IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

PolyFly: Polytopic Optimal Planning for Collision-Free Cable-Suspended Aerial Payload Transportation

Mrunal Sarvaiya, *Student Member, IEEE*, Guanrui Li, *Member, IEEE*, and Giuseppe Loianno, *Member, IEEE*

***Abstract*—Aerial transportation robots using suspended cables have emerged as versatile platforms for disaster response and rescue operations. To maximize the capabilities of these systems, robots need to aggressively fly through tightly constrained environments, such as dense forests and structurally unsafe buildings, while minimizing flight time and avoiding obstacles. Existing methods geometrically over-approximate the vehicle and obstacles, leading to conservative maneuvers and increased flight times. We eliminate these restrictions by proposing PolyFly, an optimal global planner which considers a non-conservative representation for aerial transportation by modeling each physical component of the environment, and the robot (quadrotor, cable and payload), as independent polytopes. We further increase the model accuracy by incorporating the attitude of the physical components by constructing orientation-aware polytopes. The resulting optimal control problem is efficiently solved by converting the polytope constraints into smooth differentiable constraints via duality theory. We compare our method against the existing state-of-the-art approach in eight maze-like environments and show that PolyFly produces faster trajectories in each scenario. We also experimentally validate our proposed approach on a real quadrotor with a suspended payload, demonstrating the practical reliability and accuracy of our method.**

***Index Terms*—Aerial systems: applications, aerial systems: mechanics and control.**

I. Introduction

MICRO Aerial Vehicles (MAVs), particularly quadrotors equipped with cable-suspended payload systems, have gained significant attention due to their versatility and effectiveness in diverse applications such as inspection [1], search and rescue missions [2], and package transportation [3]. Their ability to deliver essential materials in regions inaccessible by ground robots makes them invaluable during disaster relief operations [4]. Furthermore, aerial transportation systems offer considerable advantages in energy efficiency over ground vehicles for last-mile logistics, significantly reducing environmental footprints in urban delivery contexts [5].

In time critical scenarios, such as rescue missions, it is crucial for aerial vehicles to optimize their flight path to reduce the mission time, especially in challenging environments like dense forests or earthquake-damaged and structurally unsafe buildings. This requires the robot planners to avoid obstacles while maneuvering through narrow gaps and around tight corners, which need non-conservative physical approximations of the system components. Existing methods often simplify these systems into spheres [6] or large prisms [7], [8] to address computational demands, trading off solution quality for computational speed. In contrast, our approach directly addresses these limitations by representing each component - quadrotor, payload, and cable - as distinct polytopes. We further enhance our planner by incorporating the quadrotor's orientation into the polytopic representation. This orientation-aware modeling enables more precise collision prediction during aggressive maneuvers in confined spaces such as the environment shown in Fig. 5. This representation provides planners with detailed geometric information, enhancing their capabilities to navigate maze-like and highly constrained environments.

Current aerial transportation planning methodologies can broadly be categorized into sampling-based and optimization-based approaches. Within optimization methods, trajectories are either parameterized as polynomials [6], [9] or discretized and solved via Model Predictive Control (MPC) frameworks [7], [10], [11], [12]. Our work builds upon MPC-based global planning techniques and specifically addresses the limitations posed by existing approaches that simplify system geometry. Methods that employ Euclidean Signed Distance Fields (ESDFs) result in discontinuous optimization constraints [13] while cylindrical obstacle representations limit the complexity of environments that can be modeled. We tackle these issues by formulating obstacle avoidance explicitly through polytopic models and represent each robot component and every obstacle as distinct polytopes. As we show in this work, this representation allows us to both incorporate smooth differentiable constraints into our global planner, and generate aggressive trajectories in tightly constrained environments.

To summarize, we present the following contributions:

- A novel non-conservative representation for aerial transportation that models the quadrotor, cable, and payload as separate components. To enhance physical accuracy,

Received 13 August 2025; accepted 28 December 2025. Date of publication 19 January 2026; date of current version 27 January 2026. This article was recommended for publication by Associate Editor T. Berrueta and Editor S.-J. Chung upon evaluation of the reviewers' comments. This work was supported in part by NSF CPS under Grant CNS-2603416, in part byNSF CAREER under Award 2546659, and in part by DARPA YFA under Grant D22AP00156-00. *(Corresponding author: Mrunal Sarvaiya.)*

Mrunal Sarvaiya and Giuseppe Loianno are with the Department of Electrical Engineering and Computer Sciences, University of California Berkeley, Berkeley, CA 94720 USA (e-mail: mrunaljsarvaiya@eecs.berkeley.edu; loiannog@eecs.berkeley.edu).

Guanrui Li is with Robotics Engineering, Worcester Polytechnic Institute, Worcester, MA 01609 USA (e-mail: gli7@wpi.edu).

https://mrunaljsarvaiya.github.io/polyfly.github.io/

This article has supplementary downloadable material available at https://doi.org/10.1109/LRA.2026.3655278, provided by the authors.

Digital Object Identifier 10.1109/LRA.2026.3655278

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

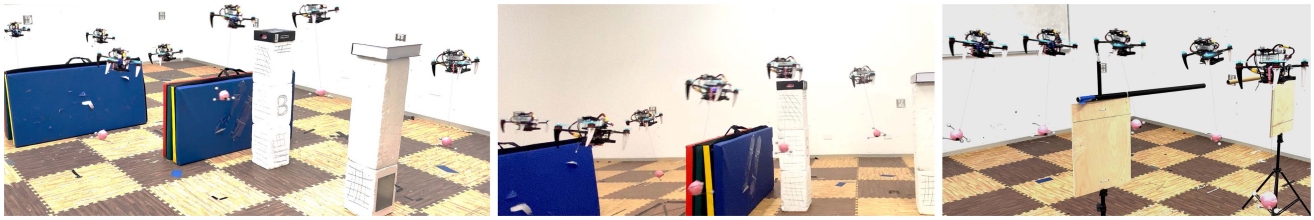


Fig. 1. A quadrotor with a suspended payload navigating through cluttered environments. The left and middle images show different perspectives of Env. 3, where the system maneuvers around multiple obstacles. The right image depicts Env. 7, where the system successfully passes through a narrow gap - just wide enough for the cable - demonstrating the planner's ability to exploit the geometry of individual components.

we incorporate the attitude of both the robot and the obstacles into the environment representation, constructing orientation-aware polytopes that more accurately capture the system's spatial footprint.
- An optimal global planning method for a robot composed of independent polytopes that produces collision-free trajectories in maze-like environments by leveraging duality theory to formulate non-linear polytopic collision constraints as smooth differentiable constraints.
- Experimental validation on hardware by performing real-world experiments using a quadrotor carrying a slung payload. We illustrate the advantages of our representation and planning method by tracking trajectories in tightly constrained environments. We also demonstrate that our approach generates faster trajectories than the existing state-of-the-art method in all ten test environments.

We release our implementation as an open-source, standalone package with minimal dependencies.

II. Related Works

In this section, we survey the main design choices in (i) how the environment and robots are spatially represented and (ii) how the states and inputs are mathematically modeled.

