified regions correspond to a wide range of task-space positions. We sample three configurations from the same certified region and plot the corresponding task-space position in different colors.



(b) A single region for the 7-DOF KUKA iiwa is grown over the course of 11 iterations of Algorithm 1. We compare the volume of the maximum volume inscribed ellipsoid to the volume of the polytopic region at each iteration and show that the volume improves by a factor of 10,000.

Figure 11. Algorithm 1 grows certified regions which contain configurations reaching a large portion of the task space. We show that our algorithm is capable of growing the volume of a certified region by a factor 10,000 over the course of just 11 iterations.

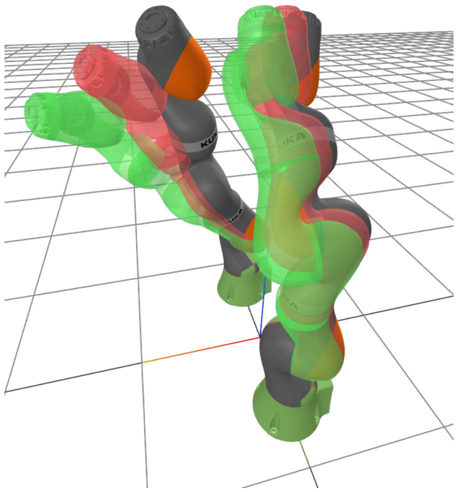
semidefinite matrix in both programs have 64 rows and correspond to certifying that the two tips of the iiwas do not collide.

Nonetheless, our algorithm again finds certified, 30-face polytopic regions of TC-space which correspond to a wide range of task-space positions as seen in Figure 12a. Moreover, the same region is quite tight to the TC-space obstacle; one sampled configuration in the certified region, shown in Figure 12b, corresponds to just 7.3mm of separation between the two arms.

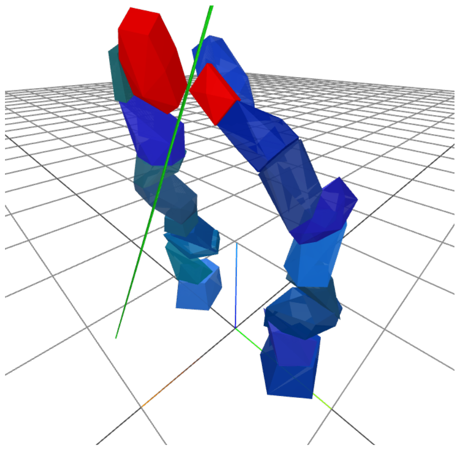
6.3 UR3e Robot

In this section, we test our algorithm on a UR3e robot with a gripper mounted at the wrist. The robot's links are approximated by cylinders and we weld the gripper's prismatic joints so that each UR3e has a total of 6 DOFs. This section differs from the KUKA iiwa experiment in Section 6.2 due to the introduction of non-polytopic collision geometries into the scene. Similar to Section 6.2, we test our approach for a scene where the robot is reaching into a shelf, as well as a bimanual set up.

6.3.1 6-DOF UR3e With a Shelf In Figure 13, we consider a UR3e robot reaching into a shelf to grasp a small box shaped object. To simulate a situation where the robot is attempting to pick up the red



(a) Multiple configurations of the 12-DOF, bimanual iiwa manipulator sampled from a single certified region of TC-free. Each configuration is shown in a separate color.



(b) The geometries of the bimanual iiwa from Figure 12a are tightly approximated using polytopes. At one position in the certified TC-free region, the two geometries highlighted in red are separated by just 7.3mm.

Figure 12. Algorithm 1 finds certified polytopic regions of TC-free even for high DOF systems in reasonable times. The algorithm is also not conservative. It finds large regions which correspond to a broad range of task-space positions. Moreover, the regions are very tight to the TC-space obstacle, finding configurations which lead to very small separation between the task-space objects.

object, we use Algorithm 1 to grow a certified, TC-free polytope (with 12 faces) near the object. Figure 13 shows a variety of postures sampled from the final TC-free polytope and demonstrates that within a single region, our robot is able to reach into the shelf to grasp the object, retract away from the shelf, and maneuver within the shelf while avoiding the object.

Similar to Section 6.2.1, the largest semidefinite variables in programs (23) and (32) has 16 rows with program (23) taking about 56s to solve.

6.3.2 12-DOF Bimanual UR3e Finally, we demonstrate our algorithm on a dual UR3e platform shown in Figure 14. Again, we emphasize that we are able to find large regions of TC-configuration space which correspond to diverse positions in task space with the postures in Figure 14a and 14b being drawn from the same certified region (a 13-face polytope). Moreover, these regions are very tight to the TC-configuration space obstacle with the two bodies highlighted in red in Figure 14b being just 0.3mm apart. For this example, the largest positive semidefinite matrices in (23) and (32) has 128 rows with the largest program taking about 35 minutes to solve. This program solves faster than the analogous program for the bimanual iiwa from Section 6.2.2 because we require fewer polynomial positivity conditions to

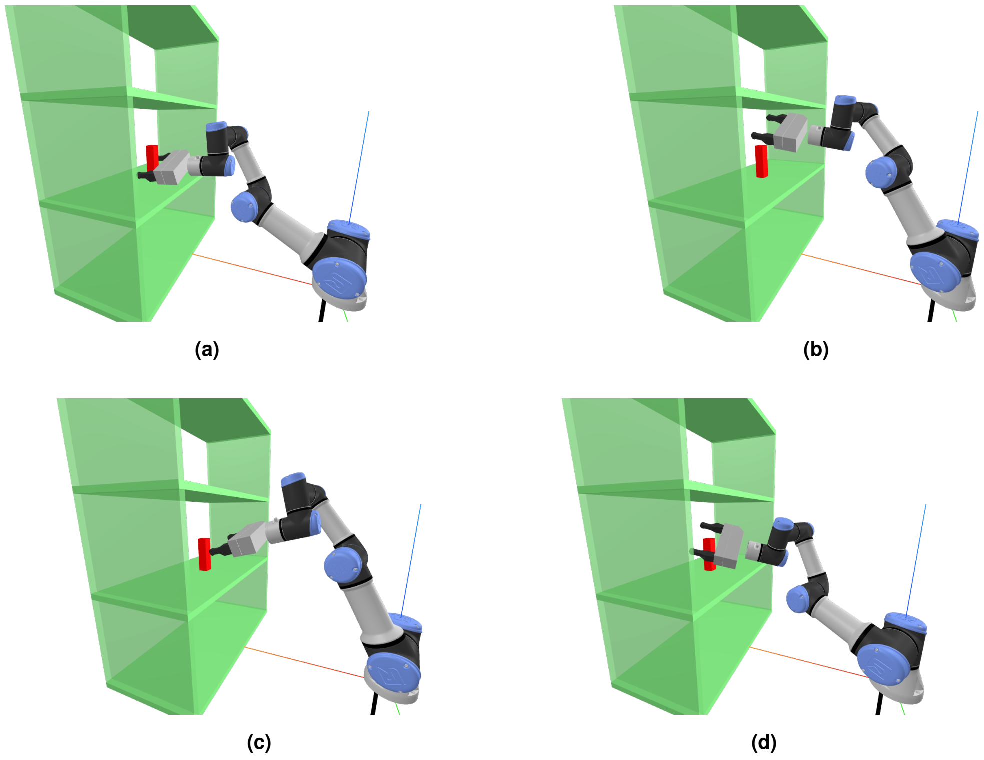


Figure 13. Different postures sampled within one certified TC-space region for a UR3e robot with gripper. The certified-region include both the gripper reaching the red box in the center of the shelf (Fig.13a), retracting from the shelf (Fig.13c), and reaching different regions within the shelf while avoiding the red box (Fig.13b and 13d). An animation of the range of configurations attainable in this region are available here.

certify that the UR3e's cylindrical geometries are on a given side of a plane compared to the polytopic approximation used for the iiwa.

