IEEE Transactions on Robotics (T-RO) paper presented at
2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Impact-Aware Planning and Control for Aerial Robots With Suspended Payloads

Haokun Wang, *Graduate Student Member, IEEE*, Haojia Li, *Graduate Student Member, IEEE*, Boyu Zhou, Fei Gao, *Member, IEEE*, and Shaojie Shen

***Abstract*—A quadrotor with a cable-suspended payload imposes great challenges in impact-aware planning and control. This joint system has dual motion modes, depending on whether the cable is slack or not, and presents complicated dynamics. Therefore, generating feasible agile flight while preserving the retractable nature of the cable is still a challenging task. In this article, we propose a novel impact-aware planning and control framework that resolves potential impacts caused by motion mode switching. Our method leverages the augmented Lagrangian method to solve an optimization problem with nonlinear complementarity constraints, which ensures trajectory feasibility with high accuracy while maintaining efficiency. We further propose a hybrid nonlinear model predictive control method to address the model mismatch issue in agile flight. Our methods have been comprehensively validated in both simulation and experiments, demonstrating superior performance compared to existing approaches. To the best of our knowledge, we are the first to successfully perform automatic multiple motion mode switching for aerial payload systems in real-world experiments.**

***Index Terms*—Aerial systems: applications, intelligent transportation systems, motion and path planning, motion control.**

Nomenclature

$\mathcal{Q}, \mathcal{L}, \mathcal{W}$	Quadrotor frame, payload frame, and world frame.
$\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z \in \mathbb{R}^3$	Unit vector in the axis-direction of the world frame.
$\mathbf{x}_\mathcal{B}, \mathbf{y}_\mathcal{B}, \mathbf{z}_\mathcal{B} \in \mathbb{R}^3$	Unit vector in the axis-direction of the quadrotor frame.
$\mathbf{x}_\mathcal{Q}, \mathbf{x}_\mathcal{L} \in \mathbb{R}^3$	Position of the quadrotor and the payload.
$\mathbf{v}_\mathcal{Q}, \mathbf{v}_\mathcal{L} \in \mathbb{R}^3$	Linear velocity of quadrotor and the payload.
$\mathbf{R}_\mathcal{Q} \in \mathbb{R}^3$	Orientation of the quadrotor.
$\boldsymbol{\omega}_\mathcal{Q} \in \mathbb{R}^3$	Body rate of the quadrotor.
$m_\mathcal{Q}, m_\mathcal{L} \in \mathbb{R}$	Mass of the quadrotor and the payload.
$r_\mathcal{Q}, r_\mathcal{L} \in \mathbb{R}$	Safe margin radius of the quadrotor and the payload.
$\tilde{\mathbf{p}} \in \mathbb{R}^3$	Vector pointing from the quadrotor to the payload.
$\mathbf{p} \in \mathbb{S}^2$	Unit vector in the direct of $\tilde{\mathbf{p}}$.
$\dot{\mathbf{p}} \in \mathbb{R}^3$	First derivative with respect to time of vector $\mathbf{p}$.
$l \in \mathbb{R}$	Distance between the quadrotor and the payload.
$l_0 \in \mathbb{R}$	Cable length.
$f, f_T \in \mathbb{R}$	Thrust of the quadrotor, and tension in the cable.

Manuscript received 17 October 2023; revised 25 February 2024; accepted 5 March 2024. Date of publication 25 March 2024; date of current version 12 April 2024. This paper was recommended for publication by Guest Associate Editor Alessandro Saccon and Editor P. Robuffo Giordano upon evaluation of the reviewers' comments. This work was supported in part by the Research Grants Council General Research Fund (RGC GRF) project RMGS20EG20, in part by the HKUST-DJI Joint Innovation Laboratory, and in part by the National Natural Science Foundation of China under Grant 62322314. *(Haokun Wang and Haojia Li contributed equally to this work.) (Corresponding authors: Fei Gao; Boyu Zhou.)*

Haokun Wang, Haojia Li, and Shaojie Shen are with the Cheng Kar-Shun Robotics Institute, The Hong Kong University of Science and Technology, Hong Kong (e-mail: hwangeh@connect.ust.hk; hlied@connect.ust.hk; eeshaojie@ust.hk).

Boyu Zhou is with the School of Artificial Intelligence, Sun Yat-Sen University, Zhuhai 519082, China (e-mail: zhouby23@mail.sysu.edu.cn).

Fei Gao is with the Institute of Cyber-Systems and Control, College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China, and also with the Huzhou Institute, Zhejiang University, Huzhou 313000, China (e-mail: fgaoaa@zju.edu.cn).

The video supplement is available at https://sites.google.com/view/suspended-payload/.

This article has supplementary downloadable material available at https://doi.org/10.1109/TRO.2024.3381555, provided by the authors.

Digital Object Identifier 10.1109/TRO.2024.3381555

I. Introduction

AERIAL robots with suspended payloads play a vital role in logistics, transportation, disaster relief, and other scenarios. One intriguing aspect of this system is that it exhibits dual motion modes in agile flights, as shown in Fig. 1. Specifically, the retractable cable allows the system to operate in two dynamic modes depending on whether the cable is taut or slack (Fig. 2).

It is attractive to exploit mode switching for the suspended payload system, which allows for performing transportation tasks in complex and narrow spaces. For example, the suspended payload system can leverage its unique capabilities to navigate through narrow windows and inaccessible areas, aiding in fire extinguishing and reconnaissance tasks. By seamlessly switching between slack and taut modes, these robots enhance efficiency and safety in rescue operations.

However, producing feasible and practical trajectories with hybrid modes poses two significant challenges. First, dual individual subsystem dynamical models are required to describe the system dynamics since the robot has two motion modes. The existence of dual system dynamics descriptions dramatically increases the complexity of feasible planning and practical control. Second, potential *impacts* caused by transitions between two subsystems will break the robotic status consistency. Therefore, to address the above issues, it is crucial to design a concise system dynamics description that effectively captures the intricate mode-switching behavior. On the other hand, *impact-aware*

Fig. 1. Quadrotor with a cable-suspended payload successfully navigates through a narrow circular gate, demonstrating natural mode-switching movements in the suspended payload system.

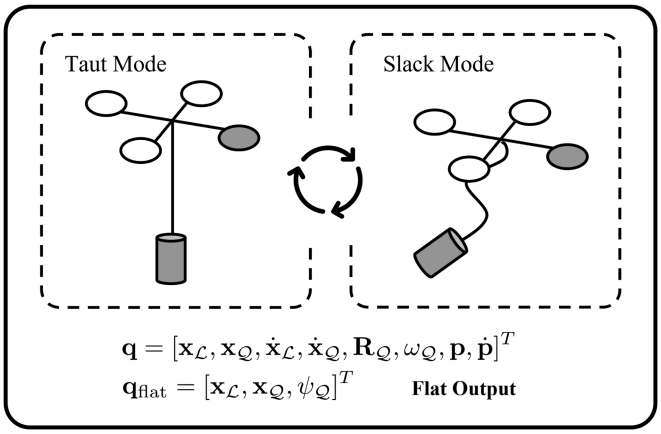


Fig. 2. Diagram of the system's two motion modes, with differential platform outputs that apply to both modes.

planning and control methods are essential for generating feasible trajectories and ensuring accurate trajectory tracking with hybrid motion modes.