Environment Representations: Obstacle-aware planners typically encode the workspace either implicitly, through Euclidean Signed Distance Fields (ESDFs), or explicitly, by representing obstacles with geometric shapes. Recent aerial transportation planners [6], [9], voxelize the environment and store an ESDF, turning collision checks into fast table look-ups, but introduce non-smooth, discontinuous gradients when used in optimization-based solvers. Other works express the environment via geometric objects. Refs. [10], [11], [12] employ cylindrical and spherical objects, while [7], [8], [14] rely on polytopes. While ellipsoids and cylinders limit the range of scenes that can be modeled, polytopes significantly increase the complexity of environments that can be captured. In fact, works that use ESDFs also primarily consider environments composed of multiple cuboids [9]. Furthermore, in the context of trajectory planning, adopting polytopes to represent the environment yields distance constraints with good optimization properties. Duality theory can be used to convert non-linear polytope–to-polytope distance inequalities into smooth, differentiable constraints. Refs. [15], [16] used polytopic collision constraints for short-horizon reactive MPC frameworks but are unable to optimize over global paths due to the short MPC horizon. Refs. [7] used this approach for global planning but did not consider complex environments and cannot navigate environments through narrow gaps, eg. Fig. 1, due to their conservative representation that models the robot as a single cuboid.

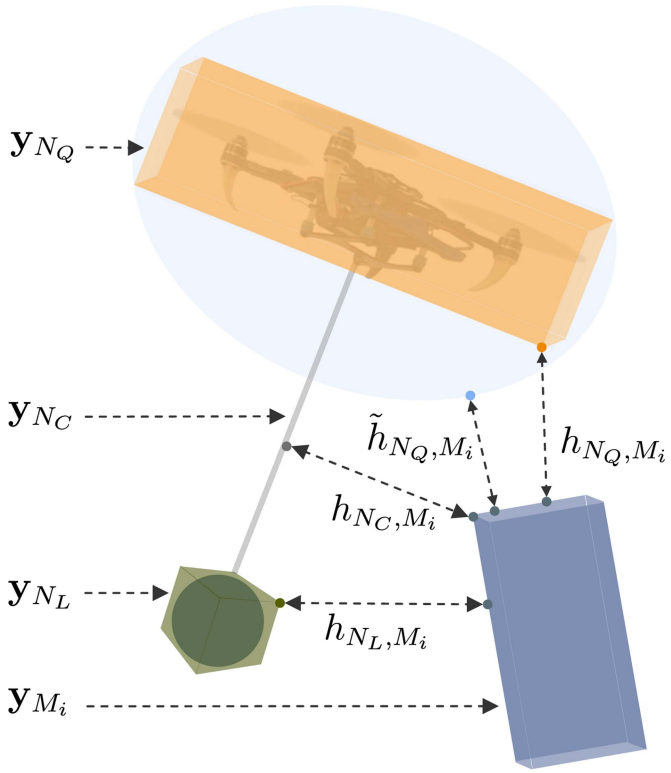


Fig. 2. Different polytopes used in PolyFly: **Quadrotor** N_Q, **Payload** N_L, **Cable** N_C, and **Obstacles** M_i. The quadrotor polytope enables accurate distance computations h_{N_Q,M_i}, whereas most methods use a sphere around the quadrotor resulting in conservative distance computations $\tilde{h}_{N_Q,M_i}$.

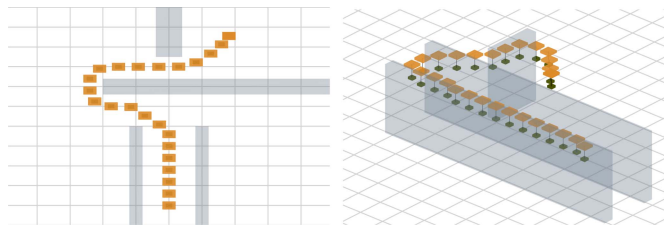


Fig. 3. A* trajectories for Env. 6 (left) and Env. 8 (right).

Spatial Robot Representation: There are two primary design decisions with respect to modeling the robot. First, how should the physical geometry of the robot be modeled? Existing methods typically use multiple spheres or ellipses to represent the quadrotor and its payload [6], [9], [10], [11], [12] or a single prism or polytope [7], [8]. These representations over-approximate at least one component (quadrotor, payload, cable

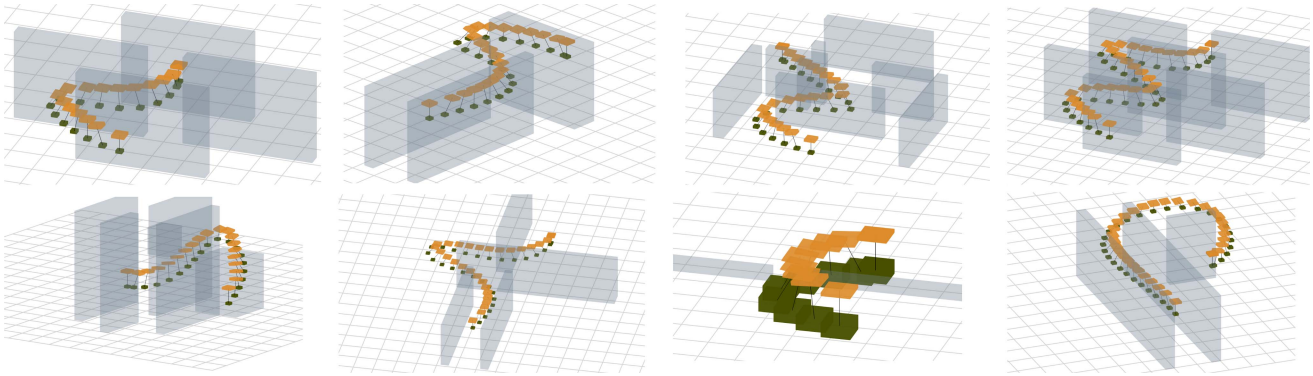


Fig. 4. Visualization of trajectories generated using PolyFly in 8 tightly cluttered environments. The quadrotor is in yellow, payload in green, cable in gray and obstacles in light blue. Envs. 1-4 are shown in the top row and 5-8 in the bottom row.

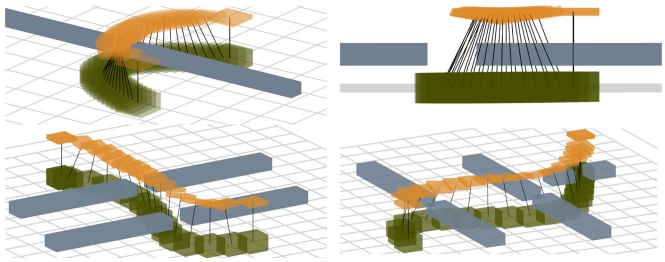
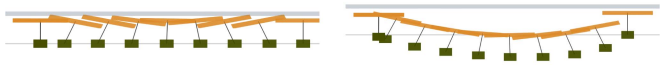


Fig. 5. PolyFly's trajectories for Env. 7 (top), 9 (bottom-left) and 10 (bottom-right) showcasing narrow-gap navigation.



(a) Robot Collides (b) Robot Avoids Collision

Fig. 6. Left: An orientation-agnostic planner produces a trajectory that collides with the ceiling. Right: By accounting for the robot's orientation, the optimized path descends to prevent collisions while accelerating.