7 Conclusion

Understanding the complicated geometry of C-free is an essential step to designing safe, collision-free motion plans. In this work, we presented an approach for describing a rational parametrization of C-free, known as TC-free, using a union of polytopes. Our primary contributions are two Sums-of-Squares program (23) and (29) which can certify that a polytopic region of TC-space is collision-free, as well as another program (32) which finds a local improvement that increases the size of a TC-free polytope. We prove that feasibility of our certification programs (23) and (29) are both necessary and sufficient for proving that a polytopic region of TC-space is collision-free and we combine programs (23) and (32) into a practical algorithm for describing TC-free as a union of certified, collision-free polytopes in the TC-space. We deployed our algorithm on both simple and realistic environments and demonstrate

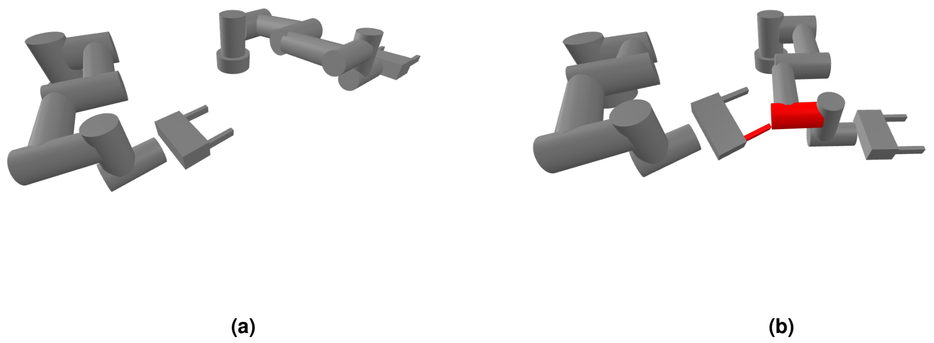


Figure 14. Top down view of two postures sampled within one certified TC-space region on the dual UR3e platform. In the right figure we highlight the two collision geometries that are separated by only 0.3mm. A dynamic visualization of the range of attainable postures is available by running this notebook

that Algorithm 1 finds large TC-space regions which correspond to diverse positions in task space. We demonstrate that these regions are not conservative and very tight to the TC-space obstacle even for 12-DOF systems by showing postures with just millimeters of separation.

The presented method works for TC-spaces of arbitrary dimensions, makes only very mild assumptions on the kinematics of our robot, and makes no assumptions about the shape of the TC-space obstacles. Moreover, it only relies on the mild assumption that obstacles in the task space are described as unions of convex sets, an assumption that is frequently satisfied whenever a given environment is simulated.

Such certified descriptions of TC-free find practical application in both randomized and optimization-based collision-free motion planning algorithms, providing a means to certify safety of an entire trajectory by checking membership in a set rather than by finite sampling which can be prone to false assertions of safety. Moreover, the convexity of the generated regions is particularly attractive to optimization-based methods such as the GCS framework of (Marcucci et al. 2022). Future work intends to further explore these applications as well as practical algorithms for seeding Algorithm 1 to obtain good coverage of TC-free with few regions.

8 Acknowledgement

This work was supported by the MIT Quest For Intelligence.

References

Ahmadi AA, Hall G, Makadia A and Sindhwani V (2016) Geometry of 3d environments and sum of squares polynomials. *arXiv preprint arXiv:1611.07369* .

Ahmadi AA, Krstic M and Parrilo PA (2011) A globally asymptotically stable polynomial vector field with no polynomial lyapunov function. In: *2011 50th IEEE Conference on Decision and Control and European Control Conference*. IEEE, pp. 7579–7580.

Amice A, Dai H, Werner P, Zhang A and Tedrake R (2022) Finding and optimizing certified, collision-free regions in configuration space for robot manipulators. In: *Algorithmic Foundations of Robotics XV: Proceedings of the Fifteenth Workshop on the Algorithmic Foundations of Robotics*. Springer, pp. 328–348.

Andersen ED and Andersen KD (2000) The mosek interior point optimizer for linear programming: an implementation of the homogeneous algorithm. *High performance optimization* : 197–232.

ApS M (2019) *The MOSEK optimization toolbox for MATLAB manual. Version 9.0.* URL http://docs.mosek.com/9.0/toolbox/index.html.

Baldi L and Mourrain B (2021) On moment approximation and the effective putinar's positivstellensatz. *arXiv preprint arXiv:2111.11258* .

Blekherman G, Parrilo PA and Thomas RR (2012) *Semidefinite optimization and convex algebraic geometry*. SIAM.

Boyd S, Boyd SP and Vandenberghe L (2004) *Convex optimization*. Cambridge university press.

Branicky M and Newman W (1990) Rapid computation of configuration space obstacles. In: *Proceedings., IEEE International Conference on Robotics and Automation*. pp. 304–310 vol.1. DOI:10.1109/ROBOT.1990.125992.

Brossette S and Wieber PB (2017) Collision avoidance based on separating planes for feet trajectory generation. In: *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. IEEE, pp. 509–514.

Canny J (1988) *The complexity of robot motion planning*. MIT press.

Craig JJ (2005) *Introduction to robotics: mechanics and control*. Pearson Educacion.

Deits R and Tedrake R (2015a) Computing large convex regions of obstacle-free space through semidefinite programming. In: *Algorithmic foundations of robotics XI*. Springer, pp. 109–124.

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

Dyer ME and Frieze AM (1988) On the complexity of computing the volume of a polyhedron. *SIAM Journal on Computing* 17(5).

Eidenbenz SJ and Widmayer P (2003) An approximation algorithm for minimum convex cover with logarithmic performance guarantee. *SIAM Journal on Computing* 32(3): 654–670.

Ferrier C (2000) Computation of the distance to semi-algebraic sets. *ESAIM: Control, Optimisation and Calculus of Variations* 5: 139–156.

Ghosh M, Amato NM, Lu Y and Lien JM (2013) Fast approximate convex decomposition using relative concavity. *Computer-Aided Design* 45(2): 494–504.

Gill PE, Murray W and Saunders MA (2005) Snopt: An sqp algorithm for large-scale constrained optimization. *SIAM review* 47(1): 99–131.

Han Y, Zhao W, Pan J, Ye Z, Yi R and Liu YJ (2019) A configuration-space decomposition scheme for learning-based collision checking. *arXiv preprint arXiv:1911.08581* .

Jarvis-Wloszek Z, Feeley R, Tan W, Sun K and Packard A (2003) Some controls applications of sum of squares programming. In: *42nd IEEE international conference on decision and control (IEEE Cat. No. 03CH37475)*, volume 5. IEEE, pp. 4676–4681.

Kavraki LE (1995) Computation of configuration-space obstacles using the fast fourier transform. *IEEE Transactions on Robotics and Automation* 11(3): 408–413.

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

Latombe JC (2012) *Robot motion planning*, volume 124. Springer Science & Business Media.

LaValle SM (1998) Rapidly-exploring random trees: A new tool for path planning .

Lien JM and Amato NM (2007) Approximate convex decomposition of polyhedra. In: *Proceedings of the 2007 ACM symposium on Solid and physical modeling*. pp. 121–131.

Lin X, Fernandez GI and Hong DW (2022) Reduce: Reformulation of mixed integer programs using data from unsupervised clusters for learning efficient strategies. In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 4459–4465.

Lingas A (1982) The power of non-rectilinear holes. In: *International Colloquium on Automata, Languages, and Programming*. Springer, pp. 369–383.

Lozano-Perez T (1983) Spatial planning: A configuration space approach. *IEEE Transactions on Computers* 100(32).

Majumdar A and Tedrake R (2017) Funnel libraries for real-time robust feedback motion planning. *The International Journal of Robotics Research* 36(8): 947–982.

Mamou K and Ghorbel F (2009) A simple and efficient approach for 3d mesh approximate convex decomposition. In: *2009 16th IEEE international conference on image processing (ICIP)*. IEEE, pp. 3501–3504.

Marcucci T, Petersen M, von Wrangel D and Tedrake R (2022) Motion planning around obstacles with convex optimization. *arXiv preprint arXiv:2205.04422* .

Marcucci T, Umenberger J, Parrilo PA and Tedrake R (2021) Shortest paths in graphs of convex sets. *arXiv preprint arXiv:2101.11565* .

Marshall M (2008) *Positive polynomials and sums of squares*. 146. American Mathematical Soc.

Nie J (2011) Polynomial matrix inequality and semidefinite representation. *Mathematics of Operations Research* 36(3): 398–415.

Nie J and Schweighofer M (2007) On the complexity of Putinar's Positivstellensatz. *Journal of Complexity* 23(1): 135–150. DOI:10.1016/j.jco.2006.07.002.

Parrilo PA (2000) *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. California Institute of Technology.

