Planning along Differentiable Charts of Constraint Manifolds with the Inverse Function Theorem

Thomas Cohn, Seiji Shaw, Nicholas Roy, and Russ Tedrake

***Abstract*— When planning motions with kinematic equality constraints, analytic inverse kinematics can be used to parameterize the set of feasible configurations. We present a new approach for computing gradients through such parameterizations (a requirement for trajectory optimization), that extends compatibility beyond handwritten analytic IK solutions to general-purpose implementations, including automated symbolic solvers like IKFast.**

I. Introduction

For a variety of tasks, a robot forms a closed linkage, and must execute constrained motions. For example, a bimanual robot carrying a box must maintain a constant transformation between its end-effectors. In such cases, the set of *kinematically-valid* joint configurations is a measure-zero subset of the full configuration space of the robot, presenting a fundamental challenge for motion planning [1–3].

One conceptual approach for planning with kinematic equality constraints is to build an intrinsic representation of the so-called *constraint manifold*. Given an *atlas*, a collection of local coordinate systems on the manifold (called *charts*), the planner must reason about which charts to move through and the paths within those charts [4, p. 411]. This eliminates the equality constraints, enabling the use of ordinary collision-free motion planning algorithms. The primary obstacle to such intrinsic approaches is the difficulty in describing the charts [5, p. 168; 6, p. 151; 7, p. 28; 8, p. 60]. Today, such approaches have been limited to certain special manifolds [9–12].

However, many of the constraint manifolds relevant to robotics arise from end-effector (EE) constraints. A recent line of work has examined using *analytic inverse kinematics* (IK) as a practical tool for constructing charts. Analytic IK serves a principled solution to the one-to-many property of IK, taking in additional arguments to return a unique solution for an EE target. Together with automatic differentiation through the IK mapping, a trajectory optimizer can represent the robot's path in the parameterized space while imposing costs and constraints in configuration space. This approach has been effective for collision-free constrained planning [13, 14] and as a tool for solving complex IK problems like collision-free IK and humanoid stability [15]. Some of the more commonly-used robot arms have specific analytic IK solutions [16–21], but otherwise, the simplest way forward is to use automated tools that solve the kinematic equations and generate a solution [22–24].

This project was supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. 2141064, and MIT Siegel Family Quest for Intelligence. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation or the other sponsors acknowledged in this work.

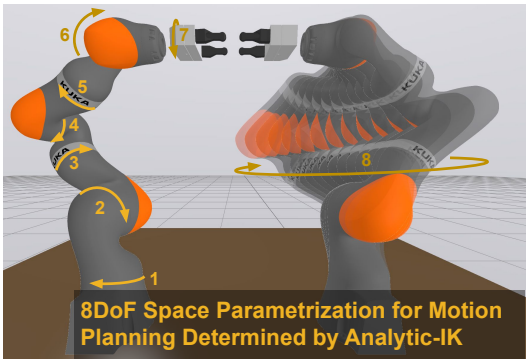

Fig. 1: We construct a minimal coordinates parameterization for a bimanual carrying task. Figure reproduced from [25].

While these solutions provide everything needed for our framework (e.g. returning all solutions, explicit self-motion parameterization), modifying the resulting implementation for compatibility with autodiff (for gradient-based trajectory optimization) is impractical because they are optimized for speed rather than interpretability. [1] We present a novel approach for differentiating through a black box IK function, towards trajectory optimization on constraint manifolds. We construct the augmented forward kinematics [26, 27], a variant of ordinary FK that also returns the self-motion parameters corresponding to an analytic IK formulation. The augmented FK Jacobian has a simple expression and can be used to calculate the IK Jacobian via the inverse function theorem. We verify the efficacy of this approach with direct numerical experiments and downstream motion planning tasks.