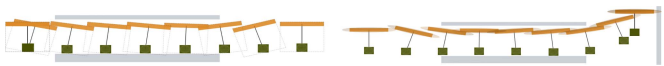


Fig. 7. Optimized trajectories by PolyFly for the ceiling-floor (left) and ceiling-wall (right) environments. The dashed lines are the polytopic approximation by PolySingle and the semi-transparent ellipse is the ellipsoidal approximation.

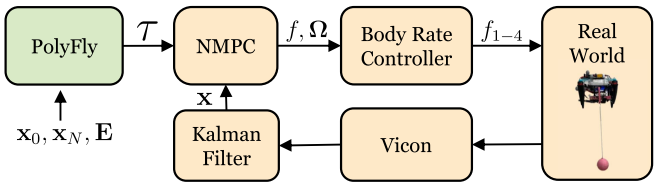


Fig. 8. System architecture for real-world deployment.

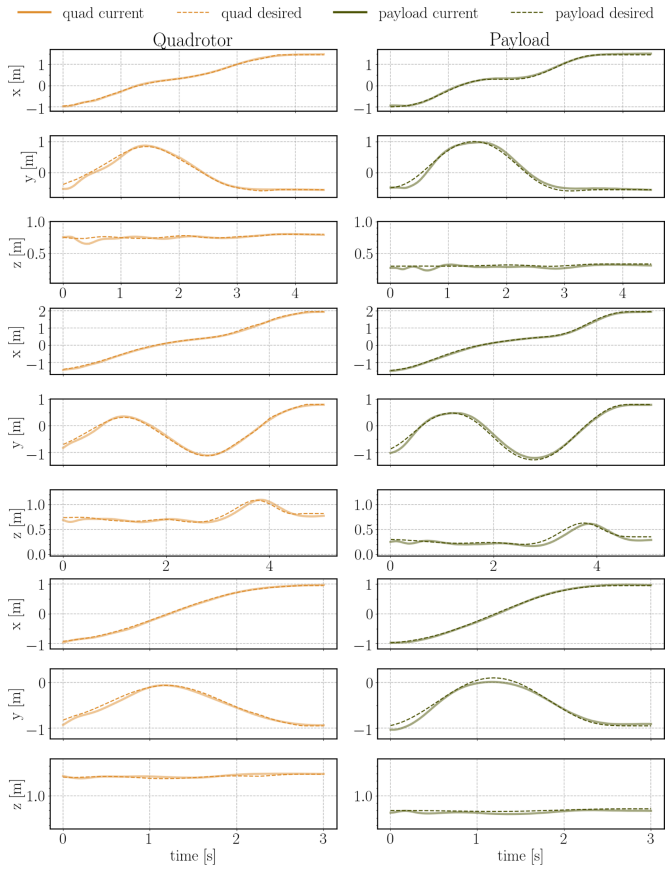


Fig. 9. Desired vs. Measured positions of the quadrotor and payload during real-world experiments for Env. 1, 3, and 7.

or environment obstacles), therefore limiting their applicability in highly constrained or cluttered environments. Spheres over-approximate the quadrotor, cable, and obstacles while ellipses over-approximate the quadrotors and obstacles used in this work. For example, ellipsoidal representations fail in the ceiling-wall environments in Fig. 7, while single polytope representations [7] fails in the ceiling-floor and narrow gap environments in Figs. 5 and 7. In contrast, our method addresses these limitations by leveraging orientation-aware polytopes for each robot component - quadrotor, cable, and payload. Combined with polytopic representation of the environments, we enable non-conservative distance evaluation, as shown in Fig. 2, facilitating aerial transportation in maze-like environments and narrow passages that are slightly larger than the width of the cable.

State and Input Modeling: The second design decision is how states and inputs are mathematically represented. A popular strategy is to express the states and inputs as polynomials to facilitate trajectory optimization [6], [9]. This is often paired

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

TABLE I
NOTATION TABLE

$\mathcal{I}$, $\mathcal{B}$	inertial, and robot frame
$m_L, m_Q \in \mathbb{R}$	mass of payload and robot
$\mathbf{x}_L, \mathbf{x}_Q \in \mathbb{R}^3$	position of payload and robot in $\mathcal{I}$
$\dot{\mathbf{x}}_L, \ddot{\mathbf{x}}_L, \dddot{\mathbf{x}}_L \in \mathbb{R}^3$	payload velocity, acceleration, jerk in $\mathcal{I}$
$\dot{\mathbf{x}}_Q, \ddot{\mathbf{x}}_Q \in \mathbb{R}^3$	robot velocity, acceleration in $\mathcal{I}$
$\boldsymbol{R}_Q \in SO(3)$	robot orientation with respect to $\mathcal{I}$
$\boldsymbol{\Omega} \in \mathbb{R}^3$	robot angular velocity in $\mathcal{B}$
$f \in \mathbb{R}$, $\mathbf{M} \in \mathbb{R}^3$	collective thrust and moment on robot in $\mathcal{B}$
$J \in \mathbb{R}^{3\times3}$	moment of inertia of robot in $\mathcal{B}$
$\boldsymbol{\xi} \in S^2$	unit vector from robot to payload in $\mathcal{I}$
$f_i, l, g \in \mathbb{R}$	i^{th} motor thrust, cable length, gravity
$N_O \in \mathbb{Z}$	total number of obstacles
M_i, N_j	i^{th} obstacle and j^{th} robot polytope,

with MINCO [17] to reformulate the problem into an unconstrained optimization to boost computational efficiency. While this approach can be used to achieve real-time capabilities, our experiments in Section V-Q1 indicate that it may compromise both solution quality and convergence. In contrast, similar to [7], [10], [11], [12], PolyFly operates directly in the native state space. Unlike polynomial parameterizations, the native state space approach places no restrictions on the shape of the state or control trajectories, making them inherently more expressive. The method presented in [6], [9], serves as our primary baseline, employs an ESDF to represent the environment, approximates the quadrotor and payload with spheres, and parameterizes the states and inputs via polynomials. Ref. [9] reports an ablation study comparing the computational times and success rates of their method with IPOPT-MP, a state-based global planning approach. However, they omit the comparison of the optimized trajectory times of their method with IPOPT-MP, and do not address navigation close to obstacles in maze-like environments. We demonstrate that by using the native state-based modeling and a non-conservative spatial workspace representation via polytopes, PolyFly outperforms this method in all test environments.