Developing a unified and concise dynamical model for the suspended payload system is crucial for enabling agile flight with automatic switching between modes. The previous work proposed utilizing a smooth transition function to map the current subsystem's flat output to the following subsystem [1]. Nevertheless, this approach suffers from two weaknesses. First, it is challenging to constrain the robot's behaviors through trajectory planning [2]. Second, the plan of trajectories and the assignment of motion modes are independent [3]. In contrast, a unified dynamic model with complementarity constraints allows simultaneous optimization of the trajectories and allocation of motion modes. However, existing approaches oversimplify system state and control input constraints on the consistency of robot movement, which hinders the feasibility and practice of trajectories [4], [5].

To address the challenge of maintaining the consistency of robotic status and avoiding the potential discontinuous impacts, we begin by observing an ideal trajectory, as shown in Fig. 3. The potential impact always occurs at the moment of mode change. More precisely, the cable tension increases dramatically when the suspended payload system changes from slack to taut mode. As a result, the quadrotor is involved in an implicit inelastic collision and possibly crashes.

Based on the above observations, a viable agile flight for the suspended payload system needs to satisfy two essential features. 1) *Consistency:* A feasible trajectory with multiple motion modes must maintain the status continuity, especially during the mode transitions. 2) *Compactness:* An ideal planning algorithm should simultaneously optimize both the trajectory and the allocation of motion modes. Moreover, the allocation of modes is autonomously determined by the robot itself to minimize a general purpose, i.e., energy consumption. Consistency ensures that the robot avoids infeasible status caused by sudden state changes. Furthermore, compactness avoids redundant mode-switching while maintaining the large feasible region for the robot.

In our impact-aware framework, multiple segments of polynomial parameterization are employed to preserve an intrinsic consistency of trajectories without losing efficiency. Nonlinear complementarity constraints that are used to select a specific mode and the necessary state-input constraints are considered in a compact formulation of an optimization problem. This problem is an optimization problem with nonlinear complementarity constraints (ONCC). From the perspective of practicability, we emphasize the simplicity and compactness of an objective function in the optimization problem. The feasibility of trajectories is ensured by considering the tight and necessary constraints. As a result, the transition of motion modes occurs automatically.

Despite the compact formulation of ONCC, complementarity constraints violate several basic constraint qualifications (CQs) assumptions, such as the Mangasarian–Fromovitz CQ (MFCQ), as discussed in many studies [6], [7]. Therefore, many commonly used optimization solvers, such as SNOPT [8] and Ipopt [9], cannot obtain trajectories with sufficient accuracy and efficiency [6]. Thus, we propose an efficient *impact-aware* planning algorithm for the suspended payload system to overcome the challenge. This algorithm is a novel implementation of the Powell–Hestenes–Rockafellar augmented Lagrangian method (ALM) [10], [11], [12] by utilizing the Lewis and Overton line search method [13] and the limited memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) quasi-Newton method [14] . This line search method enables our algorithm to solve a nonconvex and partly nonsmooth unconstrained optimization problem in the inner loop while maintaining efficiency. Our implementation also preserves the analytic update of the dual variables in the outer loop of the framework to conserve accuracy.

In addition, an *impact-aware* control strategy is also essential in practical flight. Although an excellent performance of model predictive control (MPC) methods is applied in the suspended payload system [15], [16], [17], avoiding the impacts caused by mode switching is still challenging. The remarkable control effect of the MPC methods relies on an accurate model description [18], [19], [20]. A mismatch of system dynamics

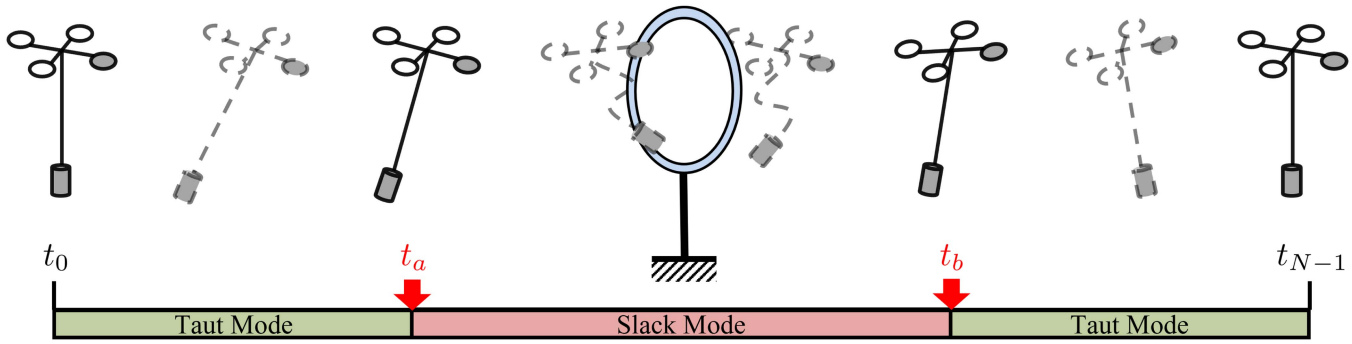


Fig. 3. Trajectory with mode switching of the quadrotor with cable-suspended payload. The robot is required to pass through a narrow gate without collision. The robot has two motion modes: 1) when the cable is taut, the quadrotor and payload behave as a single entity, and 2) when the cable is slack, the payload behaves independently of the quadrotor. The moment of mode switching is marked in red. During the first phase of the motion (left part of the figure), the cable is taut, and the system behaves as a single entity. When passing through the gate, the cable becomes slack, and after clearing the gate, the cable becomes taut again, coupling the two subsystems together once again.

leads to huge tracking errors. However, general MPC methods considering only a single model have a natural mismatch problem at the point of mode transition for the suspended payload system. We propose a hybrid nonlinear MPC (HNMPC) method to solve this issue. The HNMPC employs a discrete switching variable to accurately distinguish the subsystem dynamics to fit the suspended payload system's performance. Consequently, excessive tracking errors can be avoided near the points when the transition between modes occurs.

In summary, this article proposes a systematic solution to the feasible and practical agile flight for a quadrotor with a cable-suspended payload. Our main contributions are as follows.

1) An impact-aware planning method is proposed to address the optimization problem with a compact formation that can automatically generate feasible trajectories with hybrid modes (see Section IV).
2) A novel hybrid nonlinear MPC method is designed to solve the model mismatch problem (see Section V). Our results demonstrate that the proposed method significantly reduces control errors in practical scenarios. Our method is shown to prevent catastrophic failure of the suspended system during experimental tests.
3) The planning framework and controller are extensively verified in simulations and physical experiments. To the best of our knowledge, we are the first to successfully achieve flight with automatic mode-switching in the real world (see Section VI).
4) Moreover, we open-source the code to facilitate further validation and expansion of related work by the community.[1]

II. Related Work

In this section, we review previous works on planning and control for aerial robots with cable-suspended payload. We remark that a compact dynamics description of the suspended payload system, an impact-aware planning method, and a robust controller are essential to conducting a practical flight.

A. Hybrid System Dynamics of Aerial Robots With Suspended Payload

Sreenath et al. [1], [2] pioneered the study of the differential flatness property of a quadrotor with a cable-suspended payload. Since the cable can be in two different states, relaxed or taut, the differential flat outputs of the suspended payload system are different in these two states. They characterize the switching of the system state between different dynamical models by defining two transition maps, which is a more sophisticated way of modeling such a robot. Previous works [4], [5] introduce complementarity constraints into the trajectory optimization model to build a unified dynamical model of this suspended payload system with multiple modes in a concise form. Although these works allow the robotic system to switch motion mode, none can generate trajectories that have automatic mode-switching without additional operations (e.g., preassigning modal sequences or adding a penalty term for rope tension to the objective function).