II. Background

We use monogram notation [28, §3.1] to describe poses: ${}^{A}X^{B} \in \mathrm{SE}(3)$ is the pose of frame B relative to (and expressed in) frame A. When this pose is a function of a variable q, we write ${}^{A}X^{B}(q)$. The forward kinematics are $\mathrm{FK} : \mathbb{R}^n \to \mathrm{SE}(3)$. An analytic IK function is a mapping

$$\mathrm{IK} : \mathcal{U} \times \Psi \times \mathscr{K} \to \mathbb{R}^n, \tag{1}$$

where $\mathcal{U} \subseteq \mathrm{SE}(3)$ is a set of reachable end-effector positions, Ψ parameterizes the set of continuous self-motions and $\mathscr{K}$ enumerates the discrete self-motions (e.g. "up-elbow" vs "down-elbow"). In practice, we restrict ourselves to a specific $\kappa \in \mathscr{K}$, and drop the corresponding discrete self-motion

[1] The IKFast [22] solution for the UR3 arm is over 110 000 lines of code.

from the notation. We may extend the domain of IK to all of $\mathrm{SE}(3)$ via a projection onto the reachable workspace, but then $\mathrm{FK}(\mathrm{IK}(X, \psi, \kappa)) = X$ only for $X \in \mathcal{U}$ [13]. Analytic IK allows us to chart the constraint manifold resulting from certain closed kinematic linkages: if removing analytically-solvable subchains from the linkage graph results in a forest, we naturally have a parameterization [29]. Automatic differentiation through this mapping enables gradient-based optimization with variables in the minimal coordinates, but costs and constraints in configuration space.

Robot manipulators often have limited workspaces, so optimization problems in the parameterized coordinates must impose a *reachability constraint*. Within analytic IK functions, the domain limitations appear as functions like $\arccos$. By modifying the function to clip to the domain, the function will now have no domain limits, but will return an incorrect result outside of the reachable set. One way to enforce reachability is with *direct reachability* constraints, which require the achieved end-effector transform matches the desired, but this constraint is active for all feasible configurations [13]. If we have detailed knowledge of the internals of the analytic IK solutions, a better formulation of reachability uses *probing functions* that return intermediate values before domain-limited functions are applied [15].

For a constraint manifold defined implicitly as $\mathcal{M} = \{q \in \mathbb{R}^n : F(q) = 0\}$, a minimum-cost path (with respect to a cost functional L) between $q_0, q_1 \in \mathcal{M}$ can be found via

$$\begin{aligned} \operatorname{argmin} \quad & L(\gamma) \\ \text{s.t.} \quad & \gamma \in \mathcal{C}^1([0,1], \mathbb{R}^n), \\ & \gamma(t) \in \mathcal{M}, && \forall t \in [0,1], \\ & \gamma(t) \text{ is collision free}, && \forall t \in [0,1], \\ & \gamma(0) = q_0, \gamma(1) = q_1. \end{aligned}$$

Given a parameterization $\phi : \mathcal{U} \to \mathcal{M}$ with $\mathcal{U} \subseteq \mathbb{R}^m$, $m \leq n$, and points $\tilde{q}_0, \tilde{q}_1$ such that $\phi(\tilde{q}_0) = q_0$ and $\phi(\tilde{q}_1) = q_1$, this problem can be rewritten as

$$\begin{aligned} \operatorname{argmin} \quad & L(\phi \circ \tilde{\gamma}) \\ \text{s.t.} \quad & \tilde{\gamma} \in \mathcal{C}^1([0,1], \mathcal{U}), \\ & (\phi \circ \tilde{\gamma})(t) \text{ is collision free}, && \forall t \in [0,1], \\ & \tilde{\gamma}(0) = \tilde{q}_0, \tilde{\gamma}(1) = \tilde{q}_1. \end{aligned}$$

For many constrained trajectory optimization problems, the elimination of the equality constraint $\gamma(t) \in \mathcal{M}$ via an IK parameterization results in an easier-to-solve problem [13, 15].

III. Methodology

Suppose we are given an analytic IK implementation that exposes the discrete and redundant self-motion but does not support automatic differentiation. This would prevent the use of gradient-based optimization, as we need the Jacobian of this mapping (or some other way of computing Jacobian-vector products) to obtain the gradient of the costs and constraints. To resolve this issue, we can leverage the inverse function theorem [30, Thm. 2.11] to compute the Jacobian of IK in terms of the derivative of FK, which can easily be computed with standard robotics toolboxes [31, 32].