III. System Dynamics

In this section, we summarize the non-linear dynamics of a quadrotor carrying a suspended payload. We follow the derivation presented in [3] and assume that the cable remains taut. Applying the Lagrange–d'Alembert principle, we obtain

$$\frac{d\mathbf{x}_L}{dt} = \dot{\mathbf{x}}_L, \ \frac{d\mathbf{x}_Q}{dt} = \dot{\mathbf{x}}_Q, \ \dot{\mathbf{R}}_Q = \boldsymbol{R}_Q\hat{\boldsymbol{\Omega}}, \tag{1}$$

$$(m+m_L)\left(\ddot{\mathbf{x}}_L + \mathbf{g}\right) = \left(\boldsymbol{\xi}\cdot f\boldsymbol{R}_Q\mathbf{e}_3 - ml\left(\dot{\boldsymbol{\xi}}\cdot\dot{\boldsymbol{\xi}}\right)\right)\boldsymbol{\xi}, \tag{2}$$

$$ml\left(\ddot{\boldsymbol{\xi}} + \left(\dot{\boldsymbol{\xi}}\cdot\dot{\boldsymbol{\xi}}\right)\boldsymbol{\xi}\right) = \boldsymbol{\xi}\times\left(\boldsymbol{\xi}\times f\boldsymbol{R}_Q\mathbf{e}_3\right), \tag{3}$$

$$\mathbf{M} = \mathbf{J}\dot{\boldsymbol{\Omega}} + \boldsymbol{\Omega}\times\mathbf{J}\boldsymbol{\Omega}, \tag{4}$$

$$\left[f, \mathbf{M}\right]^\top = \mathbf{D}\left[f_1, f_2, f_3, f_4\right]^\top. \tag{5}$$

where $\mathbf{g} = g\mathbf{e}_3$, $g = 9.81\,\text{m/s}^2$, $\mathbf{e}_3 = [0\ 0\ 1]^\top$, and $\hat{\boldsymbol{\Omega}}$ is the skew-symmetric matrix of the quadrotor angular velocity $\boldsymbol{\Omega}$. Table I defines the remaining variables. Qualitatively, (2) describes the relationship between the payload acceleration and collective motor thrust, while (3) shows that cable's angular acceleration depends on the collective thrust and the quadrotor's orientation. The matrix $\mathbf{D}$ provides a linear mapping between collective thrust-moments and individual motor thrusts. It is defined in terms of the robot's physical properties and propellers aerodynamics characteristics. Readers can refer to [18] for additional information.

By leveraging the differential flatness property of the system [3], we can model the state dynamics described in (1)–(5) in terms of the flat outputs $\{\mathbf{x}_L, \psi\}$, where ψ is the quadrotor's yaw angle. In our method, we set $\psi = 0$ and assume negligible cable angular accelerations. This allows us to express the system through the state $\mathbf{x} = \left[\mathbf{x}_L{}^\top, \dot{\mathbf{x}}_L{}^\top, \ddot{\mathbf{x}}_L{}^\top\right]^\top \in \mathbb{R}^9$ and input $\mathbf{u} = \left[\dddot{\mathbf{x}}_L\right] \in \mathbb{R}^3$. The induced system dynamics are therefore modeled as a third-order system. Given $\mathbf{x}$, $\mathbf{u}$, the robot's physical parameters, and assuming that cable angular accelerations are zero at the states optimized by the planner, we can use the non-linear differential flatness mapping [3] to calculate the robot position $\mathbf{x}_Q$, robot velocity $\dot{\mathbf{x}}_Q$, and orientation $\boldsymbol{R}_Q$. This allows our trajectory generator to plan in payload space and add constraints to the quadrotor's states. Since our planner operates in a discrete space, we convert the continuous time representation above to its discrete form. We obtain the Runge-Kutta approximation $\mathbf{x}(k+1) = F(\mathbf{x}(k), \mathbf{u}(k), \Delta t_k)$, where F is the discretized dynamics function and Δt_k the time step duration at stage k.

IV. Planning

A. Preliminaries

We develop a global planner that computes a collision-free trajectory $\boldsymbol{\tau}$ for a quadrotor with a suspended payload where

$$\boldsymbol{\tau} = \left[\boldsymbol{\tau}_0, \ldots, \boldsymbol{\tau}_N\right], \boldsymbol{\tau}_k = \left[\mathbf{x}_L(k),\ \dot{\mathbf{x}}_L(k),\ \ddot{\mathbf{x}}_L(k)\right]^\top, \tag{6}$$

where $k \in (0, N)$. The robot starts at state $\mathbf{x}_O$ and must navigate to its goal state $\mathbf{x}_N$. The environment is described by the set $\mathbf{E}$, which contains N_O polytopic obstacles M_i for $i \in (0, N_O)$. The aerial system is modeled using three separate polytopes, N_j for $j \in \{Q, C, L\}$ that represent the quadrotor, cable, and load polytopes respectively, as shown in Fig. 2.

B. Polytopic Approximation

Let us consider two polytopes $\mathbf{M_i}$ and $\mathbf{N_j}$, as defined in Table I. Let $\mathbf{y}_{M_i}$ be any point within the polytope $\mathbf{M_i}$ and $\mathbf{y}_{N_j}$ be any point within the polytope $\mathbf{N_j}$. $\mathbf{A_{M_i}}, \mathbf{B_{M_i}}, \mathbf{A_{N_j}}$ and $\mathbf{B_{N_j}}$ are constants that define their geometric shape. These polytopes can be represented by

$$\mathbf{A}_{M_i}\mathbf{y}_{M_i} \le \mathbf{B}_{M_i},\ \mathbf{A}_{N_j}\mathbf{y}_{N_j} \le \mathbf{B}_{N_j}. \tag{7}$$

While not strictly necessary, we simplify computation by representing one polytope's points in the other polytope's frame. We represent the obstacle points $\mathbf{y}_{M_i}$ using the relative position vector $\mathbf{y}_{M_i}^{N_j}$, which gives

$$\mathbf{y}_{M_i} = \boldsymbol{R}_{N_j}\mathbf{y}_{M_i}^{N_j} + \mathbf{O}_{N_j}, \tag{8}$$

where $\boldsymbol{R}_{N_j}$ is N_j's rotation matrix, and $\mathbf{O}_{N_j}$ is its origin. Substituting this into the polytopic form expressed by (7), we can represent polytope M_i as

$$(\mathbf{A}_{M_i}\boldsymbol{R}_{N_j})\mathbf{y}_{M_i}^{N_j} \le \mathbf{B_{M_i}} - \mathbf{A}_{M_i}\mathbf{O}_{N_j}. \tag{9}$$

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Since $\boldsymbol{R}_{N_j}$ and $\mathbf{O}_{N_j}$ are functions of the chosen state vector $\mathbf{x}$, M_i is represented by

$$\mathbf{A}_{M_i}^{N_j}(\mathbf{x})\mathbf{y}_{M_i}^{N_j} \leq \mathbf{B}_{M_i}^{N_j}(\mathbf{x}),$$
$$\mathbf{A}_{M_i}^{N_j}(\mathbf{x}) = \mathbf{A}_{M_i}\boldsymbol{R}_{N_j}(\mathbf{x}), \mathbf{B}_{M_i}^{N_j}(\mathbf{x}) = \mathbf{B}_{\mathbf{M_i}} - \mathbf{A}_{M_i}\mathbf{O}_{N_j}(\mathbf{x}). \tag{10}$$