We present a unified system dynamical model of the suspended payload system using complementarity constraints. Two major differences exist between [4], [5], and our methods. First, we use multiple segments of polynomials rather than discrete robot states to preserve the natural consistency of trajectories. Second, the necessary state-input constraints are considered in a compact formation as an ONCC. Both are essential to the feasibility and practice of trajectories with hybrid modes.

B. Impact-Aware Trajectory Planning

The impact often occurs when the robot makes or breaks contact with the environment [21], [22], [23] or when its subsystem dynamics switch. For example, a cable carrying a heavy payload at its endpoints changes from slack to taut, the fingertips of a multifingered robot hand make or break contact with the surface of a manipulated object, and a bipedal robot interacts periodically with the ground while walking continuously. Previous works [21], [22], [23], and [24] have extensively investigated

[1]The open-source code is available at https://github.com/HKUST-Aerial-Robotics/IMPACTOR

tasks typically associated with contact dynamics, such as manipulation and locomotion. Impact-aware planning remains a challenging problem due to explicit or implicit contacts and potentially complex collisions during mode switching, as shown in Fig. 3.

Complementarity constraints provide concise formation of expressing switching modes and making or breaking the contact. Therefore, modeling the motion planning problem of a robotic system containing multiple dynamical modes or contacts with the environment as ONCC is a natural consequence. However, the complementarity constraints are usually MFCQ violating [7], so the convergence of many numerical optimization algorithms cannot be guaranteed when dealing with such degenerate optimization problems. Previous works [4], [21], and [24] used the sequential quadratic program (SQP) to solve the ONCC with impressive results for trajectory planning of robotic systems, such as bipedal robots and a quadrotor with a cable-suspended payload. As a classical optimization algorithm framework, ALM has good robustness in dealing with degenerate optimization problems [6], [25], [26]. Theoretical analysis [25], [26] shows that the convergence of ALM depends only on the weak CQs and not on any CQ. There are similarities in principle between SQP and ALM, while the latter has better global convergence guarantees [7]. In this work, we propose an optimization framework preserving the ALM framework for a quadrotor with a cable-suspended payload. Combining polynomial parameterization and the minimum control effort theory [27], we can perform accurate optimization and significantly accelerate the speed of solving the optimization problem.

C. Impact-Aware Control for Suspended Payload System

Designing impact-aware control methods for a quadrotor with a cable-suspended payload system is still extremely challenging due to the hybrid mode.

Various controllers for a quadrotor with a suspended payload system have been proposed. Several past works [28], [29], [30] aim to minimize the swing angle of the suspended payload for stabilization. However, they often constrain the range of motion of the payload, which limits its ability to execute high-speed agile maneuvers.

Some geometric controllers [2], [31] involve a coordinate-free dynamic model derived from the properties of Lie groups. The cascade structure employed by the above controllers, where the desired load angle is determined from the desired load position, and then the desired quadrotor attitude is determined from the desired load angle. However, this approach cannot effectively control both the quadrotor and the load simultaneously.

As a result, directly applying these controllers for tracking trajectories containing both taut and slack modes may not be suitable. Based on the coordinate-free dynamic model, researchers [15], [16], [17] propose the NMPC for the suspended payload system, but they do not consider the slack mode. The work proposed by Crousaz et al. [32], [33] is close to our purpose and gave an impressive simulation. The drawback of this work is that using a linearized dynamic model cannot guarantee the accuracy, and there is no demonstration on a real quadrotor.

Our impact-aware control method addresses the crucial challenge of flying the trajectories with natural hybrid motion modes. To fill this gap, we propose the HNMPC method to distinguish dual system dynamics to fit the actual performance of the robot. Our HNMPC preserves the excellent predictive performance of MPC methods in complex scenarios and provides enhanced robustness in the trajectories with hybrid modes tracking.

III. Preliminary

This section presents a unified and concise dynamic model of a quadrotor with a cable-suspended payload (see Section III-A). Then, the primary form of the impact-aware trajectory optimization problem is introduced (see Section III-B).

A. System Dynamics

A quadrotor with a cable-suspended payload is a complex robotic system with dual motion modes. In each mode, we need an independent subsystem dynamical model to describe the change in the robotic states. Building a unified and concise dynamic model in advance is necessary to generate feasible and practical trajectories for this system.

Compared to the transition map [1], [2], using complementarity constraints to represent natural motion mode switching has two advantages. It does not require a predetermined sequence of motion modes. In addition, the robot's behaviors can be constrained in trajectory planning [4]. The complementarity constraints for the suspended payload system is

$$f_T(t) \geq 0 \tag{1}$$

$$l_0 - l(t) \geq 0 \tag{2}$$

$$f_T(t)(l_0 - l(t)) = 0 \tag{3}$$

where $l(t)$ represents the distance between the quadrotor and the payload at time t. $f_T(t)$ represents a tension in the cable at time t. The equality equation describes that only one motion mode of the suspended payload system holds at the moment t, i.e., the slack mode holds if tension is zero, or the taut mode holds if the distance is equal to the cable length l_0.

A compact dynamical model of the robot clearly describes its differential flatness. As shown in Fig. 2, we chose $\mathbf{q}_{\text{flat}} = [\mathbf{x}_{\mathcal{L}}, \mathbf{x}_{\mathcal{Q}}, \psi_{\mathcal{Q}}]$ as flat outputs of the suspended payload system regardless of the system is in taut mode or slack mode. Consequently, the equations of motion for the robotic system are obtained as

$$f_T = \|m_{\mathcal{L}}(\ddot{\mathbf{x}}_{\mathcal{L}} + g\mathbf{e}_z)\| \tag{4}$$

$$\tilde{\mathbf{p}} = \mathbf{x}_{\mathcal{L}} - \mathbf{x}_{\mathcal{Q}} \tag{5}$$

$$l = \|\tilde{\mathbf{p}}\| = \|\mathbf{x}_{\mathcal{L}} - \mathbf{x}_{\mathcal{Q}}\| \tag{6}$$

$$\mathbf{p} = \frac{\tilde{\mathbf{p}}}{\|\tilde{\mathbf{p}}\|} \tag{7}$$

$$\mathbf{z}_{\mathcal{B}} = \frac{m_Q(\ddot{\mathbf{x}}_Q + g\mathbf{e}_z) - f_T\mathbf{p}}{\|m_Q(\ddot{\mathbf{x}}_Q + g\mathbf{e}_z) - f_T\mathbf{p}\|} = \frac{1}{f}(m_Q(\ddot{\mathbf{x}}_Q + g\mathbf{e}_z) - f_T\mathbf{p}) \tag{8}$$

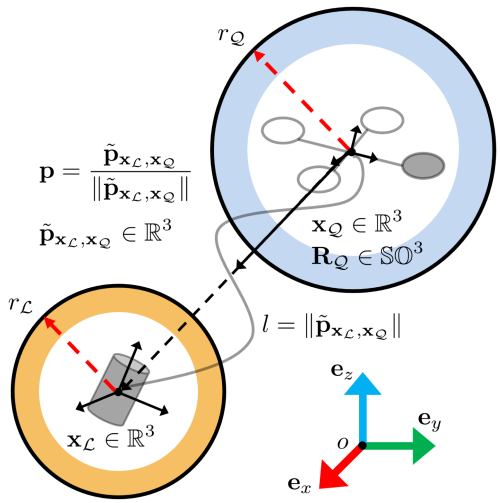


Fig. 4. State variables of a quadrotor with a cable-suspended payload. The state variables are used in the description of dynamic modeling and control of the suspended payload system.