Parrilo PA (2004) Sum of squares programs and polynomial inequalities. In: *SIAG/OPT Views-and-News: A Forum for the SIAM Activity Group on Optimization*, volume 15. pp. 7–15.

Provan JS and Ball MO (1983) The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM Journal on Computing* 12(4): 777–788.

Putinar M (1993) Positive polynomials on compact semi-algebraic sets. *Indiana University Mathematics Journal* 42(3): 969–984.

Rudin W (1976) *Principles of mathematical analysis*, volume 3. McGraw-hill New York.

Schouwenaars T, De Moor B, Feron E and How J (2001) Mixed integer programming for multi-vehicle path planning. In: *2001 European control conference (ECC)*. IEEE, pp. 2603–2608.

Schwarzer F, Saha M and Latombe JC (2004) Exact collision checking of robot paths. In: *Algorithmic foundations of robotics V*. Springer, pp. 25–41.

Shen S and Tedrake R (2020) Sampling quotient-ring sum-of-squares programs for scalable verification of nonlinear systems. In: *2020 59th IEEE Conference on Decision and Control (CDC)*. IEEE, pp. 2535–2542.

Spivak M (1994) *Calculus Third Edition*. Cambridge University Press.

Stengle G (1996) Complexity estimates for the schmüdgen positivstellensatz. *Journal of Complexity* 12(2): 167–174.

Sturmfels B (1994) On the newton polytope of the resultant. *Journal of Algebraic Combinatorics* 3(2): 207–236.

Tedrake R (2021) *Robotic Manipulation*. URL https://manipulation.mit.edu/pick.html.

Tedrake R (2022) *Underactuated Robotics*. URL https://underactuated.csail.mit.edu.

Tedrake R, Manchester IR, Tobenkin M and Roberts JW (2010) Lqr-trees: Feedback motion planning via sums-of-squares verification. *The International Journal of Robotics Research* 29(8): 1038–1052.

Tedrake R and the Drake Development Team (2019) Drake: Model-based design and verification for robotics. URL https://drake.mit.edu.

Trutman P, Mohab SED, Henrion D and Pajdla T (2020) Globally optimal solution to inverse kinematics of 7dof serial manipulator. *arXiv preprint arXiv:2007.12550* .

Verghese M, Das N, Zhi Y and Yip M (2022) Configuration space decomposition for scalable proxy collision checking in robot planning and control. *arXiv preprint arXiv:2201.04314* .

Wampler CW and Sommese AJ (2011) Numerical algebraic geometry and algebraic kinematics. *Acta Numerica* 20.

Wong TH, Leach G and Zambetta F (2014) An adaptive octree grid for gpu-based collision detection of deformable objects. *The Visual Computer* 30(6): 729–738.

Yin H, Arcak M, Packard A and Seiler P (2021) Backward reachability for polynomial systems on a finite horizon. *IEEE Transactions on Automatic Control* 66(12): 6025–6032.

A Algebraic Kinematics

An in depth review of algebraic kinematics and low order pairs can be found in (Wampler and Sommese 2011, Chapter 4). We include a brief review in this appendix for completeness.

A mechanism composed of $N+1$ links is considered algebraic if each link is connected by one of the following five joints:

- Revolute (R): a 1-DOF joint permitting revolution about an axis of symmetry. An example is a door handle.
- Prismatic (P): a 1-DOF joint permitting translation along an axis. An example is a linear rail.
- Cylindrical (C): a 2-DOF joint permitting both revolution about an axis of symmetry and independent translation along a given axis. An example is the rods of a Foosball table.
- Planar (E): A 3-DOF joint permitting translation and rotation in a two-dimensional plane. An example is hockey puck moving on the surface of the ice.
- Spherical (S): A 3-DOF joint permitting free rotation between two links. An example is the human shoulder.

We recall from Section 3.3 that the pose of a point A expressed in the reference frame F, written as a function of the robot configuration q can be expressed as

$$\begin{bmatrix} {}^FR^A(q) & {}^Fp^A(q) \\ 0_{1\times 3} & 1 \end{bmatrix} = \prod_{i\in\mathcal{I}_{F,A}} {}^{P_i}X^{C_i}(q_i)\ {}^{C_i}X^{P_{i+1}} \tag{33}$$

where ${}^{P_i}X^{C_i}(q_i)$ is a rigid transform describing the relative motion allowed by the i^{th} joint. The matrices ${}^{P_i}X^{C_i}(q_i)$ are in general restriction of the following forms

$${}^{P_i}X^{C_i}(q_i) = \begin{cases} \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) & 0 & x_i \\ \sin(\theta_i) & \cos(\theta_i) & 0 & y_i \\ 0 & 0 & 1 & z_i \\ 0 & 0 & 0 & 1 \end{bmatrix} & \text{if } i^{\text{th}} \text{ joint is one of R, P, C, or E} \\ \begin{bmatrix} U(\psi_i) & 0_{3\times 1} \\ 0_{1\times 3} & 1 \end{bmatrix} & \text{if } i^{\text{th}} \text{ joint is S} \end{cases} \tag{34}$$

The specific restrictions for R, P, C, and E joints are given in Table 3. The matrix U is an element of $SO(3)$ parametrized using Euler angles $\{\phi_{i,x}, \phi_{i,y}, \phi_{i,z}\}$.

Joint	Restriction	Definition of q_i
R	$x_i = y_i = z_i = 0$	$q_i = \{\theta_i\}$
P	$\theta_i = x_i = y_i = 0$	$q_i = \{z_i\}$
C	$x_i = y_i = 0$	$q_i = \{\theta_i, z_i\}$
E	$z_i = 0$	$q_i = \{\theta_i, x_i, y_i\}$
S	see equation (37)	$q_i = \{\phi_{i,x}, \phi_{i,y}, \phi_{i,z}\}$

Table 3. parameterization of algebraic joints in terms of the matrix given in (34).

We remark that the joints C, E, and S can be constructed by the composition of R and P joints.

- A C joint is a composition of an R joint and a P joint:

$$\begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) & 0 & 0 \\ \sin(\theta_i) & \cos(\theta_i) & 0 & 0 \\ 0 & 0 & 1 & z_i \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) & 0 & 0 \\ \sin(\theta_i) & \cos(\theta_i) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & z_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{35}$$

- An E joint is the composition of one R joint and two P joints

$$\begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) & 0 & x_i \\ \sin(\theta_i) & \cos(\theta_i) & 0 & y_i \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & x_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & y_i \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) & 0 & 0 \\ \sin(\theta_i) & \cos(\theta_i) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{36}$$

- An S joint is the composition of three R joints expressed as Euler angles.

$$U(\psi_i) = \begin{bmatrix} \cos(\psi_{i,x}) & -\sin(\psi_{i,x}) & 0 \\ \sin(\psi_{i,x}) & \cos(\psi_{i,x}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\psi_{i,y}) & 0 & -\sin(\psi_{i,y}) \\ 0 & 1 & 0 \\ \sin(\psi_{i,y}) & 0 & \cos(\psi_{i,y}) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\psi_{i,z}) & -\sin(\psi_{i,z}) \\ 0 & \sin(\psi_{i,z}) & \cos(\psi_{i,z}) \end{bmatrix} \tag{37}$$

Our approach presented for a robot composed of R and P joints can be extended to handle any algebraic mechanism by consider the other algebraic joints as compositions of R and P joints.

B Definition of Archimedean

In this section we formally define the Archimedean property that appears in Theorem 1 and Theorem 2.