C. Distance Metric Between Polytopes

Our optimization problem seeks to lower bound h_{M_i,N_j}, the distance between polytopes M_i and N_j, with a positive margin β. Specifically,

$$h_{M_i,N_j}(\mathbf{x}) \geq \beta,\ h_{M_i,N_j}(\mathbf{x}) := d(M_i, N_j), \tag{11}$$

where d is the distance function described in [19, Eq. (8a)]. The constraint defined by (11) is non-differentiable and difficult to integrate into optimization problems when the obstacles and robots are represented as convex polytopes [19]. To handle this complexity, we follow the approach in [19, Proposition 1]. and reformulate (11) using duality theory into the following set of smooth non-linear differentiable constraints. The equivalent dual problem is

$$\begin{aligned} g_{M_i,N_j}(\mathbf{x}) = \max_{\boldsymbol{\lambda}_{N_j},\boldsymbol{\lambda}_{M_i}} & -\boldsymbol{\lambda}_{N_j}^\top \mathbf{B}_{N_j} - \boldsymbol{\lambda}_{M_i}^\top \mathbf{B}_{M_i}^{N_j}(\mathbf{x}) \\ \text{s.t.}\ & \boldsymbol{\lambda}_{N_j}^\top \mathbf{A}_{N_j} + \boldsymbol{\lambda}_{M_i}^\top \mathbf{A}_{M_i}^{N_j}(\mathbf{x}) = 0 \\ & \boldsymbol{\lambda}_{M_i} \geq 0, \quad \boldsymbol{\lambda}_{N_j} \geq 0, \quad \|\boldsymbol{\lambda}_{M_i} * \mathbf{A}_{M_i}\|_2 \leq 1, \end{aligned} \tag{12}$$

where $\boldsymbol{\lambda}_{N_j} \in \mathbb{R}^{n_j}$ and $\boldsymbol{\lambda}_{M_i} \in \mathbb{R}^{m_i}$ are the dual variables associated with the obstacle avoidance problem. For our setup, $m_i = n_j = 6$. Results in [19], [20] show that Strong Duality holds for the dual problem defined in (12), which gives us the lower bound of the primal cost function $h_{M_i,N_j}(\mathbf{x})$

$$\begin{aligned} \bar{g}_{M,N}(\mathbf{x}, \boldsymbol{\lambda}_{M_i}, \boldsymbol{\lambda}_{N_j}) := & -\boldsymbol{\lambda}_{N_j}^\top \mathbf{B}_{N_j} - \boldsymbol{\lambda}_{M_i}^\top \mathbf{B}_{M_i}^{N_j}(\mathbf{x}) \\ & \leq h_{M_i,N_j}(\mathbf{x}). \end{aligned} \tag{13}$$

Therefore, to enforce a minimum distance of β between M_i and N_j, we constrain the lower bound $\bar{g}_{M,N} \geq \beta$, giving us a smooth non-conservative reformulation of (11)

$$\begin{aligned} & -\boldsymbol{\lambda}_{N_j}^\top \mathbf{B}_{N_j} - \boldsymbol{\lambda}_{M_i}^\top \mathbf{B}_{M_i}^{N_j}(\mathbf{x}) \geq \beta, \\ & \boldsymbol{\lambda}_{N_j}^\top \mathbf{A}_{N_j} + \boldsymbol{\lambda}_{M_i}^\top \mathbf{A}_{M_i}^{N_j}(\mathbf{x}) = 0, \\ & \boldsymbol{\lambda}_{M_i} \geq 0, \quad \boldsymbol{\lambda}_{N_j} \geq 0, \quad \|\boldsymbol{\lambda}_{M_i} * \mathbf{A}_{M_i}\|_2 \leq 1. \end{aligned} \tag{14}$$

We add the set of constraints defined in (14) for collisions between each (M_i, N_j) pair, where $i \in (0, N_O)$. For notational convenience, we combine the dual variables into a vector $\boldsymbol{\Lambda} \in \mathbb{R}^Q$, where $Q = N \times N_O \times N_R \times (m_i + n_j)$, and $N_R = 3$ is the number of robot components.

D. Optimization Problem Formulation

Our planner solves the following optimization problem

$$\begin{aligned} \min_{\mathbf{x}(k),\mathbf{u}(k),\Delta t_k,\boldsymbol{\Lambda}} & \sum_{k=0}^{N-1} L(\mathbf{x}(k), \mathbf{u}(k), \Delta t_k) \\ \text{subject to } & \mathbf{x}(k+1) = F(\mathbf{x}(k), \mathbf{u}(k), \Delta t_k, \\ & \mathbf{x}(0) = \mathbf{x}_0,\ \mathbf{x}(N) = \mathbf{x}_N, \\ & H_l(\mathbf{x}(k), \mathbf{u}(k), \boldsymbol{\Lambda}) \leq 0,\ \Delta t_{min} \leq \Delta t_k \leq \Delta t_{max}, \end{aligned} \tag{15}$$

where $\forall k \in (0, N-1)$, $\forall l \in (0, N_h)$, N is the horizon length, Δt_k is the time duration for stage k, and x_0 and x_N are the initial and terminal states respectively. Δt_{min} and Δt_{max} are the min. and max. time durations set by the user and are discussed in Section IV-D1. H_l comprises the N_h inequality constraints from the collision-avoidance formulation in (14) together with the state and input bounds specified in Section IV-D7. Finally, $L = L_t + L_u + L_g + L_{td}$ is the loss function defined in the subsequent subsections.

We now describe the optimization constraints, costs and initialization methods. The variables Δt_{min}, Δt_{max}, α_{to}, α_u, α_g, α_{td} and N, are user defined constants whose values are defined in Section V.

1) Time Minimization: We incentivize the planner to minimize the total trajectory time by adding a large cost to each time step duration. Specifically,

$$L_t = T\frac{\alpha_{to}}{N}, T = \sum_{k=0}^{N-1} \Delta t_k, \Delta t_{min} \leq \Delta t_k \leq \Delta t_{max}, \tag{16}$$

where $\Delta t_{min}, \Delta t_{max}, \alpha_{to} > 0$.

2) Input Smoothness Regularization: We add a small cost that penalizes large changes between consecutive inputs, thereby regularizing their rate of change and improving the smoothness of the resulting robot states.

$$L_u = \frac{\alpha_u}{N}\sum_{k=1}^{N} \|\mathbf{u}_i - \mathbf{u}_{i-1}\|^2, \quad \alpha_u > 0. \tag{17}$$

3) Regularizing Proximity to the Initial Guess: Even though we use a simple A* initialization, we expect the optimal trajectory's positions to lie in the vicinity of the initialized positions. We achieve this by adding a small penalty to deviations of the optimized positions from the initial guess. The study supporting the benefits of this term is discussed in Section V-Q4.

$$L_g = \frac{\alpha_g}{N-1}\sum_{k=1}^{N-1} \|\mathbf{x}(k) - \mathbf{x}_g(k)\|^2, \tag{18}$$

where $\mathbf{x}_g(k)$ is the initial guess at stage index k and $\alpha_g > 0$.

4) Regularizing Subsequent Time Step Durations: We add a small cost to the difference between subsequent time step durations to help our solver converge. Specifically, we add

$$L_{td} = \frac{\alpha_{td}}{N}\sum_{k=0}^{N-1} \|\Delta t_{k+1} - \Delta t_k\|^2, \quad \alpha_{td} > 0. \tag{19}$$

5) Solver Initialization: Since the optimization problem combines non-linear dynamics with non-convex collision constraints, the resulting solution is sensitive to its initial guess. Whereas many prior planners [6], [9] adopt a kino-dynamic initialization strategy, our approach begins from a simple A* path that ignores the non-linear dynamics and assumes the cable is always aligned with the global z-axis, i.e., $\ddot{\mathbf{x}}_L = 0$. Fig. 3 visualizes this initialization for two representative environments. We then use a lightweight initialization method for the load velocities using the equation

$$\dot{\mathbf{x}}_{L,k} = \frac{\mathbf{x}_{L,k+1} - \mathbf{x}_{L,k}}{\Delta t_{max} - \Delta t_{min}}, \tag{20}$$

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

where $0 < \Delta t_{min} < \Delta t_{max}$. Even with this relatively simple initialization method, our planner produces high quality trajectories that are consistently better than the baseline.

