Biconvex Optimization for Smooth Minimum-Time Trajectories around Convex Obstacles

Peter Werner[1], Tobia Marcucci[2], and Daniela Rus[1]

***Abstract*—We present an anytime approach for minimum-time motion planning around convex obstacles that computes high-quality trajectories, is reliable, and supports derivative constraints to arbitrary order. We convexify the minimum-time objective and all derivative constraints through a change of variables, and handle collision avoidance via time-varying separating planes, reducing the problem to a biconvex program. Our anytime algorithm alternates between computing maximum-margin separating planes and optimizing the trajectory. By only adding planes for obstacles that the current iterate collides with, the trajectory can jump around obstacles and escape some local minima. The method requires a collision-free path as initialization, e.g., a sequence of collision-free waypoints, and is then guaranteed to converge to a solution. On a dual-arm pallet unloading benchmark, our approach produces trajectories with comparable quality and computation times as a state-of-the-art decomposition-based planner, while being more general and substantially more robust to bad initialization.**

I. Introduction

Despite decades of research on motion planning, practitioners still often rely on hard-coded trajectories in high-stakes applications. To this day, existing planners require users to choose between reliability or trajectory quality. Sampling-based planners [1], [2] are reliable but produce jagged and circuitous paths that require heuristic post-processing [3], [4]. Trajectory optimization methods [5]–[7] can produce high-quality trajectories but struggle with nonconvex collision-avoidance constraints, leading to unreliable convergence.

Decomposition-based motion planners (DBMPs) [8]–[17] build on ideas from both sampling-based planning and trajectory optimization, attempting to combine the strengths of both. They decompose the free space into convex sets and restrict the trajectory to be contained in their union. However, key limitations remain: jointly optimizing trajectory duration and enforcing derivative constraints beyond second order (e.g., jerk or snap bounds) leads to nonconvex stitching constraints between segments. Existing DBMPs either fix the duration [8], [13]–[15], resort to sequential schemes [10], [12], or simply omit these constraints [16]–[18]. Additionally, DBMPs require a convex decomposition of the free space, which is computationally costly and must be recomputed when the environment changes [11], [19].

We propose a biconvex approach that allows us to plan around convex obstacles and addresses these limitations. We convexify the minimum-time objective and derivative constraints to an arbitrary degree using a change of variables inspired by the control-systems literature. Collision avoidance is enforced through time-varying separating planes, similar in spirit to the linearized constraints in [5]. These planes allow trajectories to curve around obstacles without explicit multi-segment decompositions. Additionally, the convexity of the obstacles enables us to explicitly search for maximum-margin separating planes that admit rapid convergence. Our anytime algorithm can be initialized from any collision-free path, including simple polygonal curves from a sampling-based planner, and then alternates between computing separating planes and updating the trajectory.

[1]MIT CSAIL [2]UC Santa Barbara

II. Problem Statement

We seek a minimum-time trajectory $q : [0, T] \to \mathbb{R}^n$ connecting q_0 to q_T, avoiding convex obstacles $\mathcal{O}_1, \dots, \mathcal{O}_K \subset \mathbb{R}^n$, with each derivative $q^{(i)}$ contained in a compact convex set $\mathcal{C}_i$, which contains the origin, for $i = 1, \dots, I$:

$$\begin{aligned}
\text{minimize} \quad & T && \text{(1a)}\\
\text{subject to} \quad & q(0) = q_0,\ q(T) = q_T, && \text{(1b)}\\
& q^{(i)}(0) = q^{(i)}(T) = 0, \quad i = 1, \dots, I, && \text{(1c)}\\
& q^{(i)}(t) \in \mathcal{C}_i, \quad i = 1, \dots, I,\ t \in [0, T], && \text{(1d)}\\
& q(t) \notin \mathcal{O}_k, \quad k = 1, \dots, K,\ t \in [0, T]. && \text{(1e)}
\end{aligned}$$

The variables are the trajectory q and the duration T. The existence and continuity of the derivatives are implicit constraints.

III. Convex-Analysis Background

First, we review a few useful convex sets in order to set up the minimum-time optimization problem. We use the notation $\lambda\mathcal{S} = \{\lambda x : x \in \mathcal{S}\}$ to denote the product of a scalar $\lambda \in \mathbb{R}$ and a set $\mathcal{S} \subseteq \mathbb{R}^n$.

Lemma 1. *Let $\mathcal{C} \subseteq \mathbb{R}^n$ be a convex set that contains the origin. Let $f : \mathbb{R} \to \mathbb{R}$ be a concave function. The following set is convex:*

$$\mathcal{S} = \{(x, y) : f(y) > 0,\ x \in f(y)\mathcal{C}\}.$$

In order to formulate collision avoidance constraints for an obstacle $\mathcal{O} \subseteq \mathbb{R}^n$, it is useful to characterize the set of *valid* inequalities. We say the inequality $a^\top x + b \geq 0$ is *valid* if it holds for all $x \in \mathcal{O}$.

Definition 1. *The polar $\mathcal{O}^\circ$ of $\mathcal{O} \subseteq \mathbb{R}^n$ is the set of all valid inequalities:*

$$\mathcal{O}^\circ := \left\{(a, b) \in \mathbb{R}^{n+1} :\ a^\top x + b \geq 0 \ \text{ for all } x \in \mathcal{O}\right\}. \quad (2)$$

Lemma 2. *Let $\mathcal{O} \subseteq \mathbb{R}^n$. Its polar, $\mathcal{O}^\circ \subseteq \mathbb{R}^{n+1}$, is a convex cone. (Even when $\mathcal{O}$ is not convex.)*

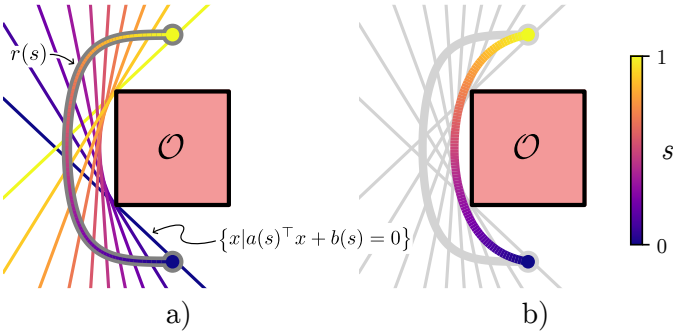

Fig. 1. A single alternation: the plane update (left) computes maximum-margin separating planes around the current trajectory, and the trajectory update (right) re-optimizes the trajectory with the planes fixed.