$$\mathbf{y}_\mathcal{B} = \frac{\mathbf{z}_\mathcal{B} \times \mathbf{x}_\mathcal{C}}{\|\mathbf{z}_\mathcal{B} \times \mathbf{x}_\mathcal{C}\|}, \text{ where } \mathbf{x}_\mathcal{C} = [\cos\psi, \sin\psi, 0]^T \tag{9}$$

$$\mathbf{x}_\mathcal{B} = \mathbf{y}_\mathcal{B} \times \mathbf{z}_\mathcal{B}. \tag{10}$$

In the slack mode, the payload's motion follows a projectile model. At the same time, the quadrotor's states, including the angular velocity, can be computed based on its position $\mathbf{x}_\mathcal{Q}$, its yaw angle $\phi_\mathcal{Q}$, and their derivatives using the differential flatness proposed in [34]. Thus, our (8) is entirely equivalent to [34, (3)] in the slack mode. In the taut mode, we utilize [1, Lemma 1] to derive the quadrotor's states. First, we calculate the quadrotor's position $\mathbf{x}_\mathcal{Q}$ and its derivatives based on the payload's position $\mathbf{x}_\mathcal{L}$ and its corresponding derivatives. Subsequently, we calculate the complete quadrotor's states, similar to the slack mode, by considering the quadrotor's position $\mathbf{x}_\mathcal{Q}$, yaw angle $\phi_\mathcal{Q}$, and their derivatives. The system's state variables are shown in Fig. 4. We provide a detailed illustration of deriving the system state from the flat output, using angular velocity as an example in Appendix A.

The flatness property of the suspended payload system reduces the number of decision variables in trajectory planning and improves optimization efficiency. Moreover, the compact dynamic model is a critical component of the HNMPC (see Section V), allowing us to accurately predict the suspended payload system's behavior and achieve robust control in challenging scenarios.

B. Trajectory Optimization Problem

In impact-aware trajectory planning, we consider the following general constrained optimization problem:

$$\begin{aligned} \min_{\mathbf{x}=[\mathbf{w},\mathbf{T}]} \quad & \mathcal{J}(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{g}(\mathbf{q}^{[k]}(t)) \leq \mathbf{0} \\ & \mathbf{h}(\mathbf{q}^{[k]}(t)) = \mathbf{0} \end{aligned}$$

$$\forall t \in [0, \|\mathbf{T}\|_1] \tag{11}$$

where $\mathcal{J} : \mathbb{R}^s \to \mathbb{R}$ is the objective function. The dimension of decision variable $\mathbf{x}$ is $s = N \times (M+1) + M$, where N and M are the robot's configuration space dimension and the number of trajectory pieces. The decision variable $\mathbf{x}$ consists of two parts: 1) the end points $\mathbf{w} = [\mathbf{w}_0^T, \mathbf{w}_1^T, \ldots, \mathbf{w}^M]^T \in \mathbb{R}^{N\times(M+1)}$ of all pieces and 2) the time allocations $\mathbf{T} = [\tau_0, \tau_1, \ldots, \tau_{M-1}] \in \mathbb{R}^M_{>0}$ of all pieces. Functions $\mathbf{g} : \mathbb{R}^{N\times(k+1)} \to \mathbb{R}^I$ and $\mathbf{h} : \mathbb{R}^{N\times(k+1)} \to \mathbb{R}^E$ are inequality and equality constraints, where I and E represent the total number of them separately. It is worth noting that all constraints at time step t are calculated given by finite derivatives of the system state, which is denoted as

$$\mathbf{q}^{[k]}(t) = [\mathbf{q}^T(t), \dot{\mathbf{q}}^T(t), \ldots, \mathbf{q}^{(k)^T}(t)]^T. \tag{12}$$

The M-piece trajectory $\mathbf{q}$ whose ith piece is denoted by a $(2k-1)$-degree polynomial

$$\mathbf{q}_i(t) = \mathbf{c}_i \phi(t - t_{i-1}),\ t \in [t_{i-1}, t_i] \tag{13}$$

where $\mathbf{c}_i \in \mathbf{R}^{N\times 2^{\cdot}k}$ is a matrix of coefficients, ϕ is a basis function, and timestamp t is defined as $t_0 = 0$ and $t_i = \sum_{j=1}^{i} \tau_{j-1}, \forall i \in \{1, \ldots, M\}$. In this work, we set $\phi(a) = [1, a, a^2, \ldots, a^{2k-1}]^T$ as the basis function. The pieces of $\mathbf{q}$ are required to satisfy boundary constraints and continuity constraints, defined as

$$\begin{cases} \mathbf{q}_i(t_{i-1}) = \mathbf{w}_{i-1} \\ \mathbf{q}_i(t_i) = \mathbf{w}_i \\ \quad\vdots \\ i = \{1, 2, \ldots, M\} \end{cases} \quad \text{boundary constraints}$$

$$\begin{cases} \mathbf{q}_{j-1}(t_{j-1}) = \mathbf{q}_j(t_{j-1}) \\ \dot{\mathbf{q}}_{j-1}(t_{j-1}) = \dot{\mathbf{q}}_j(t_{j-1}) \\ \quad\vdots \\ \mathbf{q}^{(k)}_{j-1}(t_{j-1}) = \mathbf{q}^{(k)}_j(t_{j-1}) \\ j = \{2, \ldots, M\} \end{cases} \quad \text{continuity constraints.}$$

Similar to the calculation of MINCO trajectories [27] and the minimum snap trajectories [34], we obtain the optimal polynomial coefficients by solving quadratic programming with the boundary and continuity constraints above. Therefore, as shown in the trajectory representation part of Fig. 5, given a set of values of $\mathbf{w}$ and $\mathbf{T}$, we can obtain the trajectories $\mathbf{q}$. The feasible set of the above optimization problem is the set that satisfies all constraints, i.e., $\mathcal{S} = \{\mathbf{x} \in \mathbb{R}^s \mid \mathbf{g}_\mathbf{x} \leq \mathbf{0}, \mathbf{h}_\mathbf{x} = \mathbf{0}\}$.