6) Collision Constraints: We account for the collision between the quadrotor, cable, and payload polytopes against each obstacle. For each time stage k and each polytope pair, we add the constraints derived in Section IV-C. As an example, to handle collisions between the payload polytope N_L and an obstacle M_i, we add the following constraints for all stages k and obstacles i

$$\begin{aligned}\forall k,\ \forall i \quad & -\boldsymbol{\lambda}_{N_L,k}^{\top}\mathbf{B}_{N_L} - \boldsymbol{\lambda}_{M_i,k}^{\top}\mathbf{B}_{M_i}^{N_L}(\mathbf{x}_k) \geq \beta, \\ & \boldsymbol{\lambda}_{N_L,k}^{\top}\mathbf{A}_{N_L} + \boldsymbol{\lambda}_{M_i,k}^{\top}\mathbf{A}_{M_i}^{N_L}(\mathbf{x}_k) = 0, \\ & \boldsymbol{\lambda}_{M_i,k} \geq 0, \quad \boldsymbol{\lambda}_{N_L,k} \geq 0, \quad \left\|\boldsymbol{\lambda}_{M_i,k} * \mathbf{A}_{M_i}\right\|_2 \leq 1. \end{aligned} \tag{21}$$

7) State and Input Constraints: The system's initial and final positions are specified based on the environment and are enforced via equality constraints. We also constrain the start and end velocities, accelerations and jerks to zero to account for hardware limitations. Specifically,

$$\begin{aligned} & \mathbf{x}(0) = \mathbf{x}_0,\ \mathbf{x}(N) = \mathbf{x}_N,\ \mathbf{u}(0) = \mathbf{u}(N) = 0, \\ & \dot{\mathbf{x}}_L(0) = \dot{\mathbf{x}}_L(N) = 0,\ \ddot{\mathbf{x}}_L(0) = \ddot{\mathbf{x}}_L(N) = 0, \\ & \dot{\mathbf{x}}_Q(0) = \dot{\mathbf{x}}_Q(N) = 0,\ \ddot{\mathbf{x}}_Q(0) = \ddot{\mathbf{x}}_Q(N) = 0. \end{aligned} \tag{22}$$

Finally, we bound the state and inputs based on the robot's limitations and the environment dimensions.

$$\mathbf{x}_l \leq \mathbf{x} \leq \mathbf{x}_h,\ \mathbf{u}_l \leq \mathbf{u} \leq \mathbf{u}_h, \tag{23}$$

where $\mathbf{x}_l, \mathbf{u}_l$ and $\mathbf{x}_h, \mathbf{u}_h$ are the lower and upper bounds.

8) Incorporating the Quadrotor's Rotation Into the Polytopic Constraints: To improve the accuracy of the polytopic representation, we rotate the quadrotor polytope by the quadrotor's rotation. As discussed in Section III, we can derive the robot's rotation $\boldsymbol{R}_Q$ using $\mathbf{x}$ and $\mathbf{u}$, and use it to compute the $\mathbf{A}_{M_i}^{N_Q}$ matrix in (10).

V. Experimental Results

We conduct various experiments to evaluate the capabilities of this method. We seek to answer the following:

- Q1 Can this planning method produce trajectories that navigate tightly around corners of polytopic obstacles in cluttered maze environments?
- Q2 What are the benefits of modeling each component of the robot separately?
- Q3 What are the benefits of adding the quadrotor's orientation into the polytopic representation?
- Q4 Is adding a cost that regularizes the proximity to the initial guess important?
- Q5 Does our velocity initialization method help the solver converge?
- Q6 Can we run these aggressive trajectories on a real quadrotor with a suspended payload?

For all experiments, we set $\Delta t_{min} = 0.01s$, $\Delta t_{max} = 0.20s$, $\alpha_u = 5.0$, $\alpha_g = 5.0$, $\alpha_{td} = 600.0$ and $N = 100$. As described in Section V-Q1, α_{to} is tuned to maximize performance in each environment.

Q1 Navigating Cluttered Environments

We create 10 maze environments to evaluate the performance of our planning method. The optimization problem is constructed in CasADi [21] and solved with IPOPT [22].

We select AutoTrans, the MINCO approach [17] adopted in [6], [9] as our primary baseline, which is an optimization-based trajectory generator that parametrizes trajectories as a sequence of 7^{th} order polynomials, initialized with kinodynamic hybrid A^*. They model the quadrotor and payload as independent spheres, the cable as a set of discrete points, and encode the environment using an ESDF. To solve the resulting problem, they reformulate the problem into an unconstrained one by applying L_1 penalties on the constraints.

To ensure a fair comparison, we adjusted AutoTrans's safety margins, ESDF map resolution, kinodynamic search resolution, and quadrotor parameters to be consistent with our environment setup and removed AutoTrans' hardware limitations constraints on the quadrotor thrust and tilt angle. Furthermore, for each environment, we tune AutoTrans' and PolyFly's time minimization weights and report the mean results based on the planning output over 10 trials in Table II. PolyFly's trajectories are shown in Figs. 4 and 5.

We include an additional baseline, PolySingle, which mirrors our method but adopts the modeling of [7], where a single polytope represents the entire robot. AutoTrans fails to produce a collision-free trajectory in $5/10$ environments, as indicated by a $\times$. We also observed that in some cases, AutoTrans generated shorter paths in terms of distance, yet required more time to traverse them. This indicates that our method is able to better optimize for total flight time, even if that entails following a slightly longer path at higher speeds. Compared to PolySingle, our method successfully navigates the narrow gaps in Env. 7, 9 and 10, whereas PolySingle fails due to its modeling over-approximation. PolyFly's boost in performance compared to the AutoTrans may stem from the geometric modeling of the workspace and the use of the dual formulation, which produces smooth, non-conservative collision constraints [19]. These properties enable it to plan paths around obstacles while maintaining speed.

Q2 Independent Polytopic Models for Robot Components

Modeling each robot component as a separate polytope allows the trajectory generator to accurately represent the system's geometry. This fidelity is important when producing agile trajectories to navigate through narrow gaps and in tightly constrained environments where the robot must cut corners aggressively while still avoiding obstacles.

The advantage is evident in Env. 7, 9 and 10, where the robot must pass through a narrow opening wider than the cable but thinner than the payload and quadrotor. Both the baselines fail, but our method successfully produces trajectories that thread the cable through the gaps, allowing the robot to safely navigate the environment. The real world experiment snapshots are shown in Fig. 1.

PolySingle fails in these narrow-gap environments due to its single-polytope representation that over-approximates the robot's geometry. On the other hand, AutoTrans models the cable as a discrete set of points and performs collision checks on each point, which should geometrically allow the cable to pass through the narrow gaps. Even though a solution exists, AutoTrans fails to produce a trajectory, even after extensive tuning of

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

TABLE II

POLYFLY VS. POLYSINGLE VS AUTOTRANS. STATISTICS ARE CALCULATED AFTER RUNNING 10 TRIALS IN EACH ENVIRONMENT. T IS THE MEAN TRAJECTORY DURATION IN SECONDS, PATH IS MEAN PATH LENGTH IN METERS, COMP. IS THE MEAN OPTIMIZATION WALL TIME IN SECONDS, AND SR IS THE SUCCESS RATIO. % $T\uparrow$ W/ A DENOTES THE % BY WHICH POLYFLY'S TRAJECTORY DURATION T IS SHORTER THAN METHOD A'S. × MARKS A FAILURE.