Definition 1. *A semialgebraic set $\mathcal{S}_g = \{x \mid g_i(x) \geq 0, i \in [n]\}$ is Archimedean if there exists $N \in \mathbb{N}$ and $\lambda_i(x) \in \boldsymbol{\Sigma}$ such that:*

$$N - \sum_{i=1}^{n} x_i^2 = \lambda_0(x) + \sum_{i=1}^{n} \lambda_i(x) g_i(x)$$

C Semialgebraic Descriptions of Set Membership for Common Convex Bodies

Body	Variables	Description of $\mathcal{A}(s)$ as a semi-algebraic set
V-rep Polytope with m vertices v_i at position ${}^Fp^{v_i}(s) = \frac{{}^Ff^{v_i}(s)}{{}^Fg^{v_i}(s)}$	$\{s, x, \mu\}$	$h_1(s,x,\mu) = \left(\prod_i {}^Fg^{v_i}\right)\left(x - \sum_{i=1}^m \mu_i \left(\frac{{}^Ff^{v_i}(s)}{{}^Fg^{v_i}(s)}\right)\right)$ $h_2(\mu) = 1 - \sum_{i=1} \mu_i$ $\gamma_i(\mu_i) = \mu_i,\ i \in [m]$
Sphere with center o at position ${}^Fp^o(s) = \frac{{}^Ff^o(s)}{{}^Fg^o(s)}$ and radius r	$\{s, x\}$	$\gamma_1(s,x) = \left({}^Fg^o(s)\right)^2 \left(r^2 - \left\lVert x - \frac{{}^Ff^o(s)}{{}^Fg^o(s)}\right\rVert^2\right)$
Capsule, the convex hull of two spheres with centers c_1 and c_2 at positions ${}^Fp^{o_i}(s) = \frac{{}^Ff^{o_i}(s)}{{}^Fg^{o_i}(s)}$ and radii r_1, r_2	$\{s, x, \mu\}$	$\frac{{}^Ff^{o_\mu}}{{}^Fg^{o_\mu}} = \mu\frac{{}^Ff^{o_1}(s)}{{}^Fg^{o_1}(s)} + (1-\mu)\frac{{}^Ff^{o_2}(s)}{{}^Fg^{o_2}(s)}$ $r_\mu = \mu r_1 + (1-\mu)r_2$ $\gamma_1(s,x,\mu) = \left({}^Fg^{o_\mu}(s)\right)^2 \left(r_\mu^2 - \left\lVert x - \frac{{}^Ff^{o_\mu}}{{}^Fg^{o_\mu}}\right\rVert^2\right)$ $\gamma_2(\mu) = \mu$ $\gamma_3(\mu) = 1 - \mu$
Cylinder, the convex hull of two circles with centers o_1 and o_2, at position ${}^Fp^{o_i}(s) = \frac{{}^Ff^{o_i}(s)}{{}^Fg^{o_i}(s)}$, lying in the plane normal to ${}^Fp^{o_1}(s) - {}^Fp^{o_2}(s)$, and with radii r_1 and r_2.	$\{s, x, v, \mu\}$	$\frac{{}^Ff^{o_\mu}(s)}{{}^Fg^{o_\mu}(s)} = \mu\frac{{}^Ff^{o_1}(s)}{{}^Fg^{o_1}(s)} + (1-\mu)\frac{{}^Ff^{o_2}(s)}{{}^Fg^{o_2}(s)}$ $r_\mu = \mu r_1 + (1-\mu)r_2$ $h_1(v,s) = v^T\left(\frac{{}^Ff^{o_1}(s)}{{}^Fg^{o_1}(s)} - \frac{{}^Ff^{o_2}(s)}{{}^Fg^{o_2}(s)}\right)$ $h_2(s,x,\mu,v) = x - \frac{{}^Ff^{o_\mu}(s)}{{}^Fg^{o_\mu}(s)} - v$ $\gamma_1(v,\mu) = r_\mu^2 - v^Tv$ $\gamma_2(\mu) = \mu$ $\gamma_3(\mu) = 1 - \mu$

Table 4. Parameterizations of the condition that x lies in a convex body that moves rigidly as a function of s.

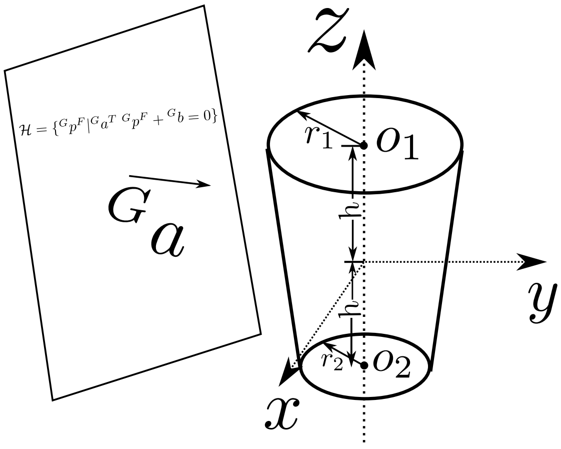

$\mathcal{H} = \{{}^G p^F | {}^G a^T \, {}^G p^F + {}^G b = 0\}$
${}^G a$
z
y
x
r_1
o_1
r_2
o_2
h
h

Figure 15. Illustration of the cylinder on one side of the plane $\mathcal{H}$, with the plane normal being ${}^G a$, expressed in the cylinders geometry frame G.

D Parametrized Hyperplane Separation Condition for the Cylinder

To derive the hyperplane separation condition for cylinder, we first attach a geometric frame G to the cylinder, as shown in Fig.15. The cylinder's geometric frame G's origin coincides with the cylinder's center, with the z axis of the G frame along the cylinder axis. The height of the cylinder is $2h$, with the top/bottom circle radius being r_1 and r_2 respectively.

We first write the plane $\mathcal{H}$ with its parameters ${}^G a(s), {}^G b(s)$ in the cylinder's geometric frame G and derive the conditions on ${}^G a(s), {}^G b(s)$. The cylinder is in the positive side of the plane if and only if both its top and bottom rim are on the positive side of the plane, namely

$$\left({}^G a(s)\right)^T \begin{bmatrix} r_1 \cos\alpha \\ r_1 \sin\alpha \\ h \end{bmatrix} + {}^G b(s) \geq 0 \ \forall \alpha \tag{38a}$$

$$\left({}^G a(s)\right)^T \begin{bmatrix} r_2 \cos\alpha \\ r_2 \sin\alpha \\ -h \end{bmatrix} + {}^G b(s) \geq 0 \ \forall \alpha. \tag{38b}$$

Taking the infimum of both sides with respect to α makes the above conditions equivalent to

$${}^G a_z(s) h + {}^G b(s) \geq r_1 \left\| \begin{bmatrix} {}^G a_x(s) & {}^G a_y(s) \end{bmatrix} \right\| \tag{39a}$$

$$-{}^G a_z(s) h + {}^G b(s) \geq r_2 \left\| \begin{bmatrix} {}^G a_x(s) & {}^G a_y(s) \end{bmatrix} \right\| \tag{39b}$$

Next, we use the Schur complement, to reformulate (39a) and (39b) the positive semidefinite matrix conditions. For example, (39a) is equivalent to

$$\begin{bmatrix} {}^Ga_z(s)h + {}^Gb(s) & 0 & r_1\, {}^Ga_x(s) \\ 0 & {}^Ga_z(s)h + {}^Gb(s) & r_1\, {}^Ga_y(s) \\ r_1\, {}^Ga_x(s) & r_1\, {}^Ga_y(s) & {}^Ga_z(s)h + {}^Gb(s) \end{bmatrix} \succeq 0 \tag{40a}$$

As explained in Section 4.1, this polynomial PSD condition can be reformulated as the condition

$$u^T \begin{bmatrix} {}^Ga_z(s)h + {}^Gb(s) & 0 & r_1\, {}^Ga_x(s) \\ 0 & {}^Ga_z(s)h + {}^Gb(s) & r_1\, {}^Ga_y(s) \\ r_1\, {}^Ga_x(s) & r_1\, {}^Ga_y(s) & {}^Ga_z(s)h + {}^Gb(s) \end{bmatrix} u \geq 0\; \forall u. \tag{40b}$$

To avoid the trivial solution ${}^Ga(s) = 0, {}^Gb(s) = 0$ (which is not in fact a separating plane), we add the extra constraint ${}^Ga^T \left(\frac{{}^Gp^{o_1} + {}^Gp^{o_2}}{2}\right) + {}^Gb \geq 1$. Here $\frac{{}^Gp^{o_1} + {}^Gp^{o_2}}{2}$ is the position of the cylinder center expressed in the geometric frame G which coincides with the frame's origin. Therefore $\frac{{}^Gp^{o_1} + {}^Gp^{o_2}}{2} = 0$, and so it is sufficient to introduce the constraint

$${}^Gb(s) \geq 1 \tag{41}$$

to exclude the trivial solution ${}^Ga = 0, {}^Gb = 0$.