Theorem 1 (Inverse Function Theorem)**.** *If a function $f : \mathbb{R}^n \to \mathbb{R}^n$ is continuously differentiable at a point p and has a full-rank Jacobian, then there is an inverse function f^{-1} defined on a neighborhood of $f(p)$, and $Df(p)^{-1} = D(f^{-1})f(p)$.*

For a nonredundant manipulator, there is no continuous self-motion, so the inverse of IK (restricted to a specific $\kappa \in \mathscr{K}$) is precisely FK. For a redundant manipulator, we call the inverse of the analytic IK function the *augmented* forward kinematics (inspired by the language of Elias and Wen [27]). We write $\mathrm{FK}_A : \mathbb{R}^n \to \mathrm{SE}(3) \times \Psi \times \mathscr{K}$ and restrict to a single discrete self-motion $\kappa \in \mathscr{K}$; its Jacobian will have the block structure

$$D\,\mathrm{FK}_A(q) = \left(D\,\mathrm{FK}(q), D\,\mathrm{FK}_\psi(q)\right)^T, \tag{2}$$

where FK_ψ maps the joint angles q to their corresponding self-motion parameter. If certain joint angles are used to parameterize the self motion (as done in IKFast [22]), the rows of $D\,\mathrm{FK}_\psi(q)$ are standard basis vectors. For a 7DoF arm using the shoulder-elbow-wrist parameterization, it is still simple to compute this Jacobian [27].

Now, we can describe how to leverage the Jacobian of the forward-kinematics mapping $D\,\mathrm{FK}_A(q)$ to compute gradients of the analytic IK mapping. Suppose we have decision variables (X, ψ), partials $(\frac{\partial X}{\partial y}, \frac{\partial \psi}{\partial y})$ for some earlier decision variable y. We obtain $q = \mathrm{IK}(X, \psi)$, and want to compute $\frac{\partial q}{\partial y} = D\,\mathrm{IK}(X, \psi)(\frac{\partial X}{\partial y}, \frac{\partial \psi}{\partial y})^T$. Due to the IFT, $D\,\mathrm{FK}_A(q) = D\,\mathrm{IK}(X, \psi)^{-1}$, so we solve the linear system

$$D\,\mathrm{FK}_A(q)\frac{\partial q}{\partial y} = \left(\frac{\partial X}{\partial y}, \frac{\partial \psi}{\partial y}\right)^T. \tag{3}$$

This is faster and more numerically-stable than inverting $D\,\mathrm{FK}_A(q)$ directly. For non-reachable configurations (and representational singularities [27]), $D\,\mathrm{FK}_A$ loses rank, so we compute a least-squares solution instead.

Example. We now discuss how to apply the IFT machinery to the constrained bimanual setup in Fig. 1. Let $q_c, q_s \in \mathbb{R}^7$ be the joint angles of the controlled and subordinate arms, E_c and E_s their end-effector frames, and let $\psi_s \in \mathbb{R}$ be the subordinate arm's continuous self-motion parameter. Let FK_c be the forward kinematics of the controlled arm and FK_A refer to the augmented forward kinematics of the subordinate arm. We compute the desired pose of the subordinate arm's end-effector in terms of q_c via ${}^W X^{E_s}(q_c) = {}^W X^{E_c}(q_c) {}^{E_c} X^{E_s}$, where ${}^{E_c} X^{E_s}$ is the fixed transform between the end-effectors. Then we compute $q_s = \mathrm{IK}({}^W X^{E_s}(q_c), \psi_s)$, obtaining the full configuration $(q_c, q_s) = \phi(q_c, \psi_s)$. Thus, ϕ parameterizes the constraint manifold, and we wish to compute the derivatives $\frac{\partial (q_c, q_s)}{\partial y} = D\phi(q_c, \psi_s)(\frac{\partial q_c}{\partial y}, \frac{\partial \psi}{\partial y})^T$.