There are many benefits to using the formulation (11). First, we can optimize the trajectory in both spatial and temporal space, i.e., the decision variables contain the endpoints of multiple trajectory segments and the duration of each piece. Second, we can use the chain rule to conveniently compute the gradients of the objective function for the endpoints and time allocation. For example, if we use a polynomial basis to represent pieces, then coefficients $\mathbf{c}_\mathbf{x}$ of the polynomial trajectories with minimum efforts can be calculated analytically according to [27, Theorem 2]. After obtaining the gradient of the objective function concerning the coefficients $\mathbf{c}_\mathbf{x}$, we can calculate the gradient to waypoints

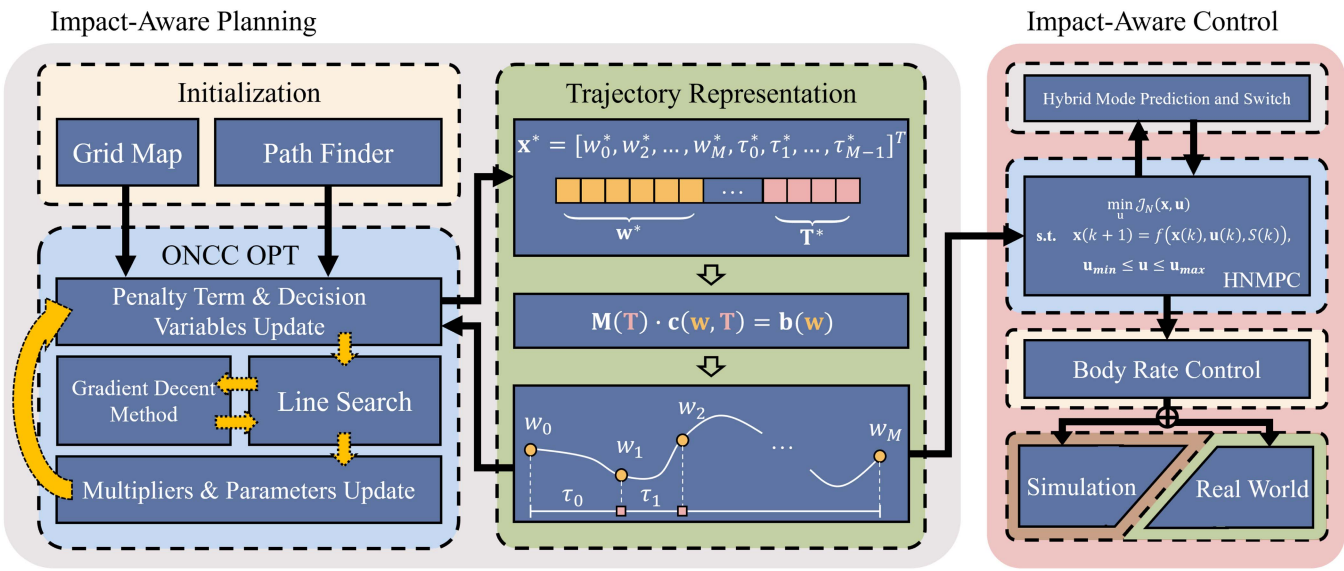


Fig. 5. Proposed impact-aware planning and control methods consist of two main components: trajectory optimization and control. The trajectory optimization module includes initialization, ONCC optimization, and trajectory representation. The initialization utilizes a grid map and a pathfinder module to generate an initial collision-free path. The ONCC optimization constructs an optimization problem with constraints based on the initial path and transcribes it into an unconstrained optimization problem. Both line search and gradient descent methods are used to solve the unconstrained subproblem and obtain the optimal solution. The dual variables and penalty parameters are updated iteratively until the solution is optimal. The trajectory representation subpart converts the waypoints and time allocation into continuous polynomial trajectories, which are used as a reference input for the control module. The control module includes two subparts: high-level control and low-level control. The high-level control utilizes an HNMPC approach to track the planned trajectory. The low-level control computes the desired motor torques and sends them to the quadrotor's propellers. The overall system can be tested in a simulation or real-world setting.

$\mathbf{w}$ and time allocation $\mathbf{T}$ with the chain rule. In addition, we can take advantage of the differential flatness of the suspended payload system to further reduce the dimension of the decision variable.

IV. Impact-Aware Planning for the Suspended Payload System

This section addresses the challenge of obtaining feasible and practical trajectories with hybrid modes for a quadrotor with a cable-suspended payload. First, we formulate a nonlinear trajectory optimization problem with complementarity and other necessary state-input constraints based on the general formulation 11 (Section IV-A). Then, we propose an efficient algorithm to obtain feasible and consistent trajectories in Section IV-B. Furthermore, multiple segments of polynomial parameterization and the state-input constraints used to ensure feasibility and practice are detailed in Section IV-C. The overview of our complete framework is shown in Fig. 5.

A. Problem Formulation

We propose the trajectory planning problem with nonlinear complementarity constraints based on formulation (11). We consider an essential cost consists of control energy and total time penalty in the objective function

$$\min_{\mathbf{x}} \underbrace{\int_0^{\|\mathbf{T}\|_1} \mathbf{q}^{(k)^T}(\tau)\mathbf{Q}_w\mathbf{q}^{(k)}(\tau)\mathrm{d}\tau}_{\text{Lagrange term}} + \underbrace{\alpha\|\mathbf{T}\|_1}_{\text{Mayer term}} \tag{14a}$$

$$\text{subject to} \quad \mathbf{g}(\mathbf{q}^{[k]}(t)) \le \mathbf{0} \tag{14b}$$

$$\mathbf{h}(\mathbf{q}^{[k]}(t)) = \mathbf{0} \tag{14c}$$

$$\forall t \in [0, \|\mathbf{T}\|_1] \tag{14d}$$

where $\mathbf{Q}_w \in \mathbb{R}^{N\times N}$ is a positive diagonal matrix, α is a positive scalar, and the specific forms of equality and inequality constraints are shown in Table I.

Remark 1: Considering only energy and time consumption in the objective function has two advantages. First, the feasibility of the proposed trajectories is maximized. Second, as no penalty terms related to cable tension or direction are included, the relaxation of the cable in the optimized trajectories occurs naturally.

Remark 2: The above equality and inequality constraints fully describe complementarity constraints. A complementarity constraint is equivalent to the combination of two inequality constraints and an equality constraint, i.e., $g_a(\mathbf{x}) \le 0, g_b(\mathbf{x}) \le 0$ and $h_c(\mathbf{x}) := \langle -g_a(\mathbf{x}), -g_b(\mathbf{x})\rangle = 0$.

B. Efficient Trajectory Optimization With Constraints

The problem proposed in Section IV-A is a typical ONCC. Many commonly used optimization solvers, such as SNOPT [8] and Ipopt [9], cannot obtain an ideal solution for the problem. Many studies have shown that ALM's local or global convergence does not depend on CQ or only on the weak ones enabling the robustness [6], [7]. Therefore, we follow the ALM and propose four steps to address the degeneracies caused by complementarity constraints.