IV. Convex Formulation of the Minimum-Time Objective

In a similar fashion to [20], we reparameterize the trajectory over the unit interval via $r : [0,1] \to \mathbb{R}^n$ where $q(t) = r(t/T)$, so that $q^{(i)}(t) = T^{-i} r^{(i)}(t/T)$. The derivative constraints (1d) become $r^{(i)}(s) \in T^i \mathcal{C}_i$, which are nonconvex in T. Substituting $T_I = T^I$, these become $r^{(i)}(s) \in T_I^{i/I} \mathcal{C}_i$, which are convex by Lemma 1 since $T_I^{i/I}$ is concave and $\mathcal{C}_i$ contains the origin. Since $T_I^{1/I}$ is monotonically increasing, minimizing T_I is equivalent to minimizing T. This yields:

$$\begin{aligned}
\text{minimize} \quad & T_I && \text{(3a)}\\
\text{subject to} \quad & T_I > 0, && \text{(3b)}\\
& r(0) = q_0,\ r(1) = q_T, && \text{(3c)}\\
& r^{(i)}(0) = r^{(i)}(1) = 0, \quad i = 1, \dots, I, && \text{(3d)}\\
& r^{(i)}(s) \in T_I^{i/I} \mathcal{C}_i, \quad i = 1, \dots, I,\ s \in [0,1], && \text{(3e)}\\
& r(s) \notin \mathcal{O}_k, \quad k = 1, \dots, K,\ s \in [0,1]. && \text{(3f)}
\end{aligned}$$

All constraints are convex except for collision avoidance (3f).

V. Polar Reformulation of Collision-Avoidance Constraints

We replace the nonconvex constraints (3f) with a search for time-varying separating hyperplanes $(a_k, b_k) : [0,1] \to \mathcal{O}_k^\circ$ that separate r from each obstacle $k = 1, \dots, K$:

$$\begin{aligned}
\text{minimize} \quad & \text{(3a)} && \text{(4a)}\\
\text{subject to} \quad & \text{(3b)} - \text{(3e)}, && \text{(4b)}\\
& a_k(s)^\top r(s) + b_k(s) < 0, && \text{(4c)}\\
& (a_k(s), b_k(s)) \in \mathcal{O}_k^\circ. && \text{(4d)}
\end{aligned}$$

By Lemma 2, $\mathcal{O}_k^\circ$ is convex, so the only nonconvexity is the bilinearity between a_k and r in (4c). The problem becomes convex when either the trajectory r or the planes (a_k, b_k) are fixed, motivating an alternating approach.

VI. Biconvex Procedure

Our biconvex procedure (BCP) alternates between a *plane update* (fixing r, computing separating planes) and a *trajectory update* (fixing planes, re-optimizing r and T_I). Following [21, §3.3], we only add planes for obstacles that the current candidate actually collides with, which allows the trajectory to jump around obstacles and avoid some local minima. The procedure is outlined in Alg. 1 and illustrated in Fig. 2.

Algorithm 1: Biconvex Procedure

Input: Feasible path r_{init}
Output: Optimized, collision-free trajectory (r, T).
Algorithm:
$\mathcal{P} \leftarrow \emptyset$, $\mathcal{I} \leftarrow \emptyset$, $r \leftarrow r_{\text{init}}$
while *not converged* **do**
  repeat
    $(c, T_c) \leftarrow$ TrajectoryUpdate$(\mathcal{P})$
    $\mathcal{I}_c \leftarrow$ GetObstacleCollisions(c)
    $\mathcal{I} \leftarrow \mathcal{I} \cup \mathcal{I}_c$
    $\mathcal{P} \leftarrow \mathcal{P} \cup$ PlaneUpdate$(\mathcal{I}_c, r)$
  until $\mathcal{I}_c = \emptyset$;
  $(r, T) \leftarrow (c, T_c)$
  $\mathcal{P} \leftarrow$ PlaneUpdate$(\mathcal{I}, r)$
end
return (r, T)

Plane update. Given a fixed feasible trajectory r, we compute maximum-margin separating planes by projecting $r(s)$ onto each tagged obstacle $\mathcal{O}_k$:

$$a_k^\star(s) = \frac{\text{proj}_{\mathcal{O}_k}[r(s)] - r(s)}{\|\text{proj}_{\mathcal{O}_k}[r(s)] - r(s)\|_2}, \tag{5a}$$

$$b_k^\star(s) = -a_k^\star(s)^\top \text{proj}_{\mathcal{O}_k}[r(s)]. \tag{5b}$$

This maximizes the margin between the planes and the trajectory, leaving room for improvement in subsequent iterations.

Trajectory update. We fix the planes and solve (4), which is now convex, yielding a candidate trajectory c and duration T_c.

VII. Finite-Dimensional Formulation

We discretize both the trajectory r and the plane functions (a_k, b_k) as Bézier splines with M segments. We denote the d-th control point of a curve f as π_d^f. The convex hull property of Bézier curves lets us enforce all continuous-time constraints (derivative bounds, collision avoidance, polar membership) conservatively via linear constraints on control points. Endpoint and continuity constraints follow from standard Bézier derivative and interpolation properties [22]. For the collision-avoidance constraints, we compute the scalar polynomial $v_{m,k}(s) = a_{m,k}(s)^\top r_m(s) + b_{m,k}(s)$ in Bernstein form and require all its control points to be negative, which is a conservative but effective sufficient condition.

The trajectory update requires no special treatment: fixing the plane control points, problem (4) reduces to a second-order cone program in the trajectory control points and T_I.