	PolyFly					PolySingle (PS)					AutoTrans (AT) [6]				% $T\uparrow$	% $T\uparrow$
Env.	T	**Path**	**Comp.**	**SR**	α_{to}	T	**Path**	**Comp.**	**SR**	α_{to}	T	**Path**	**Comp.**	**SR**	**w/ AT**	**w/ PS**
1	**3.90**	4.51	4.83	1.0	5000	3.96	4.50	5.26	1.0	5000	4.10	4.45	0.06	1.0	4.9	1.5
2	**4.60**	6.24	3.97	1.0	6000	4.68	6.34	2.99	1.0	6000	×	×	×	0.0	∞	1.7
3	**4.92**	6.24	7.11	1.0	5000	5.13	6.35	7.51	1.0	5000	5.68	6.47	0.11	1.0	13.4	4.1
4	**7.09**	8.82	6.80	1.0	6000	7.22	9.16	5.16	1.0	6000	7.47	8.74	0.89	0.9	5.1	1.8
5	**4.05**	5.82	11.14	1.0	5000	4.12	5.82	11.20	1.0	5000	4.30	4.92	1.63	0.6	5.8	1.7
6	**5.31**	7.95	4.90	1.0	6000	5.48	8.08	7.10	1.0	6000	6.43	8.02	49.44	0.0	17.4	3.1
7	**2.86**	3.07	3.12	1.0	800	×	×	×	0.0	800	×	×	×	0.0	∞	∞
8	**4.92**	9.40	3.31	1.0	10000	4.96	9.51	3.21	1.0	10000	×	×	×	0.0	∞	0.8
9	**3.72**	5.89	3.34	1.0	2000	×	×	×	0.0	2000	×	×	×	0.0	∞	∞
10	**4.46**	6.89	9.92	1.0	2000	×	×	×	0.0	2000	×	×	×	0.0	∞	∞

its optimization weights. Therefore, rather than the underlying geometry, the cause of the failures in AutoTrans may stem from other elements of the pipeline. First, the MINCO polynomial state parameterization, cannot, in general, represent optimal solutions [17] which limits its performance in tightly cluttered scenarios. And second, distance-field-based collision checking methods are not well-suited for applications that require accurate modeling of geometry [23]. These methods compute the gradients to prevent collisions per point of the robot/obstacle, rather than the exact gradient needed to move the polytope out of collision. These two gradients do not always agree, and can cause issues when the robot travels in close proximity to obstacles.

Q3 Incorporating the Quadrotor's Orientation Into Polytopic Representation

Accurately capturing the quadrotor's orientation further increases the fidelity of the polytopic model and therefore navigation reliability. This detail becomes critical when the vehicle flies close to a ceiling or floor, particularly for a platform with long rotor arms that form a thin yet wide rectangular footprint. Since high-acceleration maneuvers induce large orientation changes, aggressive trajectories close to surfaces can cause rotor collisions with the environment. As shown in Figs. 2 and 7, most planners over-approximate the vehicle with a sphere or ellipsoid. By employing a tight polytope that accounts for the robot's orientation, we eliminate this conservatism and ensure collision free navigation in near-ceiling flight.

We demonstrate these benefits in Figs. 6 and 7. Fig. 6 illustrates how incorporating the orientation into the planner prevents collisions. In Fig. 6(a), the quadrotor strikes the ceiling during a high-acceleration maneuver because the planner ignores the orientation change that accompanies acceleration. In contrast, Fig. 6(a) shows how the orientation-aware planner descends during the acceleration phase to avoid a collision. The ceiling-floor scenario in Fig. 7 shows the PolyFly trajectory and overlays the PolySingle polytope to illustrate the over-approximation introduced by [7]. PolySingle fails in these scenarios because [7] uses a common orientation for all components due to its single polytope representation of the entire system, which prevents navigation in such tightly constrained environments.

Q4 Regularizing Proximity to Initial Guess

We perform an ablation study to determine the importance of regularizing the proximity to the initial guess. We run the optimizer with 3 different α_g values. The results are shown in Table III. PolyFly fails in $5/10$ environments if $\alpha_g = 0$. Trajectories generated with $\alpha_g = 1$ are on average slower than those generated with $\alpha_g = 5$. This highlights the value of this regularization term. We believe this aids convergence by incentivizing the optimizer to remain in the vicinity of the initial guess, preventing it from entering regions of local minima that are difficult to escape from.

TABLE III

T FOR DIFFERENT VALUES OF α_g. % $T\uparrow$ **(5W/ 1)** INDICATES IMPROVEMENT OF $\alpha_g = 5$ OVER $\alpha_g = 1$.

Env.	$\alpha_g = 0$	$\alpha_g = 1$	$\alpha_g = 5$	% $T\uparrow$ **(5 w/ 1)**
1	×	3.92	3.92	0.0%
2	×	5.02	4.61	8.2%
3	4.94	4.96	4.97	-0.2%
4	×	7.13	7.15	0.3%
5	4.20	4.03	4.05	-0.5%
6	×	5.48	5.37	2.0%
7	3.15	4.59	3.16	31.2%
8	×	4.97	4.99	-0.4%
9	3.93	3.93	3.94	-0.3%
10	4.67	4.72	4.69	0.6%

Q5 Payload Velocity Initialization

To evaluate the effectiveness of our solver initialization strategy, we conduct an ablation study that compares the payload-velocity initialization of (20) against a variant where velocities are initialized to zero. Table IV summarizes the results. The solver fails in $2/10$ environments without the velocity initialization strategy. For all other scenarios, our strategy reduces the solver iterations by an average of 59%.

Q6 Real World Experiments

We run real world experiments with Env. 1, 3 and 7 to validate that our method produces trajectories that can be deployed on real robots. Our system architecture is shown in Fig. 8. For real-world experiments only, we use the Non-Linear Model Predictive Control (NMPC) setup from [24] as our controller. The NMPC uses the dynamics (1)–(5) with inputs f_{1-4}. After generating a trajectory with PolyFly, we send the desired $\boldsymbol{\tau}$ to the NMPC at 200 Hz. As in [24], the predicted collective thrust and bodyrates from the MPC's second stage are sent to

TABLE IV
PERFORMANCE WITH AND WITHOUT $\dot{\mathbf{x}}_L$ INITIALIZATION

	With Velocity Initialization		W/o Velocity Initialization	
Env.	T (s)	**Iterations**	T (s)	**Iterations**
1	3.90	201↓	3.98	745
2	4.60	223↓	4.61	640
3	4.92	133↓	4.92	677
4	7.09	185↓	×	×
5	4.05	353↓	×	×
6	5.31	206↓	5.31	559
7	2.86	234↓	2.87	309
8	4.92	161↓	4.90	617
9	3.72	112↓	3.72	206
10	4.46	306↓	4.43	652

TABLE V
TRACKING ERROR STATISTICS (METERS)

Axis	**Env. 1**		**Env. 3**		**Env. 7**	
	Mean	**Std**	**Mean**	**Std**	**Mean**	**Std**
x	0.027	0.017	0.029	0.016	0.021	0.094
y	0.058	0.047	0.059	0.048	0.052	0.036
z	0.024	0.012	0.039	0.029	0.029	0.087

the low-level controller. The robot's states are measured using a motion capture system and are upsampled using Kalman Filters. Snapshots of our experiments are shown in Fig. 1. The position errors for the quadrotor and payload are shown in Fig. 9 and Table V. Our results indicate that PolyFly produces dynamically feasible trajectories that can be tracked by an onboard controller with low errors. The supplementary video contains real-world footage that highlights the agile trajectories generated by our method.