TABLE I
STATE-INPUT CONSTRAINTS USED IN TRAJECTORY OPTIMIZATION

Description	Type	Function	Parameters
See section IV-C1.	Inequality	$g_o(t,\mathbf{c}) = \text{sign}_{\mathcal{E}}(\mathcal{M}_{r_{\text{safe}}}(t,\mathbf{c}))$	r_{safe}: the radius of safe margin.
See section IV-C2.	Inequality	$g_v(t,\mathbf{c}) = \|\mathbf{v}(t,\mathbf{c})\|_2^2 - v_{\max}^2$	$v_{\max}$: the maximum velocity.
See section IV-C2.	Inequality	$g_a(t,\mathbf{c}) = \|\mathbf{a}(t,\mathbf{c})\|_2^2 - a_{\max}^2$	$a_{\max}$: the maximum acceleration.
See section IV-C3.	Inequality	$g_\theta(t,\mathbf{c}) = \theta - \theta_{\max}$	$\theta_{\max}$: the maximum angle of deviation.
See section IV-C4.	Inequality	$g_u(t,\mathbf{c}) = \left(u(t,\mathbf{c}) - \frac{u_{\max}+u_{\min}}{2}\right)^2 - \left(\frac{u_{\max}-u_{\min}}{2}\right)^2$	$u_{\max}$: the maximum thrust, $u_{\min}$: the minimum thrust.
See section IV-C5.	Inequality	$g_l(t,\mathbf{c}) = \left(l(t,\mathbf{c}) - \frac{l_0+d_{\text{safe}}}{2}\right)^2 - \left(\frac{l_0-d_{\text{safe}}}{2}\right)^2$	l_0: the length of cable, d_{safe}: the minimum safe distance.
See section IV-C6.	Inequality	$g_{f_T}(t,\mathbf{c}) = \left(f_T(t,\mathbf{c}) - \frac{f_{T,\max}}{2}\right)^2 - \left(\frac{f_{T,\max}}{2}\right)^2$	$f_{T,\max}$: the maximum tension
See section IV-C7.	Equality	$\mathbf{h}_{\text{dyn}}(t,\mathbf{c}) = \|\tilde{\mathbf{p}}(t,\mathbf{c})\| \cdot (\mathbf{a}_{\mathcal{L}}(t,\mathbf{c}) + g\mathbf{e}_z) + \|\mathbf{a}_{\mathcal{L}}(t,\mathbf{c}) + g\mathbf{e}_z\| \cdot \tilde{\mathbf{p}}(t,\mathbf{c})$	$\tilde{\mathbf{p}}(t,\mathbf{c})$: the vector pointing from the quadrotor and the payload. g: the gravity, $\mathbf{e}_z$: the z-axis direction of the world frame.
See section IV-C8.	Equality	$h_{\text{comp}}(t,\mathbf{c}) = T(t,\mathbf{c}) \cdot (l_0 - l(t,\mathbf{c}))$	l_0: the length of cable

1) Initialization: The algorithm starts with an initial guess for the decision variables and the Lagrange multipliers. A constrained optimization problem is transcribed to an unconstrained subproblem by adding penalty terms to the objective function. The Powell–Hestenes–Rockafellar augmented Lagrangian function [10], [11], [12] corresponding to the constrained optimization problem defined by (11) is given by

$$\mathcal{L}_\rho(\mathbf{x},\boldsymbol{\lambda},\boldsymbol{\mu}) = f(\mathbf{x}) + \sum_{i=1}^{p} \frac{\rho}{2}\left[h_i(\mathbf{x}) + \frac{\lambda_i}{\rho}\right]^2 + \sum_{j=1}^{q} \frac{\rho}{2}\left[\max\left(0, g_j(\mathbf{x}) + \frac{\mu_j}{\rho}\right)\right]^2 \quad (15)$$

where $\mathbf{x} \in \mathcal{S}$ is the decision variables within the feasible set, $\boldsymbol{\lambda} \in \mathbb{R}^p$ and $\boldsymbol{\mu} \in \mathbb{R}_+^q$ are Lagrange multipliers, p and q are the number of equality and inequality constraints, and $\rho > 0$ is the penalty parameter.

In order to deal with the problem of low accuracy and convergence speed due to poor initial guess, we usually use a scaling formulation of the ALM. See Appendix B for details. In our initialization strategy, we utilize the A* algorithm to search for feasible paths for both the quadrotor and the payload. While this strategy identifies collision-free point pairs as initial values for the optimization process, these pairs often fail to satisfy the dynamics constraints. To address this, we introduce a warm-up strategy that refines the initial values by solving a miniature optimization problem focused on obstacle avoidance and dynamics constraints. Essentially, our warm-up strategy fine-tunes the positions of the original point pairs until they conform to obstacle avoidance and dynamics constraints.

2) Inner-Loop Optimization: Each unconstrained subproblem is solved using a nonlinear optimization solver. In this step, we utilize a line search method based on the weak Wolfe condition [13] to calculate the appropriate gradient descent step and use a quasi-Newton gradient method, such as L-BFGS [14], to obtain the proper gradient descent direction (see Algorithm 1). The line search satisfying the weak Wolfe condition improves the algorithm's robustness for nonsmooth problems. The L-BFGS method requires only first-order gradient information instead of the more computationally expensive second-order Hessian, significantly accelerating the efficiency of solving the inner-loop unconstrained optimization problem. We can choose the appropriate inner-loop optimization algorithm according to the specific optimization problem. For example, one can choose a specific class of nonlinear conjugate gradient algorithms with convergence guarantees under the weak Wolfe condition [35], [36].

3) Outer-Loop Update: In this step, the trajectory and the Lagrange multipliers are updated analytically based on the previous step's optimal solution and constraint violations. Once solving an optimal solution of the unconstrained subproblem, it is necessary to calculate the violation of each constraint and use it to analytically update the values of the dual variables.

For the method of dual variables update, we refer to a shift method instead of calculating the dual variables directly with penalty values. The shifts method can mitigate an inconsistency between the subproblem and the original problem caused by a variation of the penalty parameter ρ. Then, we update the penalty coefficient ρ according to the following rules: for the equality constraint, increase the penalty parameter to reduce the degree of constraint violation; for the inequality constraint, increase the penalty parameter to reduce both the degree of constraint violation and the degree of shift.

In practice, we also calculate the scaling factor based on the initial solution's gradient information to adjust the problem's condition number and further improve the accuracy and robustness of the solution. See Appendix B for details.

Algorithm 1: Impact-Aware Trajectory Optimization.

INNER-OPTIMIZE $(\mathbf{x}^0, \rho, \overline{\lambda})$
  $k \leftarrow 0, \mathbf{B}^0 \leftarrow \mathbf{I}$
  while $\|\nabla L_{\rho,\overline{\lambda}}(\mathbf{x}^k)\| > \delta$
    $\mathbf{d} \leftarrow -\mathbf{B}^k \nabla f(\mathbf{x}^k)$
    $t \leftarrow$ LINE SEARCH$(\mathbf{x}^k, \mathbf{d})$ [13]
    $\mathbf{x}^{k+1} \leftarrow \mathbf{x}^k + t\mathbf{d}$
    $\mathbf{B}^{k+1} \leftarrow$ L-BFGS [14]
    $k \leftarrow k + 1$
  endwhile
  return $\mathbf{x}^k$
ALM $(\mathbf{x}^0, n_{\max}, \lambda_{\min}, \lambda_{\max}, \rho_{\max}, \gamma)$
  $k \leftarrow 0, \rho^0 \leftarrow 1, \overline{\lambda}^0 \leftarrow \mathbf{0}$
  while C_{opt} **fails** or C_{feas} **fails**
    if $k > n_{\max}$, **break**
    $\mathbf{x}^* \leftarrow$ INNER-OPTIMIZE$(\mathbf{x}^k, \rho^k, \overline{\lambda}^k)$
    for c_i **in** $\mathcal{H} \cup \mathcal{G}$
      if $c_i \in \mathcal{H}$ or $\rho^k c_j(\mathbf{x}^k) \geq 0$
        $\overline{\lambda}_i^k \leftarrow \overline{\lambda}_i^k + \rho^k c_i(\mathbf{x}^k)$
      else, 0
    $\rho^k \leftarrow \min\left[(1+\gamma)\rho^k, \rho_{\max}\right]$
    $k \leftarrow k + 1$
  return $\mathbf{x}^*$

4) Convergence Checking: The algorithm converges when the optimality and precision conditions are satisfied. Otherwise, it returns to step Section IV-B2 to solve a new unconstrained optimization problem. We use the following conditions to check if the optimization is converged:

$$C_{\text{opt}} : \|\nabla L_\rho\|_\infty \leq \epsilon_{\text{opt}} \tag{16}$$

$$C_{\text{feas}} : \max\left\{\max_{i\in 1,\ldots,E}\left\{|h_i\left(\mathbf{x}^k\right)|\right\}, \max_{j\in 1,\ldots,I}\left\{g_j\left(\mathbf{x}^k\right)_+\right\}\right\} \leq \epsilon_{\text{feas}} \tag{17}$$

where ϵ_{opt} and ϵ_{feas} are positive threshold scalars.