The first step is to examine the Jacobian of ϕ, which tends to have well-organized block structure

$$D\phi = \frac{\partial (q_c, q_s)}{\partial (q_c, \psi_s)} = \begin{bmatrix} \frac{\partial q_c}{\partial q_c} & \frac{\partial q_c}{\partial \psi_s} \\ \frac{\partial q_s}{\partial (q_c, \psi_s)} & \end{bmatrix} \tag{4}$$

$$= \left[\begin{array}{c|c} I_{7\times7} & 0_{7\times1} \\ \hline \frac{\partial q_s}{\partial({}^{W}X^{E_s},\psi_s)} & \frac{\partial({}^{W}X^{E_s},\psi_s)}{\partial(q_c,\psi_s)} \end{array} \right], \tag{5}$$

$$= \left[\begin{array}{c|c} I_{7\times7} & 0_{7\times1} \\ \hline (D\,\mathrm{FK}_A(q_s))^{-1} \begin{bmatrix} {}^{E_c}X^{E_s} D\,\mathrm{FK}_c(q_c) & 0_{6\times1} \\ 0_{1\times7} & 1_{1\times1} \end{bmatrix} \end{array} \right], \tag{6}$$

which is expressed in known or easily-computable quantities.

The second step is to compute the Jacobian-vector product $\frac{\partial(q_c,q_s)}{\partial y} = D\phi \frac{\partial(q_c,\psi_s)}{\partial y}$. While we could find the bottom block of matrix $D\phi$ directly by explicitly inverting $D\,\mathrm{FK}_A(q_c)$, some symbolic manipulation reveals a more efficient strategy to uncover the derivatives $\frac{\partial q_s}{\partial y}$. Instead, it is sufficient to solve the linear system

$$D\,\mathrm{FK}_A(q_s)\frac{\partial q_s}{\partial y} = \begin{bmatrix} {}^{E_c}X^{E_s} D\,\mathrm{FK}_c(q_c) & 0_{6\times1} \\ 0_{1\times7} & 1_{1\times1} \end{bmatrix} \begin{bmatrix} \frac{\partial q_c}{\partial y} \\ \frac{\partial \psi_s}{\partial y} \end{bmatrix}. \tag{7}$$

IV. Experiments

We evaluate the derivatives in isolation with low-level calculations, and examine high level performance on downstream tasks. $\tilde{q}$ denotes a point in the parameterized space, and q denotes a point in the full configuration space. All experiment code is optimized C++, built on Drake [31].

First, we examine speed and accuracy of the derivative calculations. We sample random reachable configurations $\{\tilde{q}_i\}_{i=1}^{10\,000}$ (with rejection sampling). For each configuration, we sample random vectors of partial derivatives $\frac{\partial \tilde{q}_i}{\partial y_{ij}} \in \mathbb{R}^{2^j}$. For each $(\tilde{q}_i, \frac{\partial \tilde{q}_i}{\partial y_{ij}})$, we compute $\frac{\partial q_i}{\partial y_{ij}}$ via forward-mode autodiff and our IFT technique. The mean error for each partial size is less than 10^{-13}, and the 95th percentile is less than 10^{-12}. Autodiff is fastest up to partial size 2^7, but for larger partial sizes, IFT is faster, due to constant cost of building the augmented Jacobian $D\,\mathrm{FK}_A(q)$. Full numerical results are shown in Fig. 2.

We also run downstream tasks that leverage these gradients. For all of these experiments, we compare four settings:

- Autodiff with direct reachability constraints, as in [13],
- Autodiff with probing functions, as in [15],
- IFT with zero gradients when non-reachable, and
- IFT with least squares gradients when non-reachable.

(Both IFT gradients use direct reachability constraints, as the probing functions are unavailable.)

First, we generate convex feasible subsets of the parameterized configuration space using IrisNp2, as presented in [33]. These regions can be used to solve motion planning problems with graphs of convex sets [34, 35]. Next, we solve trajectory optimization problems [28, §6.2], using bidirectional RRT [36] plus shortcutting [37] to obtain the initial guess. Finally, we retime trajectories from all three planners to satisfy dynamics limits with TOPPRA [38].

Runtimes are presented in Table I. For trajopt, the switch from probing functions to direct reachability is entirely responsible for the runtime increase. TOPPRA does not use reachability constraints, so the runtime increase closely matches the low-level runtime differences. IrisNp2 sees much higher runtime with IFT, since it frequently has to compute gradients for non-reachable configurations, where these gradients are inaccurate. Using least-squares gradients performs somewhat better than zero gradients, suggesting this still provides some information to the optimizer even at non-reachable configurations.