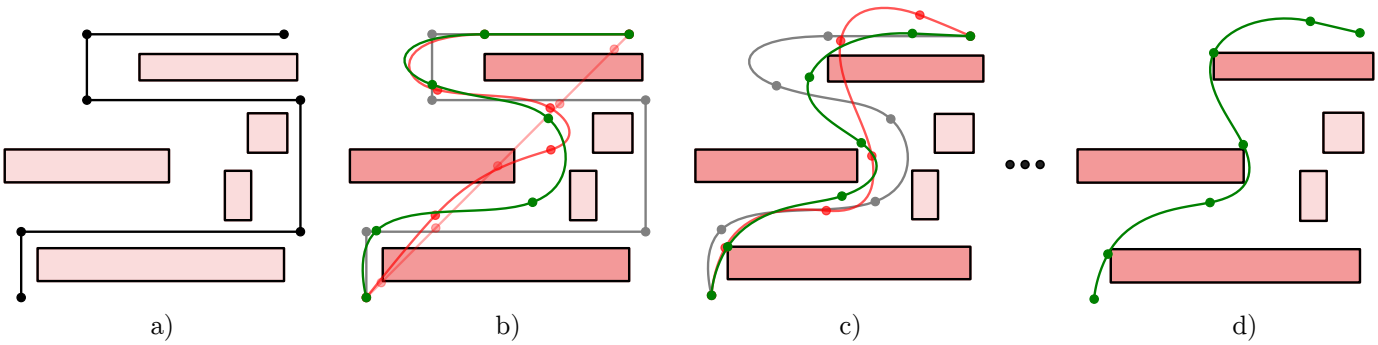

Fig. 2. Overview of the proposed biconvex procedure on a 2D environment. Grey: last feasible trajectory. Red: infeasible iterates. Green: new feasible iterate. a) Initial collision-free path. b) First iteration: the trajectory shortcuts through obstacles, triggering plane additions until a feasible iterate is found. c) Planes are updated around the new feasible trajectory, giving room to improve. d) Converged solution that happened to jump around the two obstacles on the right.

The finite-dimensional plane update, however, differs from the closed-form solution (5): projecting r onto an obstacle does not generally yield a polynomial trajectory, so we instead solve a convex surrogate for each tagged obstacle-segment pair (k, m):

$$\begin{aligned} \text{minimize} \quad & \max_{j\in\{0,\dots,J\}} \pi_j^{v_{m,k}} && \text{(6a)} \\ \text{subject to} \quad & (\pi_d^{a_{m,k}}, \pi_d^{b_{m,k}}) \in \mathcal{O}_k^\circ, \quad d = 0,\dots,D, && \text{(6b)} \\ & \|\pi_d^{a_{m,k}}\|_2 \le 1, \quad d = 0,\dots,D, && \text{(6c)} \end{aligned}$$

where D is the degree of the plane curves, J is the degree of the evaluated plane curve $v_{m,k}$, and $\pi_d^{v_{m,k}}$ are its control points in Bernstein form. The objective upper-bounds $\sup_s v_{m,k}(s)$ by the convex hull property, and the constraints on the plane control points ensure polar membership and separation hold continuously. This maximizes the margin where it is tightest, ensuring the trajectory update has room to improve. See Appendix D for a more detailed discussion of this surrogate program. We observe that this discretization guarantees that we find a new feasible trajectory in at most MK iterations of adding new planes that does not have a higher cost than the previous feasible trajectory.

VIII. Experimental Evaluation

We evaluate our BCP approach on a dual-arm pallet unloading problem in task space with convex obstacles, shown in Fig. 3. This task is designed to test the efficacy of our more general approach against SCSPlanning [12], which is specifically designed to excel at these types of planning problems.

In this task, two Franka Research 3 manipulators cooperatively move three blue and three red packages from a central pallet to their corresponding blue and red offload zones. Each offload task consists of six trajectories: three from the offload zone to the pallet, and three return trajectories. The dimensions and the placement of the packages are randomized. We plan in the six-dimensional task space where the first three dimensions represent the position of one gripper, and the last three represent the position of the other. After optimizing the trajectory in this space, we compute the robot configurations via inverse kinematics. See Appendix F for details. All collision geometries are modeled as axis-aligned boxes, including the robot bases, grippers, packages, bins, and pallet. The collision geometry setup is shown in Fig. 3.

For a pair of axis-aligned boxes centered at $p_1, p_2 \in \mathbb{R}^3$ with half-widths $h_1, h_2 \in \mathbb{R}^3$, the set of configurations where the two boxes overlap can be written in closed form as the following convex polyhedron in $\mathbb{R}^6$

$$\mathcal{O} = \{(p_1, p_2) : |p_{2,i} - p_{1,i}| \le h_{1,i} + h_{2,i},\ i = 1,2,3\}\,.$$

Its polar is computed in Appendix B.

We compare our approach to three baselines: two nonlinear formulations that solve (1) directly with an off-the-shelf NLP solver (differing in how collision avoidance is enforced, see Appendix E), and SCSPlanning [12] (henceforth referred to as SCS), which optimizes through sequences of convex safe sets computed via edge inflation [11] (referred to as EI, see Appendix G).

For all approaches, we initialize with a simple waypoint-based planner that moves the grippers concurrently in a sequence of straight-line motions. The polygonal curves in Fig. 3 show an example of this initialization. We employ the conservative collision checker from Appendix C for all approaches, which guarantees that no trajectory is incorrectly classified as collision-free.

For all approaches we optimize degree five Bézier splines, which guarantee the feasibility of the initial polygonal trajectory with velocity and acceleration constraints. We do not enforce continuous acceleration between trajectory segments, as these constraints are not supported by the SCS+EI baseline. For the BCP approach we use a plane degree of 1. We run both SCS+EI and BCP until the costs change less than 1% between iterations. For the nonconvex approach with the discretized collision avoidance constraints we use 60 evenly spaced constraints with an influence distance of $3\times10^{-2}m$ and lower bound of $5 \times 10^{-3}m$ per segment. We initialize BCP with the same approach as SCS, see [12, § VI]. The results across 50 random instances (300 trajectory optimizations per approach) are summarized in Tab. I.