In summary, we utilized ALM to obtain optimal and feasible solutions for the ONCC with four advantages as follows.

1) In inner-loop optimization, many well-established, robust, and efficient unconstrained optimization methods, such as the Lewis and Overton line search method [13] and the L-BFGS method [14], can be directly utilized. Thus, the generalization to nonlinear nonsmooth optimization problems is natural.
2) In outer-loop, the updating dual variables can be computed analytically according to the constraint violations. Thus, the algorithm works efficiently for problems even if the number of constraints is large.
3) Convergence and stability theory analysis show that the unconstrained subproblem in the inner-loop optimization can directly determine the convergence of the original constrained optimization problem in the outer loop [25].
4) This algorithm does not need to satisfy CQs [25], [26].

C. Trajectory Representation and Constraint Details

Mutiple segments of polynomial parameterization are essential to preserve a natural consistency of trajectories in our impact-aware planning framework. There are three advantages of parameterizing the trajectory with polynomial functions. First, the polynomial function parameterization can significantly reduce the number of parameters, and only a finite number of polynomial coefficients are needed to determine the entire trajectory. Second, given polynomial coefficients, their derivatives can be directly computed without adding redundant dynamical constraints, such as the trapezoidal collocation [37]. Moreover, benefiting from methods proposed by [27] that each segment of trajectories can be deformed analytically, we can increase the feasibility of trajectories in complex scenarios by increasing polynomial segments without excessive loss of efficiency.

In our approach, we ensure that the accelerations along the trajectories, including the moments of mode switching, remain continuous using the polynomial functions. Moreover, to ensure the practice of the trajectories, we employ dense sampling and perform constraint violation checks at each sampled point. Violation penalties are backpropagated to the decision variables, allowing the trajectory to be adjusted in both temporal and spatial dimensions to improve feasibility (as shown in Fig. 5). Hence, even at the moment of mode switching, the accelerations of both the quadrotor and the payload remain continuous.

Next, we describe each type of the necessary state-input constraints in detail. A summary of all constraints is listed in Table I.

1) Obstacle Avoidance: We use the signed distance function, defined as (18), to describe the relationship between a robot and obstacles

$$\begin{aligned} g_o(t,\mathbf{c}) &= \text{sign}_{\mathcal{E}}(\mathcal{M}(t,\mathbf{c})) \\ &= \begin{cases} -d(\mathcal{M}(t,\mathbf{c}), \partial\mathcal{E}), & \text{if } \mathcal{M}(t,\mathbf{c}) \cap \mathcal{E} = \emptyset \\ d(\mathcal{M}(t,\mathbf{c}), \partial\mathcal{E}), & \text{otherwise} \end{cases} \end{aligned} \tag{18}$$

where $\mathcal{M}(t,\mathbf{c})$ represents the convex set of the robot's multiple rigid bodies, $\mathcal{E}$ and $\partial\mathcal{E}$ are environment and its boundary. In this case, the suspended payload system can be described by a mass point model or a convex polyhedron, defined as

$$\mathcal{M}(t,\mathbf{c}) = \mathbf{R}(t,\mathbf{c})\mathcal{M}_o + \mathbf{p}(t,\mathbf{c}) \tag{19}$$

where $\mathbf{R}(t,\mathbf{c})$ is a diagonal block matrix as rotation of rigid bodies, $\mathbf{p}(t,\mathbf{c})$ is a translation vector, and $\mathcal{M}_o$ is the mass point or convex set of rigid bodies at the initial point. For full-dimensional robotic systems, time-varying convex polyhedrons are often used to describe the space occupied by the robot during its motion. For example, we use two spheres to represent a quadrotor and the corresponding payload separately. The rigid bodies should be $\mathcal{M}_o = \{[\mathbf{p}_\mathcal{L}, \mathbf{p}_\mathcal{Q}]^T \mid f_\mathcal{L}(\mathbf{p}_\mathcal{L}) \leq r_\mathcal{L}, f_\mathcal{Q}(\mathbf{p}_\mathcal{Q}) \leq r_\mathcal{Q}\}$, where $f_\mathcal{L}(\mathbf{p}) = \|\mathbf{p} - \mathbf{x}_\mathcal{L}^0\|$ and $f_\mathcal{Q}(\mathbf{p}) = \|\mathbf{p} - \mathbf{x}_\mathcal{Q}^0\|$ with $\mathbf{p} \in \mathbb{R}^3, r_\mathcal{L} \in \mathbb{R}$, and $r_\mathcal{Q} \in \mathbb{R}$.

2) Maximum Value of Velocity and High-Order Derivatives Constraints: To ensure that the robot's motion remains within a safe range, we limit the robot's dynamics with velocity and acceleration constraints, defined as $g_v(t,\mathbf{c}) = \|\mathbf{v}(t,\mathbf{c})\|_2^2 - v_{\max}^2$ and

$g_a(t,\mathbf{c}) = \|\mathbf{a}(t,\mathbf{c})\|_2^2 - a_{\max}^2$, where $v_{\max} \in \mathbb{R}$ and $a_{\max} \in \mathbb{R}$ are the use-defined maximum value. In some cases, we can also add constraints on higher order quantities, such as the jerk, to improve the flight stability.

3) Extreme Tilt Angle Constraint: In order to ensure safety, it is imperative to impose constraints on the maximum attainable tilt angle of the robot during motion. We denote the tilt angle $\theta(t,\mathbf{c})$ as the angle between the z-axis of the body frame $\mathbf{z}_{\mathcal{B}}$ and the z-axis of the world frame $\mathbf{e}_z$. The robot's orientation is represented by a rotation matrix $\mathbf{R} \in \mathrm{SO}(3)$. We can calculate θ given $\mathbf{R}$ as $\theta = \arccos(\mathbf{e}_z^T \mathbf{R} \mathbf{e}_z)$. Then, we use $g_\theta(t,\mathbf{c}) = \theta - \theta_{\max}$ to describe the extreme tilt angle constraint.

4) Maximum and Minimum Thrust Constraint: During trajectory planning for the robot system, incorporating constraints on the maximum and minimum thrust, denoted as $u(t,\mathbf{c})$, proves advantageous in preventing aggressive flight and ensuring disturbance rejection stability [38]. Consequently, the constraint that imposed to restrict the thrust magnitude throughout the robot's motion is given by $g_u(t,\mathbf{c}) = (u(t,\mathbf{c}) - \frac{u_{\max}+u_{\min}}{2})^2 - (\frac{u_{\max}-u_{\min}}{2})^2$, where $u_{\max}$ and $u_{\min}$ represent the maximum and minimum thrust separately.

5) Distance Between Robot and Payload: Ensuring a suitable range for the distance, denoted as $l(t,\mathbf{c})$, between the quadrotor and the payload is crucial. To prevent collisions, $l(t,\mathbf{c})$ remains at least a minimum safety distance denoted as d_{safe}. In addition, $l(t,\mathbf{c})$ must not exceed l_0, which is the length of the cable. Consequently, we define the distance constraint as $g_l(t,\mathbf{c}) = (l(t,\mathbf{c}) - \frac{l_0+d_{\text{safe}}}{2})^2 - (\frac{l_0-d_{\text{safe}}}{2})^2$.