In our optimization program, we express the separating plane in a frame F (where the choice of frame F is discussed in F.1), not in cylinder's geometric frame G. Hence we need to compute ${}^Ga(s), {}^Gb(s)$ from their corresponding terms ${}^Fa(s), {}^Fb(s)$ expressed in frame F and the relative transform ${}^FX^G$ between the two frames

$${}^Ga(s) = {}^GR^F(s)\; {}^Fa(s) \tag{42a}$$

$${}^Gb(s) = {}^Fb(s) + \left({}^Fa(s)\right)^T\; {}^Fp^G(s) \tag{42b}$$

As described in Section 3.3, both the position ${}^Fp^G(s)$ and orientation ${}^GR^F(s)$ are rational functions of s. By replacing ${}^Ga(s), {}^Gb(s)$ in (40b) and (41) with (42) and requiring the resulting numerator of the rational function to be non-negative, we derive that the plane separating cylinders can be enforced via a polynomial non-negativity condition which can be formulated as sums-of-squares condition.

E The Certification Programs are Necessary and Sufficient

In this section, we expand our discussion on the power of the certification programs presented in Sections 4.1 and 4.2. As remarked previously, Theorems 4 and 3 are necessary as programs (29) and (23) are infinite dimensional. It is not immediately obvious that for every robot and every scene, there exists a finite degree where in each program must become feasible when $\mathcal{P}$ truly contains no collisions.

A second subtlety applies specifically to (23). When generalizing (3), we argued that it was beneficial to search for a parametric hyperplane as a function of our TC-space variable s and asserted that a polynomial parameterization was a good choice. However, it is not obvious that a polynomial parameterization is sufficient, and perhaps we require a rational or even more complicated parameterization of the plane.

These questions about the power of SOS programming arise in other domains. For example, SOS is commonly used to search for polynomial Lyapunov functions to prove the stability of polynomial dynamical systems (Majumdar and Tedrake 2017). However, it is known that not every stable polynomial dynamical system admits a polynomial Lyapunov function (Ahmadi et al. 2011), and therefore SOS programming is a sufficient, but not necessary tool for proving the stability of dynamical systems.

Fortunately, our certification programs from 4.1 and 4.2 are indeed *necessary and sufficient*, in the sense that there will always exist a finite degree such that the programs become feasible if $\mathcal{P}$ contains no collision. The proof of this for the program (29) follows immediately from Theorem 2.

Proof. (of Theorem 4) Our assumptions on $\mathcal{P}$, $\mathcal{A}$, and $\mathcal{B}$ imply that $\mathcal{S}_{\mathcal{P},\mathcal{A},\mathcal{B}}$ is an Archimedean set. Therefore, the feasibility of (29) for sufficiently high degree ρ follows immediately from "effective" versions of Theorem 2 such as those given in (Nie and Schweighofer 2007; Baldi and Mourrain 2021) which give explicit degree bounds. □

Though the proof of Theorem 3 is more technically involved, the key idea is simple. In short, we construct a family of continuous functions which map each TC-space configuration $s \in \mathcal{P}$ to a separating plane. We then argue that this family of continuous functions must contain hyperplanes which are parametrized as polynomials. Finally, we again appeal to "effective" versions of Theorem 1 such as those given in (Nie and Schweighofer 2007; Baldi and Mourrain 2021) to show that these polynomials can be found using SOS programming.

We proceed in steps, first establishing that the set of separating planes at a point s in TC-free is open.

Proposition 1. *Let $\Phi(s)$ denote the set of strictly separating hyperplanes at the point s for bodies $\mathcal{A}(s)$ and $\mathcal{B}(s)$ and let $\mathcal{P}$ be a non-empty, polytopic subset of TC-free. Then $s \in \mathcal{P}$ implies that $\Phi(s)$ is a non-empty, open set.* □

Proof. By definition, a hyperplane $\begin{bmatrix} a \\ b \end{bmatrix} \in \mathbb{R}^4$ strictly separates $\mathcal{A}(s)$ and $\mathcal{B}(s)$ if and only if there exist positive constants $\varepsilon_{\mathcal{A}}$ and $\varepsilon_{\mathcal{B}}$ such that $a^T x + b \geq \varepsilon_{\mathcal{A}}\ \forall x \in \mathcal{A}(s)$and $a^T x + b \leq -\varepsilon_{\mathcal{B}}\ \forall x \in \mathcal{B}(s)$. Since the bodies $\mathcal{A}(s)$ and $\mathcal{B}(s)$ are strictly separating for every point $s \in \mathcal{P}$, the Separating Hyperplane theorem guarantees the existence of such a vector and so $\Phi(s)$ is non-empty.

Now consider $\begin{bmatrix} a \\ b \end{bmatrix} \in \Phi(s) \subseteq \mathbb{R}^4$ and its δ neighborhood

$$\mathcal{N}(\delta) = \left\{ \begin{bmatrix} a \\ b \end{bmatrix} + \delta \begin{bmatrix} v_a \\ v_b \end{bmatrix} \middle| \left\| \begin{bmatrix} v_a \\ v_b \end{bmatrix} \right\| \leq 1 \right\},$$

with $\delta > 0$.

We have that for all $x \in \mathcal{A}$

$$(a^T + \delta v_a^T)x + (b + \delta v_b) \geq \varepsilon_{\mathcal{A}} + \delta \min_{\|v\| \leq 1,\ x \in \mathcal{A}} v_a^T x + v_b$$

and similarly for all $x \in \mathcal{B}$

$$(a^T + \delta v_a^T)x + (b + \delta v_b) \leq -\varepsilon_{\mathcal{B}} + \delta \max_{\|v\| \leq 1,\ x \in \mathcal{B}} v_a^T x + v_b.$$

Letting $M_l = \min_{\|v\|=1,\ x \in \mathcal{A}} v_a^T x + v_b$ and $M_u = \max_{\|v\|=1,\ x \in \mathcal{B}} v_a^T x + v_b$, we have that all planes $\mathcal{N}(\delta)$ are separating if

$$0 < \delta < \min\left\{\frac{\varepsilon_{\mathcal{A}}}{|M_l|}, \frac{\varepsilon_{\mathcal{B}}}{|M_u|}\right\}$$

and so $\Phi(s)$ is open. □

Proposition 2. *Define*

$$\mathcal{N}(s, \delta) = \bigcap_{\|v\| \leq 1} \Phi(s + \delta v)$$

For all $s \in \mathcal{P}$ there exists $\delta_{\min}(s) > 0$, not necessarily finite such that, $\mathcal{N}(s, \delta)$ is non-empty and open for every $0 < \delta < \delta_{\min}$.

Proof. Recall that the position of every point in $\mathcal{A}(s)$ is a continuous function of s and that the distance from a point to a set is a continuous function (Rudin 1976). Therefore, the distance of every point in $\mathcal{A}(s)$ to every element of $\Phi(s)$ changes continuously. For every $\delta > 0$ and $\begin{bmatrix} a \\ b \end{bmatrix} \in \Phi(s)$ we define:

$$M_\delta(s) \coloneqq \sup_{\|v\| \leq 1} \left| \inf_{x \in \mathcal{A}(s)} a^T x + b - \inf_{x \in \mathcal{A}(s + \delta v)} a^T x + b \right|$$

We have that

$$\inf_{\|v\| \leq 1} \inf_{x \in \mathcal{A}(s + \delta v)} a^T x + b \geq \inf_{x \in \mathcal{A}(s)} a^T x + b - M_\delta(s) \geq \varepsilon_{\mathcal{A}} - M_\delta(s)$$

Moreover, if $\delta_2 < \delta_1$, then $M_{\delta_2}(s) \leq M_{\delta_1}(s)$. By continuity and monotonicity, $M_\delta(s) \to 0$ as $\delta \to 0$ and so there exists δ sufficiently small such that $\varepsilon_{\mathcal{A}} - M_\delta(s) > 0$. A similar argument shows that δ can be chosen sufficiently small such that the plane $\begin{bmatrix} a \\ b \end{bmatrix}$ continues to satisfy the separating plane conditions for $\mathcal{B}$. Therefore $\begin{bmatrix} a \\ b \end{bmatrix} \in \Phi(s + \delta v)$ for all v such that $\|v\| \leq 1$ if δ is chosen sufficiently small. It is clear that choosing δ smaller continues to ensure that $\mathcal{N}(s, \delta)$ is non-empty. Openness is immediate following a similar argument to Proposition 1. □

The above proposition enables us to establish that there exists an open family of continuous functions $f(s)$ such that their outputs are always separating hyperplanes.