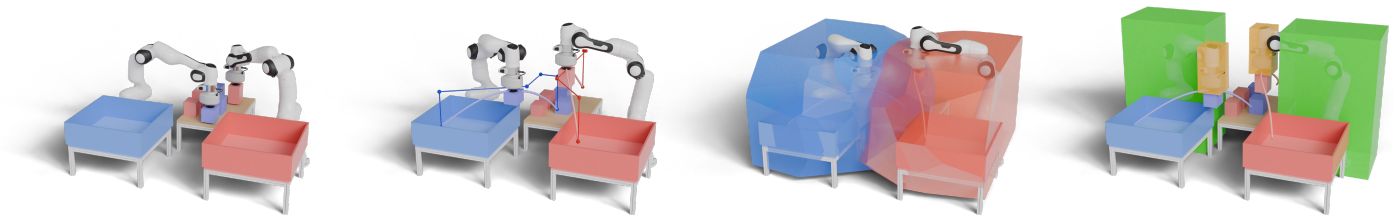

Fig. 3. Dual-arm pallet unloading in task space. From left to right: *(first)* The task: two Franka manipulators move packages from a central pallet to their respective offload zones. *(second)* The waypoint-based initialization (straight-line segments) and the optimized minimum-time trajectories with our proposed BCP approach. *(third)* Visualization of the valid domains for each gripper position. *(fourth)* Visualization of the axis-aligned bounding box collision geometries used during trajectory optimization.

TABLE I
STATISTICS OF RANDOMIZED DUAL-ARM PALLET UNLOADING TASKS (N=50). ALL RESULTS ARE REPORTED AS MEAN ± STD.

	Waypoint	Nonconvex (polar)	Nonconvex (disc.)	SCS+EI	BCP (ours)
Traj. duration [s]	6.33 ± 0.72	2.72 ± 0.42	2.62 ± 0.25	$\mathbf{2.52 \pm 0.26}$	2.59 ± 0.24
Computation time [ms]	7.1 ± 4.0	20567.5 ± 10348.3	2225.4 ± 3416.5	$\mathbf{206.9 \pm 33.4}$	257.4 ± 86.4
Success rate [%]	**100.0**	99.0	97.3	**100.0**	**100.0**
Collision-free [%]	**100.0**	81.1	**100.0**	**100.0**	**100.0**

Both our BCP approach and the SCS+EI baseline achieve a 100% success rate and produce fully collision-free trajectories on all instances. The nonlinear baselines are substantially slower and can fail or produce collisions. The polar nonconvex baseline requires over 20 seconds on average and produces collisions in 18.9% of instances, while the discretized nonconvex baseline is collision-free but still takes over two seconds and fails on 2.7% of instances. The reason the polar nonconvex formulation produces many collisions is that the solver tolerances needed to be increased in order to get the programs to solve. Our approach produces trajectories that are slightly longer on average than the SCS+EI baseline with 2.59 vs. 2.52 seconds average duration, while requiring comparable computation times on average (257 ms vs. 207 ms). The timing breakdown of our BCP approach compared to SCS+EI is shown in Fig. 4. We observe that around a quarter of the computation time in the SCS+EI approach is spent on computing the regions in the sequence of convex sets.

While there is a gap in the performance between our BCP approach and the SCS+EI baseline, we note that our approach is more general in the derivative constraints it supports, does not require pre-computed sets and we observed it to be less sensitive to the initialization. As an illustrative example of this we reran the SCS+EI and BCP experiments with a slight modification to the waypoint planner we use to initialize both approaches: Instead of moving both arms concurrently, we move the arms along the same trajectories but sequentially. The results are shown in Tab. II. The initialization is now significantly weaker taking 10.6 seconds on average. SCS+EI now only recovers trajectories of duration 3.29 seconds on average because the sets constructed around this initial trajectory fail to cover high-quality paths which have both arms moving fully concurrently. The BCP approach, on the other hand, requires a bit longer computation time but recovers trajectories of duration 2.49 seconds, on average, which is on par with the performance with the stronger concurrent initialization.

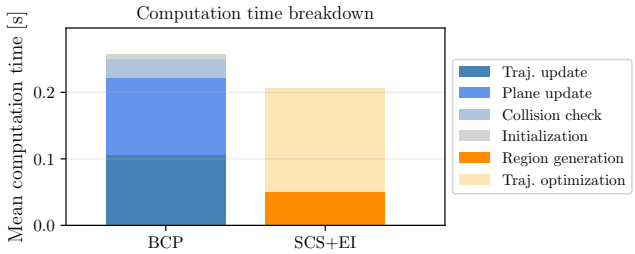

Fig. 4. Timing breakdown of our BCP approach and the SCS+EI baseline. We see that around a quarter of the computation time in the SCS+EI approach is spent on constructing the sequence of convex sets.

TABLE II
STATISTICS OF RANDOMIZED DUAL-ARM PALLET UNLOADING TASKS USING SEQUENTIAL WAYPOINT PLANNER (N=50). ALL RESULTS ARE REPORTED AS MEAN ± STD.

	Waypoint	SCS+EI	BCP (ours)
Traj. duration [s]	10.60 ± 1.43	3.29 ± 0.41	$\mathbf{2.49 \pm 0.43}$
Computation time [ms]	10.8 ± 5.3	$\mathbf{325.9 \pm 57.8}$	506.8 ± 168.3
Success rate [%]	**100.0**	99.7	**100.0**
Collision-free [%]	**100.0**	**100.0**	**100.0**

IX. CONCLUSION

We presented a biconvex procedure for minimum-time motion planning that convexifies the objective and derivative constraints to arbitrary order and handles collision avoidance through time-varying separating hyperplanes. The method requires no precomputed convex decomposition and is robust to initialization quality. On a dual-arm pallet unloading benchmark, it matches the performance of a state-of-the-art decomposition-based planner while offering broader constraint support and greater flexibility. Future work includes extending to nonconvex obstacles in configuration space and leveraging hardware-acceleration to further improve performance.

REFERENCES

[1] 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.

[2] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 2002.

[3] R. Geraerts and M. H. Overmars, "Creating high-quality paths for motion planning," *The international journal of robotics research*, vol. 26, no. 8, pp. 845–863, 2007.

[4] 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.

[5] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *The International Journal of Robotics Research*, vol. 33, no. 9, pp. 1251–1270, 2014.

[6] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "Chomp: Gradient optimization techniques for efficient motion planning," in *2009 IEEE international conference on robotics and automation*. IEEE, 2009, pp. 489–494.