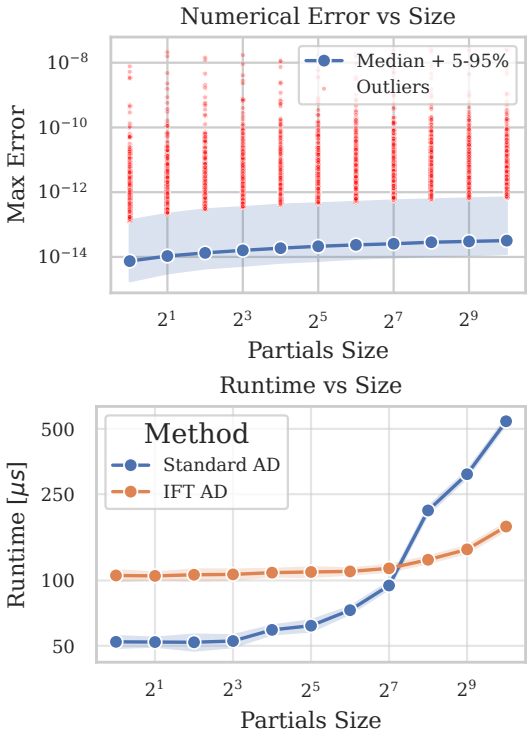

Fig. 2: Error (top) and runtime (bottom) of the IFT gradients, in comparison to autodiff, tested on 10 000 randomly-sampled reachable configurations. Axes are log-scale.

TABLE I: Downstream runtimes by gradient strategy.

Gradient Method	IrisNp2	Trajopt	TOPPRA
Autodiff, probing functions	25.99	0.29	2.73
Autodiff, direct reachability	25.01	1.89	2.71
IFT, zero gradients	99.99	1.91	4.60
IFT, pseudoinverse	90.94	1.90	4.78

V. Discussion

We have presented a new strategy for differentiating through IK mappings, based on the inverse function theorem. This avoids the potentially-prohibitive requirement to modify analytic IK functions to support autodiff. Rigorous empirical results demonstrate the accuracy and speed of the method, and downstream optimization experiments confirm that the performance decrease is a reasonable tradeoff for the greater generality as long as a feasible initial guess is given.

Future work will examine how we can handle IK formulations that simply return no solution for non-reachable configurations, instead of the approximate solutions we rely on here. However, recent general approaches, such as IK-Geo [39], are designed to return least-squares solutions, potentially allowing them to be used as-is with our method.

References

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

[2] R. Bonalli, A. Cauligi, A. Bylard, T. Lew, and M. Pavone, "Trajectory optimization on manifolds: A theoretically-guaranteed embedded sequential convex programming approach," in *Proceedings of Robotics: Science and Systems*, FreiburgimBreisgau, Germany, June 2019.

[3] R. Bordalba, T. Schoels, L. Ros, J. M. Porta, and M. Diehl, "Direct collocation methods for trajectory optimization in constrained robotic systems," *IEEE Transactions on Robotics*, vol. 39, no. 1, pp. 183–202, 2022.

[4] J.-C. Latombe, *Robot motion planning*. Springer Science & Business Media, 2012, vol. 124.

[5] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

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

[7] K. M. Lynch and F. C. Park, "Modern robotics: Mechanics, planning, and control," 2017.

[8] H. Choset, K. M. Lynch, S. Hutchinson, G. A. Kantor, and W. Burgard, *Principles of robot motion: theory, algorithms, and implementations*. MIT press, 2005.

[9] M. Watterson, S. Liu, K. Sun, T. Smith, and V. Kumar, "Trajectory optimization on manifolds with applications to quadrotor systems," *The International Journal of Robotics Research*, vol. 39, no. 2-3, pp. 303–320, 2020.

[10] T. Cohn, M. Petersen, M. Simchowitz, and R. Tedrake, "Non-euclidean motion planning with graphs of geodesically convex sets," *The International Journal of Robotics Research*, vol. 44, no. 10-11, pp. 1840–1862, 2025.

[11] S. Teng, A. Jasour, R. Vasudevan, and M. G. Jadidi, "Convex Geometric Motion Planning on Lie Groups via Moment Relaxation," in *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023.

[12] S. Teng, T.-Y. Lin, W. A. Clark, R. Vasudevan, and M. Ghaffari, "Riemannian Direct Trajectory Optimization of Rigid Bodies on Matrix Lie Groups," in *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025.