Proposition 3. *Let $\mathcal{F}$ be the set of continuous functions mapping*

$$f : s \mapsto \begin{bmatrix} a \\ b \end{bmatrix}$$

such that $f(s) \in \Phi(s)$ for all $s \in \mathcal{P}$. The set $\mathcal{F}$ is non-empty and open under the pointwise metric

$$d(f, g) = \sup_{s \in \mathcal{P}} \|f(s) - g(s)\|.$$

Proof. Suppose $\mathcal{F}$ were empty. Then every function satisfying $f(s) \in \Phi(s)\ \forall s \in \mathcal{P}$ is not a continuous function. Namely, for every f there exists a point s_0 such that for all $\delta > 0$, $f(s_0) \in \Phi(s_0)$ but $f(s_0) \notin \mathcal{N}(s_0, \delta)$. This contradicts the openness of $\mathcal{N}(s_0, \delta)$ for a sufficiently small δ from Proposition 2 and so $\mathcal{F}$ is non-empty. Openness follows from the fact that if $\delta > 0$ is chosen sufficiently small, then for every continuous g satisfying $d(f, g) < \delta$, then g must also separate $\mathcal{A}(s)$ and $\mathcal{B}(s)$ for every $s \in \mathcal{P}$. □

We are now ready to prove Theorem 3

Proof. (of Theorem 3) By Proposition 3, $\mathcal{F}$ is a non-empty open subset of continuous functions defined on the compact domain $\mathcal{P}$. The Stone-Weierstrass theorem (Rudin 1976) states that the set of polynomial functions on a compact domain is dense in the set of continuous functions in that domain under the pointwise metric. Therefore, $\mathcal{F}$ must contain a map $p : s \mapsto \begin{bmatrix} a(s) \\ b(s) \end{bmatrix}$ such that each component is a polynomial. This polynomial is of finite degree and is a strictly separating hyperplane and therefore by "effective" versions of Theorem 1 such as (Nie and Schweighofer 2007; Baldi and Mourrain 2021), there exists a Putinar certificates of finite degree certifying that $p(s)$ is a separating hyperplane. □

F Practical aspects

In this section, we discuss some practical aspects for essential for enabling Algorithm 1 to realistic examples. These include the choice of reference frame in which to express the forward kinematics, the selection of a finite basis for the polynomials in our SOS programs, and which aspects of 1 can be parallelized.

F.1 Choosing the Reference Frame

The polynomial implications upon which the certification program (23) and polytope growth program (32) are based require choosing a coordinate frame between each collision pair $\mathcal{A}$ and $\mathcal{B}$. However, as the collision-free certificate between two different collision pairs can be computed independently of each other, we are free to choose a different coordinate frame to express the kinematics for each collision pair. This is important in light of (10) and (14) that indicate that the degree of the polynomials ${}^F f^{\mathcal{A}_j}$ and ${}^F g^{\mathcal{A}_j}$ are equal to two times the number of joints lying on the kinematic chain between frame F and the frame for $\mathcal{A}$. For example, the tangent-configuration-space polynomial in the variable s describing the position of the end-effector of a 7-DOF robot is of total degree 14 when written in the coordinate frame of the robot base. However, when written in the frame of the third link, the polynomial describing the position of the end effector is only of total degree $(7-3)\times 2 = 8$. This observation is also used in (Trutman et al. 2020) to reduce the size of the optimization program.

The size of the semidefinite variables in (23) and (32) scale as the square of the degree of the polynomial used to express the forward kinematics. Supposing there are n links in the kinematics chain between $\mathcal{A}$ and $\mathcal{B}$, then choosing the jth link along the kinematics chain as the reference frame F leads to scaling of order $j^2 + (n-j)^2$. Choosing the reference frame in the middle of the chain minimizes this complexity to scaling of order $\frac{n^2}{2}$ and we therefore adopt this convention in our experiments.

F.2 Basis Selection

The condition that a polynomial can be written as a sum of squares can be equivalently formulated as an equality constraint between the coefficients of the polynomial and an associated semidefinite variable known as the Gram matrix (Parrilo 2004). Namely, a polynomial $p(s)$ is sums-of-squares if and only if $p(s) = z(s)^T X z(s), X \succeq 0$ where $z(s)$ is a vector of monomials and X is the Gram matrix. The number of rows in the positive semidefinite Gram matrix equals to the size of the vector $z(s)$. In general, a sums-of-squares polynomial in k variables of total degree $2d$ requires a Gram matrix of size $\binom{k+d}{d}$ to represent which can quickly become prohibitively large. Fortunately, the polynomials in our programs contain substantially more structure which will allow us to select a small-sized vector of monomials $z(s)$, and hence drastically reduce the size of the Gram matrices and speed up the optimization problem.

F.2.1 Polytopic collision geometry We begin with the separating plane condition for polytopic collision geometries. Note that from (14) that while both the numerator and denominator of the forward kinematics are of total degree $2n$, with n the number of links of the kinematics chain between frame A and F, both polynomials are of *coordinate* degree of at most two (i.e. the highest degree of s_i in any term is s_i^2). We will refer to this basis as $\nu(s)$ which is a vector containing terms of the form $\prod_{i=1}^n s_i^{\text{degree}(s_i)}$ with $\text{degree}(s_i) \in \{0,1,2\}$ for all 3^n possible permutations of the exponents $\text{degree}(s_i)$.

We recall that we parametrize our hyperplane using polynomial entries. If $a_{\mathcal{A},\mathcal{B}}(s) = a_{\mathcal{A},\mathcal{B}}^T\eta(s)$, $b_{\mathcal{A},\mathcal{B}}(s) = b_{\mathcal{A},\mathcal{B}}^T\eta(s)$ for some basis η in the variable s. The position of $x(s) \in \mathcal{A}(s)$ is expressed in basis $\nu(s)$, then the left hand side of (19) can be expressed as a linear function of the basis $\gamma(s)$, where $\gamma(s)$ contains all the possible entries that appear in the outer product $\eta(s)\nu(s)^T$.

Example 3. *Suppose*

$$\eta(s) = \begin{bmatrix}1 & s_1 & s_2\end{bmatrix}^T$$

and

$$\nu(s) = \begin{bmatrix}1 & s_1 & s_1^2 & s_2 & s_2^2 & s_1s_2 & s_1^2s_2 & s_1s_2^2 & s_1^2s_2^2\end{bmatrix}^T$$

.

Then:

$$\gamma(s) = \begin{bmatrix}1 & s_1 & s_1^2 & s_1^3 & s_2 & s_2^2 & s_2^3 & s_1s_2 & s_1^2s_2 & s_1^3s_2 & s_1s_2^2 & s_1^2s_2^2 & s_1^3s_2^2 & s_1s_2^3 & s_1^2s_2^3\end{bmatrix}$$

Namely $\gamma(s)$ contains the monomials whose degree for each s_i is at most 3, and only one of s_i can have degree 3 (hence $s_1^3s_2^3$ is not included in $\gamma(s)$).

Similarly, we must select a basis $\rho(s)$ for our multiplier polynomials $\lambda_{ij}^{\mathcal{A},\mathcal{B}}(s)$. The equality in (19) determines the minimum necessary basis $\rho(s)$. If the polynomial $p(s)$ is expressed in basis $\gamma(s)$, then the minimal such basis is related to an object known in computational algebra as the Newton polytope of γ denoted $\mathbf{New}(\gamma(s))$ (Sturmfels 1994). Denoting the linear basis

$$l(s) = \begin{bmatrix} 1 & s_1 & s_2 & \dots & s_N \end{bmatrix},$$

then exact condition is that

$$\mathbf{New}(\gamma(s)) = \mathbf{New}(\eta(s)) + \mathbf{New}(\nu(s)) \subseteq \mathbf{New}(\rho(s)) + \mathbf{New}(l(s))$$

where the sum in this case is the Minkowski sum.