[7] B. Sundaralingam, S. K. S. Hari, A. Fishman, C. Garrett, K. Van Wyk, V. Blukis, A. Millane, H. Oleynikova, A. Handa, F. Ramos *et al.*, "Curobo: Parallelized collision-free robot motion generation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 8112–8119.

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

[9] T. Marcucci, M. Petersen, D. von Wrangel, and R. Tedrake, "Motion planning around obstacles with convex optimization," *Science robotics*, vol. 8, no. 84, p. 7843, 2023.

[10] T. Marcucci, P. Nobel, R. Tedrake, and S. Boyd, "Fast path planning through large collections of safe boxes," *IEEE Transactions on Robotics*, 2024.

[11] P. Werner, R. Cheng, T. Stewart, R. Tedrake, and D. Rus, "Superfast configuration-space convex set computation on gpus for online motion planning," *arXiv preprint arXiv:2504.10783*, 2025.

[12] T. Marcucci, M. Halm, W. Yang, D. Lee, and A. D. Marchese, "A biconvex method for minimum-time motion planning through sequences of convex sets," *Accepted for publication in Robotics: Science and Systems Foundation*, 2025.

[13] J. Chen, T. Liu, and S. Shen, "Online generation of collision-free trajectories for quadrotor flight in unknown cluttered environments," in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 1476–1483.

[14] S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, and V. Kumar, "Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-d complex environments," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1688–1695, 2017.

[15] Y. Wu, I. Spasojevic, P. Chaudhari, and V. Kumar, "Optimal convex cover as collision-free space approximation for trajectory generation," *arXiv preprint arXiv:2406.09631*, 2024.

[16] R. Natarajan, C. Liu, H. Choset, and M. Likhachev, "Implicit graph search for planning on graphs of convex sets," *arXiv preprint arXiv:2410.08909*, 2024.

[17] S. Y. C. Chia, R. H. Jiang, B. P. Graesdal, L. P. Kaelbling, and R. Tedrake, "Gcs*: Forward heuristic search on implicit graphs of convex sets," *arXiv preprint arXiv:2407.08848*, 2024.

[18] T. Marcucci, "Graphs of convex sets with applications to optimal control and motion planning," Ph.D. dissertation, Massachusetts Institute of Technology, 2024.

[19] R. Deits and R. Tedrake, "Computing large convex regions of obstacle-free space through semidefinite programming," in *Algorithmic Foundations of Robotics XI: Selected Contributions of the Eleventh International Workshop on the Algorithmic Foundations of Robotics*. Springer, 2015, pp. 109–124.

[20] M. Leomanni, G. Costante, and F. Ferrante, "Time-optimal control of a multidimensional integrator chain with applications," *IEEE Control Systems Letters*, vol. 6, pp. 2371–2376, 2022.

[21] T. Lipp and S. Boyd, "Variations and extension of the convex–concave procedure," *Optimization and Engineering*, vol. 17, pp. 263–287, 2016.

[22] R. T. Farouki and V. Rajan, "Algorithms for polynomials in Bernstein form," *Computer Aided Geometric Design*, vol. 5, no. 1, pp. 1–26, 1988.

[23] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge University Press, 2004.

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

[25] 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.

APPENDIX

A. A Recipe for Computing the Polar of Convex Sets in Conic Standard Form

Let $\mathcal{O} = \{x \in \mathbb{R}^n \mid \exists y \in \mathbb{R}^m : Ax + By + b \in \mathcal{K}\}$ be a convex set described in conic form, where $\mathcal{K}$ is a closed convex cone with dual cone $\mathcal{K}^*$ (see [23, §2.6]). The homogenization of $\mathcal{O}$ reads

$$\tilde{\mathcal{O}} = \{(z,t) \in \mathbb{R}^{n+1} \mid \exists y : Az + By + bt \in \mathcal{K},\ t \geq 0\},$$

and the polar $\mathcal{O}^\circ$ is its dual cone $\tilde{\mathcal{O}}^*$, see [18, §2.1.2]. We have $(\lambda, \omega) \in \mathcal{O}^\circ$ if and only if

$$\lambda^\top z + \omega t \geq 0 \quad \text{for all } (z,t) \in \tilde{\mathcal{O}}.$$

Therefore, the element (λ, ω) is in the polar if the minimum of $\lambda^\top z + \omega t$ subject to $Az + By + bt \in \mathcal{K}$, $t \geq 0$ is nonnegative. The Lagrangian of this minimization reads

$$\mathcal{L} = z^\top(\lambda - A^\top \nu) - y^\top B^\top \nu + t(\omega - b^\top \nu - \theta),$$

with $\nu \in \mathcal{K}^*$ and $\theta \geq 0$. For the minimum over the primal variables (z, y, t) to be finite, we require $\lambda = A^\top \nu$, $B^\top \nu = 0$, and $\omega \geq b^\top \nu$ (absorbing θ). The polar is therefore

$$\mathcal{O}^\circ = \left\{ (\lambda, \omega) \in \mathbb{R}^{n+1} \;\middle|\; \begin{array}{l} \lambda = A^\top \nu,\ B^\top \nu = 0, \\ \omega \geq b^\top \nu,\ \nu \in \mathcal{K}^* \end{array} \right\}. \tag{7}$$

In order to formulate obstacle avoidance constraints for a given $\mathcal{O}$, we can simply add a separating plane (λ, ω) by introducing auxiliary decision variables ν and adding the constraints on the decision variables (λ, ω, ν) in (7).