VI. CONCLUSION

This letter presented PolyFly, a global planner for a single robot aerial transportation system that models the robot and environment using a set of polytopes and enables the generation of collision-free trajectories in cluttered scenarios. Our proposed method yielded consistently faster trajectories than a popular state-of-the-art baseline in ten different maze-like environments. Experimental results confirm the trajectories' dynamic feasibility and the applicability of the proposed solution in real-world settings. One limitation is that PolyFly does not yet run at real-time frequencies or account for onboard obstacle sensing. Future work will address this by efficiently updating trajectories using onboard local obstacle information and leveraging advances in polytopic collision-avoidance checks [25] to reduce computation.

REFERENCES

[1] M. A. Trujillo, J. R. Martinez-de Dios, C. Martin, A. Viguria, and A. Ollero, "Novel aerial manipulator for accurate and robust industrial NDT contact inspection: A new tool for the oil and gas inspection industry," *Sensors*, vol. 19, no. 6, 2019, Art. no. 1305.

[2] G. Loianno and V. Kumar, "Cooperative transportation using small quadrotors using monocular vision and inertial sensing," *IEEE Robot. Automat. Lett.*, vol. 3, no. 2, pp. 680–687, Apr. 2018.

[3] K. Sreenath, T. Lee, and V. Kumar, "Geometric control and differential flatness of a quadrotor UAV with a cable-suspended load," in *Proc. 52nd IEEE Conf. Decis. Control*, 2013, pp. 2269–2274.

[4] E. N. Barmpounakis, E. I. Vlahogianni, and J. C. Golias, "Unmanned aerial aircraft systems for transportation engineering: Current practice and future challenges," *Int. J. Transp. Sci. Technol.*, vol. 5, no. 3, pp. 111–122, 2016.

[5] T. A. Rodrigues, N. L. O. Jay Patrikar, H. S. Matthews, S. Scherer, and C. Samaras, "Drone flight data reveal energy and greenhouse gas emissions savings for very small package delivery," *Patterns*, vol. 3, no. 8, 2022, Art. no. 8100569.

[6] H. Li, H. Wang, C. Feng, F. Gao, B. Zhou, and S. Shen, "AutoTrans: A complete planning and control framework for autonomous UAV payload transportation," *IEEE Robot. Automat. Lett.*, vol. 8, no. 10, pp. 6859–6866, Oct. 2023.

[7] J. Zeng, P. Kotaru, M. W. Mueller, and K. Sreenath, "Differential flatness based path planning with direct collocation on hybrid modes for a quadrotor with a cable-suspended payload," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 3074–3081, Apr. 2020.

[8] S. Tang and V. Kumar, "Mixed integer quadratic program trajectory generation for a quadrotor with a cable-suspended payload," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2015, pp. 2216–2222.

[9] H. Wang, H. Li, B. Zhou, F. Gao, and S. Shen, "Impact-aware planning and control for aerial robots with suspended payloads," *IEEE Trans. Robot.*, vol. 40, pp. 2478–2497, 2024.

[10] C. Y. Son, H. Seo, D. Jang, and H. J. Kim, "Real-time optimal trajectory generation and control of a multi-rotor with a suspended load for obstacle avoidance," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 1915–1922, Apr. 2020.

[11] P. Foehn, D. Falanga, C. Luethi, M. Achtelik, R. Siegwart, and D. Scaramuzza, "Fast trajectory optimization for agile quadrotor maneuvers with a cable-suspended payload," in *Proc. Robot., Sci. Syst.*, Cambridge, MA, USA, Jul. 2017, doi: 10.15607/RSS.2017.XIII.030}.

[12] C. Y. Son, D. Jang, H. Seo, T. Kim, H. Lee, and H. J. Kim, "Real-time optimal planning and model predictive control of a multi-rotor with a suspended load," in *Proc. Int. Conf. Robot. Automat.*, 2019, pp. 5665–5671.

[13] J. Schulman et al., "Motion planning with sequential convex optimization and convex collision checking," *Int. J. Robot. Res.*, vol. 33, no. 9, pp. 1251–1270, 2014.

[14] G. Yu, D. Cabecinhas, R. Cunha, and C. Silvestre, "Aggressive maneuvers for a quadrotor-slung-load system through fast trajectory generation and tracking," *Auton. Robots*, vol. 46, Apr. 2022, pp. 499–513.

[15] A. Pallar, G. Li, M. Sarvaiya, and G. Loianno, "Optimal trajectory planning for cooperative manipulation with multiple quadrotors using control barrier functions," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2025, pp. 2808–2814.

[16] A. Thirugnanam, J. Zeng, and K. Sreenath, "Safety-critical control and planning for obstacle avoidance between polytopes with control barrier functions," in *Proc. Int. Conf. Robot. Automat.*, 2022, pp. 286–292.

[17] Z. Wang, X. Zhou, C. Xu, and F. Gao, "Geometrically constrained trajectory optimization for multicopters," *IEEE Trans. Robot.*, vol. 38, no. 5, pp. 3259–3278, Oct. 2022.

[18] T. Lee, M. Leok, and N. H. McClamroch, "Geometric tracking control of a quadrotor UAV on SE(3)," in *Proc. 49th IEEE Conf. Decis. Control*, 2010, pp. 5420–5425.

[19] X. Zhang, A. Liniger, and F. Borrelli, "Optimization-based collision avoidance," *IEEE Trans. Control Syst. Technol.*, vol. 29, no. 3, pp. 972–983, May 2021.

[20] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[21] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi—A software framework for nonlinear optimization and optimal control," *Math. Program. Computation*, vol. 11, pp. 1–36, 2019.

[22] A. Wachter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Math. Program.*, vol. 106, pp. 25–57, Mar. 2006.

[23] J. Schulman, J. Ho, A. X. Lee, I. Awwal, H. Bradlow, and P. Abbeel, "Finding locally optimal, collision-free trajectories with sequential convex optimization," in *Proc. Robot., Sci. Syst.*, 2013, vol. 9, no. 1, pp. 1–10.

[24] M. Sarvaiya, G. Li, and G. Loianno, "HPA-MPC: Hybrid perception-aware nonlinear model predictive control for quadrotors with suspended loads," *IEEE Robot. Automat. Lett.*, vol. 10, no. 1, pp. 358–365, Jan. 2025.

[25] Z. Wu, Z. Wang, and H. Zhang, "GPU-accelerated optimization-based collision avoidance," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2024, pp. 7561–7567.