By using affine polynomials for separating plane parameters $a_{\mathcal{A},\mathcal{B}}(s), b_{\mathcal{A},\mathcal{B}}(s)$, we know that $\eta(s)$ is the same as the linear basis $l(s)$, then we obtain the condition that $\mathbf{New}(\rho(s)) = \mathbf{New}(\nu(s))$ and since $\nu(s)$ is a dense, even degree basis we must take $\rho(s) = \nu(s)$. A sums-of-squares polynomial in the basis of $\nu(s)$ has Gram matrix with 2^n rows. Choosing $\eta(s)$ as the constant basis would in fact result in the same condition, and therefore searching for separating planes which are linear functions of the tangent-configuration-space variable does not increase the size of the semidefinite variables. As the complexity of (23) and (32) are dominated by the size of these semidefinite variables, separating planes which are linear functions changes do not substantially affect the solve time but can dramatically increase the size of the regions which we can certify.

Because of this, we choose to parametrize all of our hyperplanes throughout our experiments as linear functions of the TC-space variables. We stress that in general, the choice of a linearly parametrized hyperplane, and the selection of $\rho(s)$ to be the minimum size to match the degree of the left hand side of (19) may not be sufficient to prove that a region $\mathcal{P}$ is collision-free, even if $\mathcal{P}$ truly is collision-free. Indeed due of many complexity-theoretic results, we expect that in general $\eta(s)$ and $\rho(s)$ may need to have exponentially high degree for some robots, scenes, and polytopes $\mathcal{P}$ (Stengle 1996). However, in practice we have observed that the choices in this section are sufficient to certify many regions of interest, while keeping the optimization problem size tractable for state-of-art numerical solvers.

Remark 5. *Attempting to certifying that the end-effector of a 7-DOF robot will not collide with the base using program (23) using linearly parametrized hyperplanes and choosing to express conditions (19) in the world frame with naïvely chosen bases would result in semidefinite variables of size* $\binom{7+7}{7} = 3432$. *Choosing to express the same conditions according to the discussion in Section F.1 and choosing the basis $\gamma(s)$ described in this section results in semidefinite matrices of rows at most* $2^{\lceil 7/2 \rceil} = 2^4 = 16$. *The division by 2 comes from choosing the middle link as the expressed frame, hence halving the kinematic chain length.*

F.2.2 Non-polytopic collision geometry In this section, we use the sphere as a running example for explaining how we choose the monomial bases for certifying separation of the non-polytopic geometries; the monomial bases for capsules and cylinders can be derived in a similar manner.

As mentioned in (20), we need to impose

$$s \in \mathcal{P} \implies \begin{bmatrix} \left((a(s))^T\, {}^F f^o(s) + b(s)\, {}^F g^o(s)\right) I_3 & r a(s)\, {}^F g^o(s) \\ r (a(s))^T\, {}^F g^o(s) & (a(s))^T\, {}^F f^o(s) + b(s)\, {}^F g^o(s) \end{bmatrix} \succeq 0. \quad (43)$$

By the definition of positive semidefinite matrix [§§], we know that the 4×4 matrix in the right of $\Longrightarrow$ in (43) is positive semidefinite if and only if

$$\forall \bar{u} \in \mathbb{R}^3, \underbrace{\begin{bmatrix} \bar{u} \\ 1 \end{bmatrix}^T \begin{bmatrix} \left((a(s))^T\,{}^F f^o(s) + b(s)\,{}^F g^o(s)\right) I_3 & r a(s)\,{}^F g^o(s) \\ r(a(s))^T\,{}^F g^o(s) & (a(s))^T\,{}^F f^o(s) + b(s)\,{}^F g^o(s) \end{bmatrix} \begin{bmatrix} \bar{u} \\ 1 \end{bmatrix}}_{\sigma(\bar{u},s)} \geq 0. \tag{44}$$

We impose the following sufficient condition for (43), where $\mathcal{P} = \{s | c_j^T(s) \leq d_j, j = 1, \ldots, m\}$

$$\sigma(\bar{u}, s) = \lambda_0(\bar{u}, s) + \sum_{j=1}^{m} \lambda_j(\bar{u}, s)(d_j - c_j^T s) \tag{45a}$$

$$\text{for } j = 0, \ldots, m, \lambda_j(\bar{u}, s) \geq 0\ \forall \bar{u}, s. \tag{45b}$$

Now we analyze the degree of the polynomial $\sigma(\bar{u}, s)$ defined in (44). As mentioned in the previous subsection, each monomial in ${}^F f^o(s), {}^F g^o(s)$ are of the form $\prod_{i=1}^n s_i^{\text{degree}(s_i)}$, $\text{degree}(s_i) \in \{0, 1, 2\}$. Combining this with the choice of a separating plane $a(s), b(s)$ being affine functions of s, we derive that each monomial in $\sigma(\bar{u}, s)$ is of the form $\bar{u}_j^{\text{degree}(\bar{u}_j)} \prod_{i=1}^n s_i^{\text{degree}(s_i)}$, where $\text{degree}(\bar{u}_j) \in \{0, 1, 2\}$, $\text{degree}(s_i) \in \{0, 1, 2, 3\}$, and at most one of $\text{degree}(s_i)$ can be 3. As an example, $\bar{u}_1^2 s_1^3 s_2 s_3^2$ is a valid monomial in $\sigma(\bar{u}, s)$ but $\bar{u}_1 \bar{u}_2$ is not (because $\sigma(\bar{u}, s)$ doesn't contain the cross product between $\bar{u}_j, \bar{u}_k, j \neq k$). Similarly, $s_1^3 s_2^3$ is not in the basis because at most one of s_i can have degree 3. Given these properties on the monomials in $\sigma(\bar{u}, s)$- specifically there being no cross-product term $\bar{u}_j \bar{u}_k, j \neq k$ in $\sigma(\bar{u}, s)$- we can write the positive polynomials $\lambda_j(\bar{u}, s)$ as the summation of three SOS polynomials

$$\lambda_j(\bar{u}, s) = \sum_{k=1}^{3} \lambda_{j,k}(\bar{u}_k, s) \tag{46a}$$

$$\lambda_{j,k}(\bar{u}_k, s) \in \mathbf{\Sigma}. \tag{46b}$$

For each monomial in the SOS polynomial $\lambda_{j,k}(\bar{u}_k, s)$, the degree of $\bar{u}_k$ and s_i for $i = 1, \ldots, n$ is either 0, 1, or 2. Hence the number of rows in the Gram matrix in $\lambda_{j,k}(\bar{u}_k, s)$ is of size 2^{n+1}. By choosing the reference frame according to the convention from Appendix F.1, n is no larger than $\lceil N/2 \rceil$ where N is the number of joints in the robot.

Remark 6. *For a 6-DOF UR3erobot whose collision geometries are approximated by cylinders, to certify the collision-avoidance between the robot and objects in the world (or self-collision), the largest positive semidefinite matrix in our optimization problem has rows at most* $2^{\lceil 6/2 \rceil + 1} = 2^4 = 16$, *where the division by 2 comes from choosing the middle link as the expressed link, hence halving the kinematic chain length to* $\lceil 6/2 \rceil$.

[§§] A matrix X is positive semidefinite if and only if $\forall \bar{u}, \begin{bmatrix} \bar{u} \\ 1 \end{bmatrix}^T X \begin{bmatrix} \bar{u} \\ 1 \end{bmatrix} \geq 0$

F.3 Parallelization

While it is attractive from a theoretical standpoint to write (23) as a single, large program it is worth noting that it can in fact be viewed as K individual SOS programs, where K is the number of collision pairs in the environment. Indeed, certifying whether pairs $(\mathcal{A}_1, \mathcal{A}_2)$ are collision-free for all s in the polytope $\mathcal{P}$ can be done completely independently of the certification of another pair $(\mathcal{A}_1, \mathcal{A}_3)$ as the constraint are not coupled between any pairs. Similarly, the search for the largest inscribed ellipsoid can be done independently of the search for the separating hyperplanes.

Solving the certification program (23) as K individual SOS programs has several advantages. First, as written (23) has $2(m+1)K\sum_i |\mathcal{A}_i|$ semidefinite variables of various sizes, where m is the number of inequalities in $\mathcal{P}$ and $|\mathcal{A}_i|$ denotes the number of inequalities required to express that body $\mathcal{A}_i$ is on a particular side of the plane (see Table 4). In the example from Section 6.1.2 this corresponds to $18,720$ semidefinite variables. This can be prohibitively large to store in memory as a single program as the size of these semidefinite variables grow. Solving for the separating plane for each pair of collision bodies independently also enables us to determine which collision bodies cannot be certified as collision-free and allows us to terminate our search as soon as a single pair cannot be certified. Finally, decomposing the problems into subproblems enables us to increase computation speed by leveraging parallel processing.