B. Polar of the Box-Box Obstacle

Consider the box-overlap obstacle from §VIII

$$\mathcal{O} = \left\{ (p_1, p_2) \in \mathbb{R}^6 \mid |p_{2,i} - p_{1,i}| \leq h_{1,i} + h_{2,i},\ i = 1,2,3 \right\}.$$

Since $\mathcal{O}$ is invariant under common translations $(p_1 + v, p_2 + v)$, any valid inequality $\hat{a}^\top (p_1, p_2) + b \geq 0$ requires $\hat{a} = (-a, a)$ for some $a \in \mathbb{R}^3$. The validity condition then reduces to

$$a^\top \delta + b \geq 0, \quad \text{for all } |\delta_i| \leq h_{1,i} + h_{2,i},$$

where $\delta = p_2 - p_1$. The minimum of $a^\top \delta$ over this box is

$$-\sum_{i=1}^{3} |a_i|(h_{1,i} + h_{2,i}),$$

which is attained at

$$\delta_i^\star = -\text{sign}(a_i)(h_{1,i} + h_{2,i}).$$

The polar is therefore

$$\mathcal{O}^\circ = \left\{ \begin{pmatrix} -a & a \end{pmatrix}, b) \in \mathbb{R}^7 \;\middle|\; b \geq \sum_{i=1}^{3} |a_i|(h_{1,i} + h_{2,i}) \right\}.$$

C. Conservative Collision Checking for Bézier Curves

We check whether a Bézier curve $B(s) = \sum_{d=0}^{D} \beta_d(s)\pi_d$ avoids an H-polyhedron obstacle $\mathcal{O} = \{x \in \mathbb{R}^n \mid Cx \leq d\}$ using the recursive subdivision procedure in Alg. 2. The algorithm is conservative: it never falsely certifies a colliding curve as collision-free. The key insight is that by the convex hull property of Bézier curves, if all control points lie outside a single halfplane of $\mathcal{O}$, the entire curve is guaranteed to avoid $\mathcal{O}$. When this test is inconclusive, we split the curve in half via De Casteljau subdivision [22, §2.4], and recursively run the same check on the two new pieces. This subdivision tightens the convex hull around the curve, making the test increasingly precise. We terminate if either an endpoint of the curve is inside an obstacle, we find a single hyperplane of the obstacle that separates all control points from the obstacle, or the segment is so small that all of its control points fit in a user-specified ε-ball.

Algorithm 2: COLLISIONFREE(B, C, d, ε)

Input: Bézier curve B with control points $\pi_0, \ldots, \pi_D$, obstacle $\mathcal{O} = \{x \mid Cx \leq d\}$, tolerance ϵ
Output: `true` if B is certified collision-free, `false` otherwise

if $\pi_0 \in \mathcal{O}$ ***or*** $\pi_D \in \mathcal{O}$ **then**
  return `false`; `// endpoint in collision`
end
if $\exists j$ *s.t.* $c_j^\top \pi_d > d_j$ *for all* $d = 0, \ldots, D$ **then**
  return `true`; `// all ctrl pts outside halfplane` j
end
if *bounding radius of* $\{\pi_0, \ldots, \pi_D\} \leq \varepsilon$ **then**
  return `false`; `// conservative: tolerance reached`
end
Split B at midpoint into B_L, B_R
return COLLISIONFREE(B_L, C, d, ε) **and** COLLISIONFREE(B_R, C, d, ε)

D. Comments on the Maximum-Margin Planes

We divide the trajectory into M equal-duration segments. For each obstacle $\mathcal{O}_k$, let $\mathcal{I}_k$ be the set of segment indices whose candidate trajectories have collided with $\mathcal{O}_k$. We define the active domain as

$$\mathcal{S}_k := \bigcup_{m \in \mathcal{I}_k} \left[\frac{m}{M}, \frac{m+1}{M}\right]. \tag{8}$$

Finding the maximum margin separating planes corresponds to solving

$$\begin{aligned}
\text{minimize} \quad & \int_{\mathcal{S}_k} a_k(s)^\top r(s) + b_k(s) ds, && \text{(9a)}\\
\text{subject to} \quad & (a_k(s), b_k(s)) \in \mathcal{O}_k^\circ, && \text{(9b)}\\
& \|a_k\|_2 \leq 1, && \text{(9c)}\\
& a_k(s)^\top r(s) + b_k(s) < 0, . && \text{(9d)}
\end{aligned}$$

In this section we derive the dual of (9) and verify that an optimal solution to this primal-dual pair corresponds to the one stated in (5).

First, we observe that the constraint (9d) is redundant since there are no continuity constraints on (a, b). We then introduce auxiliary variables $z_0(s) = 1$ and $z_1(s) = a(s)$ to separate the norm constraint from the decision variable:

$$\begin{aligned}
\text{minimize} \quad & \int_{\mathcal{S}_k} a(s)^\top r(s) + b(s)\, ds, && \text{(10a)}\\
\text{subject to} \quad & (a(s), b(s)) \in \mathcal{O}^\circ, && \text{(10b)}\\
& \|z_1(s)\|_2 \leq z_0(s), && \text{(10c)}\\
& z_0(s) = 1, && \text{(10d)}\\
& a(s) = z_1(s). && \text{(10e)}
\end{aligned}$$

We can now write the Lagrangian

$$\begin{aligned}
\mathcal{L} = \int_{\mathcal{S}_k} \Big[& a(s)^\top r(s) + b(s)\\
& - \begin{bmatrix} \Xi_a(s)^\top & \Xi_b(s) \end{bmatrix} \begin{bmatrix} a(s) \\ b(s) \end{bmatrix}\\
& - \begin{bmatrix} \Pi_0(s) & \Pi_1(s)^\top \end{bmatrix} \begin{bmatrix} z_0(s) \\ z_1(s) \end{bmatrix} + \nu(s)(z_0(s) - 1)\\
& + \Gamma(s)^\top (a(s) - z_1(s)) \Big] ds
\end{aligned} \tag{11}$$

with dual variables $(\Xi_a(s), \Xi_b(s)) \in \tilde{\mathcal{O}}$, $(\Pi_0(s), \Pi_1(s)) \in \mathcal{L}_2$, $\nu(s) \in \mathbb{R}$, and $\Gamma(s) \in \mathbb{R}^n$. Here we used the fact that the dual of the polar $\tilde{\mathcal{O}}$ is the homogenization $\tilde{\mathcal{O}}$ of the obstacle [18, §2.4.1], and the fact that the second order cone $\mathcal{L}_2$ is self-dual. Rearranging terms gives

$$\begin{aligned}
\mathcal{L} = \int_{\mathcal{S}_k} \Big[& a(s)^\top (r(s) - \Xi_a(s) + \Gamma(s))\\
& + b(s)(1 - \Xi_b(s)) + z_1(s)^\top (\Pi_1(s) - \Gamma(s))\\
& + z_0(s)(\nu(s) - \Pi_0(s)) - \nu(s) \Big] ds.
\end{aligned} \tag{12}$$