[13] T. Cohn, S. Shaw, M. Simchowitz, and R. Tedrake, "Constrained bimanual planning with analytic inverse kinematics," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 6935–6942.

[14] T. Cohn, P. Werner, and R. Tedrake, "Faster algorithms for growing collision-free regions of constrained bimanual configuration spaces," in *IROS 2025 Workshop on Frontiers in Dynamic, Intelligent, and Adaptive Multi-Arm Manipulation*, 2025.

[15] T. Cohn, L. Tang, A. Amice, and R. Tedrake, "A framework for combining optimization-based and analytic inverse kinematics," *arXiv preprint arXiv:2602.05092*, 2026.

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

[17] S. Tittel, "Analytical solution for the inverse kinematics problem of the Franka Emika Panda seven-dof lightweight robot arm," in *2021 20th International Conference on Advanced Robotics (ICAR)*. IEEE, 2021, pp. 1042–1047.

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

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

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

[21] N. Ramezani and M.-A. Williams, "Smooth robot motion with an optimal redundancy resolution for PR2 robot based on an analytic inverse kinematic solution," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2015, pp. 338–345.

[22] R. Diankov, "Automated construction of robotic manipulation programs," Ph.D. dissertation, Carnegie Mellon University, USA, 2010.

[23] D. Zhang and B. Hannaford, "Ikbt: Solving symbolic inverse kinematics with behavior tree," *Journal of Artificial Intelligence Research*, vol. 65, pp. 457–486, 2019.

[24] W. Gao, "yaik: Yet another analytical inverse kinematics generator," https://github.com/weigao95/yaik, 2024.

[25] T. B. Cohn, "Motion planning along manifolds with geodesic convexity and analytic inverse kinematics," Master's thesis, Massachusetts Institute of Technology, 2024.

[26] J. Baillieul, "Kinematic programming alternatives for redundant manipulators," in *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, vol. 2. IEEE, 1985, pp. 722–728.

[27] A. J. Elias and J. T. Wen, "Redundancy parameterization and inverse kinematics of 7-dof revolute manipulators," *Mechanism and Machine Theory*, vol. 204, p. 105824, 2024.

[28] R. Tedrake, *Robotic Manipulation*, 2024. [Online]. Available: http://manipulation.mit.edu

[29] L. Han and N. Amato, "A kinematics-based probabilistic roadmap method for closed chain systems," *Algo-*

rithmic and Computational Robotics: New Directions 2000 WAFR, p. 233, 2001.

[30] M. Spivak, *Calculus on manifolds: a modern approach to classical theorems of advanced calculus*. CRC press, 2018.

[31] R. Tedrake and the Drake Development Team, "Drake: Model-based design and verification for robotics," 2019. [Online]. Available: https://drake.mit.edu

[32] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 5026–5033.

[33] P. Werner, T. Cohn, R. H. Jiang, T. Seyde, M. Simchowitz, R. Tedrake, and D. Rus, "Faster algorithms for growing collision-free convex polytopes in robot configuration space," *The International Journal of Robotics Research*, 2026, accepted/to appear.

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

[35] T. Marcucci, M. Petersen, D. Von Wrangel, and R. Tedrake, "Motion planning around obstacles with convex optimization," *Science robotics*, vol. 8, no. 84, p. eadf7843, 2023.

[36] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proceedings 2000 ICRA. Millennium conference. IEEE international conference on robotics and automation. Symposia proceedings (Cat. No. 00CH37065)*, vol. 2. IEEE, 2000, pp. 995–1001.

[37] S. Sekhavat, P. Svestka, J.-P. Laumond, and M. H. Overmars, "Multilevel path planning for nonholonomic robots using semiholonomic subsystems," *The international journal of robotics research*, vol. 17, no. 8, pp. 840–857, 1998.

[38] H. Pham and Q.-C. Pham, "A new approach to time-optimal path parameterization based on reachability analysis," *IEEE Transactions on Robotics*, vol. 34, no. 3, pp. 645–659, 2018.

[39] A. J. Elias and J. T. Wen, "Ik-geo: Unified robot inverse kinematics using subproblem decomposition," *Mechanism and Machine Theory*, vol. 209, p. 105971, 2025.