The program (30) can also be solved completely independently of the certification program (23) and is in general a much smaller SDP than any individual certification program. Therefore, lines 3 and 4 of Algorithm 1 can be solved in parallel.

We note that (32) cannot be similarly decomposed as on this step the variables c_i^T and d_i affect all of the constraints. However, this program is substantially smaller as we have fixed $2mK\sum_i |\mathcal{A}_i|$ of the semidefinite variables as constants and replaced them with $2m$ linear variables representing the polytope. This program is much more amenable to being solved as a single program.

G Seeding Algorithm 1

Algorithm 1 must be initialized with a polytope $\mathcal{P}_0$ for which (23) is feasible. In principle, the alternation proposed in Section 5 can be seeded with an arbitrarily small polytope around a collision-free seed point. This seed polytope is then allowed to grow using Algorithm 1. However, this may require running several dozens of iterations of Algorithm 1 for each seed point which can become prohibitive as the number of degrees of freedom in our robot or the complexity of the scene grows. It is therefore advantageous to seed with as large a region as can be initially certified.

Here we discuss an extension of the IRIS algorithm in (Deits and Tedrake 2015a) which uses nonlinear optimization to rapidly generate large regions in TC-space. These regions are not guaranteed to be collision-free and therefore they must still be passed to Algorithm 1 to be certified, but do provide good initial guesses. In this section, we will assume that the reader is familiar with IRIS and will only discuss the modification required to use it to grow TC-space regions. Detailed pseudocode is available in Appendix H.

IRIS grows regions in a given space by alternating between two subproblems: SEPARATINGHYPERPLANES and INSCRIBEDELLIPSOID. The INSCRIBEDELLIPSOID is exactly the program described in (Boyd et al. 2004, Section 8.4.2) and we do not need to modify it. The subproblem SEPARATINGHYPERPLANES finds a set of hyperplanes which separate the ellipse generated by INSCRIBEDELLIPSOID from all of the obstacles. This subproblem is solved by calling two subroutines: CLOSESTPOINTONOBSTACLE and TANGENTPLANE. The former finds the closest point on a given obstacle to the ellipse, while the latter places a plane at the point found in CLOSESTPOINTONOBSTACLE that is tangent to the ellipsoid.

The original work of (Deits and Tedrake 2015a) assumes convex obstacles which enables CLOSESTPOINTONOBSTACLE to be solved as a convex program and for the output of TANGENTPLANE to be globally separating plane between the obstacle and the ellipsoid of the previous step. Due to the non-convexity of the TC-space obstacles in our problem formulation, finding the closest point on an obstacle exactly becomes a computationally difficult problem to solve exactly (Ferrier 2000). Additionally, placing a tangent plane at the nearest point will be only a locally separating plane, not a globally separating one.

To address the former difficulty, we formulate CLOSESTPOINTONOBSTACLE as a nonlinear program. Let the current ellipse be given as $\mathcal{E} = \{Qs + s_0 \mid \|s\|_2 \leq 1\}$ and suppose we have the constraint that $s \in \mathcal{P} = \{s \mid Cs \leq d\}$. Let $\mathcal{A}$ and $\mathcal{B}$ be two collision pairs and ${}^{\mathcal{A}}p_{\mathcal{A}}, {}^{\mathcal{B}}p_{\mathcal{B}}$ be some point in bodies $\mathcal{A}$ and $\mathcal{B}$ expressed in some frame attached to $\mathcal{A}$ and $\mathcal{B}$. Also, let ${}^{W}X^{\mathcal{A}}(s)$ and ${}^{W}X^{\mathcal{B}}(s)$ denote the rigid transforms from the reference frames $\mathcal{A}$ and $\mathcal{B}$ to the world frame respectively. We remind the reader that this notation is drawn from (Tedrake 2021). The closest point on the obstacle subject to being contained in $\mathcal{P}$ can be found by solving the program

$$\min_{s, {}^{\mathcal{A}}p_{\mathcal{A}}, {}^{\mathcal{B}}p_{\mathcal{B}}} (s - s_0)^T Q^T Q (s - s_0) \; \textbf{subject to} \tag{47a}$$

$${}^{W}X^{\mathcal{A}}(s) {}^{\mathcal{A}}p_{\mathcal{A}} = {}^{W}X^{\mathcal{B}}(s) {}^{\mathcal{B}}p_{\mathcal{B}} \tag{47b}$$

$$Cs \leq d \tag{47c}$$

This program searches for the nearest configuration in the metric of the ellipse such that two points in the collision pair come into contact. We find a locally optimal solution $(s^\star, {}^{\mathcal{A}}p^\star_{\mathcal{A}}, {}^{\mathcal{B}}p^\star_{\mathcal{B}})$ to the program using a fast, general-purpose nonlinear solver such as SNOPT (Gill et al. 2005). The tangent plane to the ellipse $\mathcal{E}$ at the point $s^\star$ is computed by calling TANGENTPLANE, then appended to the inequalities of $\mathcal{P}$ to form $\mathcal{P}'$. This routine is looped until (47) is infeasible at which point INSCRIBEDELLIPSE is called again.

Once a region $\mathcal{P} = \{s \mid Cs \leq d\}$ is found by Algorithm 2, it will typically contain some minor violations of the non-collision constraint. To find an initial, feasible polytope $\mathcal{P}_0$ to use in Algorithm 1, we search for a minimal uniform contraction δ of $\mathcal{P}$ such that $\mathcal{P}_\delta = \{s \mid Cs \leq d - \delta * 1\}$ is collision-free. This can be found by bisecting over the variable $\delta \in [0, \delta_{\max}]$ and solving repeated instances of (23).

Seeding Algorithm 1 with a $\mathcal{P}_0$ as above can dramatically reduce the number of alternations required to obtain a fairly large region and is frequently faster than seeding Algorithm 1 with an arbitrarily small polytope.

H Supplementary Algorithms

We present a pseudocode for the algorithm presented in Appendix G. A mature implementation of this algorithm can be found in Drake[¶¶].

[¶¶]https://github.com/RobotLocomotion/drake/blob/2f75971b66ca59dc2c1dee4acd78952474936a79/geometry/optimization/iris.cc#L440

Algorithm 2: Given an initial tangent-configuration-space point s_0 and a list of obstacles $\mathcal{O}$, return a polytopic region $\mathcal{P} = \{s \mid Cs \leq d\}$ and inscribed ellipsoid $\mathcal{E}_\mathcal{P} = \{s \mid Qs + s_0\}$ which contains a substantial portion of the free TC-space (but is not guaranteed to contain no collisions).

```
1  $(C, d) \leftarrow$ robot joint limits
2  $\mathcal{P}_0 \leftarrow \{s \mid Cs \leq d\}$
3  $\mathcal{E}_{\mathcal{P}_0} \leftarrow \textsc{InscribedEllipsoid}(\mathcal{P}_0)$
4  $j \leftarrow$ number of rows of $C$
5  do
6      do
7          $(s^\star, {}^\mathcal{A}p^\star_\mathcal{A}, {}^\mathcal{B}p^\star_\mathcal{B}) \leftarrow \textsc{FindClosestCollision}(\mathcal{P}_i, \mathcal{E}_{\mathcal{P}_i})$
8          $(c^T_{j+1}, d_{j+1}) \leftarrow \textsc{TangentHyperplane}(s^\star, \mathcal{E}_{\mathcal{P}_i})$
9          $C \leftarrow \textbf{vstack}(C, c^T_{j+1})$
10         $d \leftarrow \textbf{vstack}(d, d_{j+1})$
11         $\mathcal{P}_i \leftarrow \{s \mid Cs \leq d\}$
12         $j \leftarrow j + 1$
13     while $\textsc{FindClosestCollision}(\mathcal{P}_i, \mathcal{E}_{\mathcal{P}_i})$ is feasible;
14     $\mathcal{E}_{\mathcal{P}_i} \leftarrow \textsc{InscribedEllipsoid}(\mathcal{P}_i)$
15     $i \leftarrow i + 1$
16 while $(\textbf{vol}(\mathcal{E}_i) - \textbf{vol}(\mathcal{E}_{i-1}))\,/\textbf{vol}(\mathcal{E}_{i-1}) \geq$ tolerance;
17 return $(\mathcal{P}_i, \mathcal{E}_{\mathcal{P}_i})$
```