For the minimization of $\mathcal{L}$ over the primal variables to have a finite value, sufficient conditions for all $s \in \mathcal{S}_k$ are

$$\begin{aligned}
\Xi_a(s) &= r(s) + \Gamma(s), && \text{(13)}\\
\Xi_b(s) &= 1, && \text{(14)}\\
\Pi_1(s) &= \Gamma(s), && \text{(15)}\\
\nu(s) &= \Pi_0(s). && \text{(16)}
\end{aligned}$$

The dual is therefore

$$\begin{aligned}
\text{maximize} \quad & \int_{\mathcal{S}_k} -\nu(s)\, ds, && \text{(17a)}\\
\text{subject to} \quad & \|\Gamma(s)\|_2 \leq \nu(s), && \text{(17b)}\\
& r(s) + \Gamma(s) \in \mathcal{O}_k, && \text{(17c)}
\end{aligned}$$

where the first constraint follows from $(\Pi_0, \Pi_1) \in \mathcal{L}_2$ together with (15) and (16), and the second follows from $(\Xi_a, \Xi_b) \in \tilde{\mathcal{O}}$ together with (13) and (14) (since $(\cdot, 1) \in \tilde{\mathcal{O}}$ is equivalent to $(\cdot) \in \mathcal{O}$). Since $\nu(s) \geq \|\Gamma(s)\|_2$ and the dual maximizes $-\nu$,

the optimal choice is $\nu(s) = \|\Gamma(s)\|_2$, so the dual is equivalent to

$$\begin{aligned} \text{minimize} \quad & \int_{\mathcal{S}_k} \|\Gamma(s)\|_2 \, ds, && \text{(18a)} \\ \text{subject to} \quad & r(s) + \Gamma(s) \in \mathcal{O}_k. && \text{(18b)} \end{aligned}$$

This is a pointwise projection problem: the optimal $\Gamma^\star(s)$ is the displacement from $r(s)$ to its closest point in $\mathcal{O}_k$, i.e., $\Gamma^\star(s) = r^\star(s) - r(s)$ where $r^\star(s) = \text{proj}_{\mathcal{O}_k}(r(s))$.

The corresponding optimal primal solution is

$$a^\star(s) = \frac{r^\star(s) - r(s)}{\|r^\star(s) - r(s)\|_2}, \tag{19}$$

$$b^\star(s) = -a^{\star\top}(s)\, r^\star(s), \tag{20}$$

which is the supporting hyperplane of $\mathcal{O}_k$ at $r^\star(s)$ with normal pointing toward $\mathcal{O}_k$.

We now verify optimality via the KKT conditions. Stationarity is already encoded in (13)–(16). Dual feasibility of $\Gamma^\star$ holds since $r(s) + \Gamma^\star(s) = r^\star(s) \in \mathcal{O}_k$. Primal feasibility of $(a^\star, b^\star)$ follows from convexity of $\mathcal{O}_k$: the projection theorem gives $(x - r^\star(s))^\top (r(s) - r^\star(s)) \le 0$ for all $x \in \mathcal{O}_k$, which rearranges to $a^{\star\top}(s)\, x + b^\star(s) \ge 0$, confirming $(a^\star(s), b^\star(s)) \in \mathcal{O}_k^\circ$. It remains to check complementary slackness on the four dual-variable terms. The two equality-constraint terms (ν and Γ) are trivially zero since $z_0^\star = 1$ and $a^\star = z_1^\star$ hold exactly. For the polar-cone term:

$$\Xi_a^{\star\top}(s)\, a^\star(s) + \Xi_b^\star(s)\, b^\star(s) = r^{\star\top}(s)\, a^\star(s) + b^\star(s) = 0, \tag{21}$$

since $b^\star(s) = -a^{\star\top}(s)\, r^\star(s)$. For the second-order cone constraint, we substitute $\Pi_0^\star(s) = -\|\Gamma^\star(s)\|_2$, $\Pi_1^\star(s) = \Gamma^\star(s) = r^\star(s) - r(s)$, $z_0^\star = 1$, and $z_1^\star(s) = a^\star(s)$:

$$\Pi_0^\star(s)\, z_0^\star + \Pi_1^{\star\top}(s)\, z_1^\star(s) = \tag{22}$$

$$-\|r^\star(s) - r(s)\|_2 + \frac{(r^\star(s) - r(s))^\top (r^\star(s) - r(s))}{\|r^\star(s) - r(s)\|_2} = 0. \tag{23}$$

This confirms that the proposed primal-dual pair satisfies all KKT conditions and is therefore optimal.

a) Connection to the finite-dimensional surrogate.: The finite-dimensional plane update (6) uses a supremum cost (via the largest control point) rather than the integral cost in (9). This choice is important in the practical regime of finite degree and a moderate number of segments: the supremum cost guarantees that, when possible, the margin is positive everywhere on the segment, ensuring the subsequent trajectory update has room to improve. An integral cost, while a natural alternative, tends to trade off letting the margin collapse to zero at some points in favor of a lower overall cost, which can leave the trajectory update with no feasible improving direction. As $M \to \infty$, each segment shrinks, the convex hull of the control points tightens around the curve, and the surrogate converges to a pointwise minimization of the margin, recovering the continuous solution (5).

E. Nonlinear Baselines

Both nonlinear baselines solve the minimum-time problem (1) directly as a nonlinear program, using the same Bézier spline parameterization as our approach. We describe the common formulation first and then the two different collision-avoidance strategies.

a) Common formulation: We represent the normalized trajectory r as a Bézier spline with M segments, each a degree-D curve with control points $\pi_d^{(m)} \in \mathbb{R}^n$ for $d = 0, \dots, D$ and $m = 1, \dots, M$. Unlike our approach, both baselines keep a single per-segment time variable $T > 0$ rather than $T_I = T^I$, so the total trajectory duration is MT.

To enforce the derivative constraints (1d), we introduce auxiliary normalized velocity control points $p_{m,d}^v \in \mathbb{R}^n$ for $d = 0, \dots, D-1$, and normalized acceleration control points $p_{m,d}^a \in \mathbb{R}^n$ for $d = 0, \dots, D-2$. The d-th control point of the normalized velocity curve on segment m is $D(\pi_{d+1}^{(m)} - \pi_d^{(m)})$ by the derivative property of Bézier curves. Relating this to the actual velocity $q^{(1)} = r'/T$ gives the bilinear equality

$$D\big(\pi_{d+1}^{(m)} - \pi_d^{(m)}\big) = T\, p_{m,d}^v, \quad d = 0, \dots, D-1. \tag{24}$$

Applying the derivative property again to the velocity curve (whose control points are $T p_{m,d}^v$) and relating to the actual acceleration $q^{(2)} = r''/T^2$ gives

$$(D-1)\big(p_{m,d+1}^v - p_{m,d}^v\big) = T\, p_{m,d}^a, \quad d = 0, \dots, D-2. \tag{25}$$

The derivative constraints then become linear: $p_{m,d}^v \in \mathcal{C}_1$ and $p_{m,d}^a \in \mathcal{C}_2$. C^1 and C^2 continuity between segments is enforced by equating the last velocity (resp. acceleration) control point of segment m with the first of segment $m+1$. Both baselines therefore solve

$$\begin{aligned} \text{minimize} \quad & T && \text{(26a)} \\ \text{subject to} \quad & T > 0, && \text{(26b)} \\ & \text{boundary conditions (1b), (1c),} && \text{(26c)} \\ & \text{(24), (25),} && \text{(26d)} \\ & p_{m,d}^v \in \mathcal{C}_1,\ p_{m,d}^a \in \mathcal{C}_2, && \text{(26e)} \\ & C^1, C^2 \text{ continuity,} && \text{(26f)} \\ & \text{(collision avoidance, see below),} && \text{(26g)} \end{aligned}$$

with the nonlinear program solved by the off-the-shelf solver SNOPT.

b) Baseline 1: biconvex collision avoidance.: The first baseline adds separating planes $(a_{m,k}, b_{m,k})$ for each segment m and obstacle $\mathcal{O}_k$, using exactly the biconvex collision-avoidance constraints (4c)–(4d) from §V. The control points of $a_{m,k}(s)^\top r_m(s) + b_{m,k}(s)$ are constrained to be negative, producing a bilinear coupling between the plane variables and the trajectory control points. Unlike our BCP approach, which alternates between fixing the planes and the trajectory, this baseline optimizes both simultaneously in a single nonlinear solve.

c) Baseline 2: discretized minimum distance.: The second baseline avoids plane variables entirely. For each segment m and each of J uniformly-spaced sample parameters $s_j \in [0,1]$, we introduce an auxiliary variable $q_j^{(m)} \in \mathbb{R}^n$ with the linear equality

$$q_j^{(m)} = r_m(s_j) = \sum_{d=0}^{D} \beta_d(s_j)\, \pi_d^{(m)}, \tag{27}$$

and enforce collision avoidance at each sample point via Drake's `MinimumDistanceLowerBoundConstraint` [24]. This constraint evaluates the signed distances between all geometry pairs registered in the scene graph at configuration $q_j^{(m)}$, and enforces that the minimum signed distance is non-negative using a smooth penalty formulation.

F. Pallet Unloading Experiment Details

a) Package randomization.: Each random instance places 6 packages (3 red, 3 blue) on the pallet. Package dimensions are sampled independently and uniformly from $[8, 10, 5]$ cm to $[12, 12, 15]$ cm (width $\times$ depth $\times$ height). The 2D position of each package on the pallet is sampled uniformly within 95% of the available pallet width and 90% of the available pallet depth. Packages are placed on top of the pallet surface at the appropriate height. Placement is generated by rejection sampling until all packages are mutually separated by at least 5.5 cm.

After placement, packages are sorted by their distance to the pallet center and assigned colors (red / blue) by selecting uniformly at random from the subset of red/blue assignments (3 of each) for which all package grasp points lie within the respective arm's reachable workspace, with a 5 cm inset from the workspace boundary. Instances for which no valid assignment exists are rejected and resampled.

b) Inverse kinematics.: After optimizing the task-space trajectory, we compute joint-space configurations via a two-stage inverse kinematics (IK) procedure. We first obtain an initial joint configuration at each waypoint using the analytical IK solver of [25], which provides closed-form solutions for the Franka Emika Panda. We select a collision-free configuration for both arms by discretizing the redundant wrist angle parameters. We then track the optimized task-space trajectory by integrating a differential IK controller, initialized from the analytical IK solution and regularized with a virtual spring that penalizes deviation from a nominal reference configuration. The virtual spring discourages large excursions in the null space and prevents jumps between IK solution branches.

G. SCS + EI Implementation Details

The SCS baseline [12] optimizes a minimum-time Bézier trajectory through a sequence of convex safe sets. We describe how we construct these sets from the waypoint initialization.

a) Region generation.: Since all obstacles in our task-space formulation are convex (axis-aligned box overlaps), we can solve the edge inflation exactly. For each segment m of the piecewise-linear waypoint path and each obstacle $\mathcal{O}_k$, we solve the convex program

$$\underset{x \in \mathcal{O}_k,\, z \in \mathcal{L}_m}{\text{minimize}} \quad \|x - z\|_2^2, \tag{28}$$

where $\mathcal{L}_m = \mathbf{conv}\{v_m, v_{m+1}\}$ is the m-th line segment, following [11, Program 8]. Since both $\mathcal{O}_k$ and $\mathcal{L}_m$ are convex, this is a convex program. The optimal solution $(x^\star, z^\star)$ yields a separating hyperplane via the gradient of the distance function:

$$a = \frac{x^\star - z^\star}{\|x^\star - z^\star\|_2}, \quad \mathcal{H} = \{x \mid a^\top x \le a^\top x^\star - \delta\}, \tag{29}$$

with safety margin $\delta = 10^{-3}$. We solve all obstacle-segment pairs in parallel using Drake's Solve, producing one polytope per segment by intersecting the resulting halfspaces.

b) Handling degenerate regions.: The biconvex solver [12] requires that (i) consecutive waypoints are separated by a non-negligible distance, and (ii) non-adjacent regions are disjoint (Assumption 1 in [12]). We address both by a pruning pass: segments whose shortest-path edge length is below 10^{-4} are removed, and any region that intersects a non-adjacent neighbour is iteratively removed until no such pair remains.