6) Maximum Tension Constraint: The constraint on the maximum tension of the cable is motivated by practical considerations. We aim to prevent the risk of cable breakage during flight, which could occur if the tension exceeds the maximum limitation $f_{T,\max}$. This limitation is determined by the physical properties of the cable being used. If the maximum tension limit is a minimal value, finding feasible solutions will result in difficulty. In other words, the allowed tension in the cable will be too low to support the payload in completing the necessary extreme movement.

7) Dynamic Constraint: Since we optimize the trajectories of both the quadrotor and payload simultaneously, we need to introduce an equality constraint to ensure a reasonable dynamic relationship between them. This dynamic constraint requires that when the system is in taut mode, the acceleration of the payload is determined by the tension and the gravity acting on it. When the system is in slack mode, the relative position between the quadrotor and payload is unconstrained. Therefore, we represent the dynamic relationship of the system as $\mathbf{h}_{\text{dyn}}(t,\mathbf{c}) = \|\tilde{\mathbf{p}}(t,\mathbf{c})\| \cdot (\mathbf{a}_{\mathcal{L}}(t,\mathbf{c}) + g\mathbf{e}_z) + \|\mathbf{a}_{\mathcal{L}}(t,\mathbf{c}) + g\mathbf{e}_z\| \cdot \tilde{\mathbf{p}}(t,\mathbf{c})$.

8) Complementarity Constraints: The complementarity constraints can describe the impact caused by the switching of two motion modes. The complementarity constraints are given by

$$\begin{aligned} f_T(t,\mathbf{c}) &\geq 0 \\ l_0 - l(t,\mathbf{c}) &\geq 0 \\ h_{\text{comp}}(t,\mathbf{c}) = f_T(t,\mathbf{c})(l_0 - l(t,\mathbf{c})) &= 0. \end{aligned} \tag{20}$$

Algorithm 2: Hybrid Mode Prediction and Switching.

```
MODE SWITCH (x_0, ū, t_0)
    for k ← 1 to H do
      t ← t_0 + Δt * k
      x(t) ← f(x_0, ū, t)
      [x_L, x_Q, ẋ_L, ẋ_Q, R_Q, p, ṗ]^T ← x(t)
      l ← ||x_L − x_Q||_2
      if l < l_0
        S(k) ← 0
      else
        S(k) ← 1
      end if
    return S
```

In our case, the change in tension of the cable is described by the complementarity constraint indicating whether the cable is taut or not.

V. Hybrid Nonlinear Model Predictive Control for the Suspended Payload System

In this section, we propose an HNMPC method and effectively address the model mismatch issue, enabling agile and safe flight of the robot. We cannot ignore the influences of aerodynamic drag to track trajectories involving multiple motion modes. Therefore, we introduce a hybrid dynamics model for the air resistance (Section V-A). Next, we propose a hybrid mode prediction and switching algorithm (Section V-B). Finally, we present a detailed formulation of the HNMPC method (Section V-C).

A. Hybrid Dynamic Model With Aerodynamic Drag

In our hybrid dynamic model, the states of the payload system are written as $\mathbf{x} = [\mathbf{x}_{\mathcal{L}}, \mathbf{x}_{\mathcal{Q}}, \dot{\mathbf{x}}_{\mathcal{L}}, \dot{\mathbf{x}}_{\mathcal{Q}}, \mathbf{R}_{\mathcal{Q}}, \mathbf{p}, \dot{\mathbf{p}}]^T$, including payload and quadrotor position and velocity, quadrotor orientation, and cable direction and derivative with respect to time. The system control input is $\mathbf{u} = [f, \boldsymbol{\omega}_{\mathcal{Q}}]^T$, where f and $\boldsymbol{\omega}_{\mathcal{Q}}$ is the thrust and body rate of the quadrotor separately. Based on the Lagrange d'Alembert principle, the differential equations of this system are defined as follows:

$$\dot{\mathbf{x}}_{\mathcal{Q}} = \mathbf{v}_{\mathcal{Q}} \tag{21}$$

$$\ddot{\mathbf{x}}_{\mathcal{Q}} = S\mathbf{T}_c + \frac{f\mathbf{R}_{\mathcal{Q}}\mathbf{e}_z - k_{\mathcal{Q}}\mathbf{v}_{\mathcal{Q}}}{m_{\mathcal{Q}}} - g\mathbf{e}_z \tag{22}$$

$$\dot{\mathbf{x}}_{\mathcal{L}} = \mathbf{v}_{\mathcal{L}} = \frac{\mathrm{d}\mathbf{x}_{\mathcal{L}}}{\mathrm{d}t} \tag{23}$$

$$\ddot{\mathbf{x}}_{\mathcal{L}} = -s\mathbf{T}_c - \frac{k_{\mathcal{L}}\mathbf{v}_{\mathcal{L}}}{m_{\mathcal{L}}} - g\mathbf{e}_z \tag{24}$$

$$\dot{\mathbf{p}} = \frac{\mathrm{d}\mathbf{p}}{\mathrm{d}t} \tag{25}$$

$$\ddot{\mathbf{p}} = s\left(\mathbf{p} \times \left(\mathbf{p} \times \left(\frac{f\mathbf{R}_{\mathcal{Q}}\mathbf{e}_z + \mathbf{a}_{\mathcal{Q}}}{lm_{\mathcal{Q}}} - \frac{\mathbf{a}_{\mathcal{L}}}{lm_{\mathcal{L}}}\right)\right)\right)$$

$$- s(\dot{\mathbf{p}} \cdot \dot{\mathbf{p}})\mathbf{p} - (1-s)\left(-\frac{\mathbf{a}_{\mathcal{L}}}{m_{\mathcal{L}}} + \frac{f\mathbf{R}_{\mathcal{Q}}\mathbf{e}_z + \mathbf{a}_{\mathcal{Q}}}{m_{\mathcal{Q}}}\right) \tag{26}$$

$$\dot{\mathbf{R}}_{\mathcal{Q}} = \mathbf{R}_{\mathcal{Q}}\hat{\omega}_{\mathcal{Q}} \tag{27}$$

$$\mathbf{T}_c = \frac{\mathbf{p} \cdot (m_{\mathcal{Q}}\mathbf{a}_{\mathcal{L}} - m_{\mathcal{L}}(f\mathbf{R}_{\mathcal{Q}}\mathbf{e}_z + \mathbf{a}_{\mathcal{Q}}))\mathbf{p}}{m_{\mathcal{Q}}(m_{\mathcal{Q}} + m_{\mathcal{L}})} + \frac{m_{\mathcal{L}} l(\dot{\mathbf{p}} \cdot \dot{\mathbf{p}})\mathbf{p}}{(m_{\mathcal{Q}} + m_{\mathcal{L}})} \tag{28}$$

where the $\hat{\mathbf{v}}$ represents the skew-symmetric matrix of vector $\mathbf{v}$. The switch variable $s = 1$ corresponds to the system being in the taut mode, while $s = 0$ corresponds to the cable being slack. $\mathbf{a}_{\mathcal{Q}}$ and $\mathbf{a}_{\mathcal{L}}$ are the drag force acting on the payload and the quadrotor, respectively. It is worth noting that our HNMPC implementation code utilizes quaternions to represent the quadrotor's orientation. To improve the comprehensibility of our algorithm, we use the rotation matrix $\mathbf{R}_{\mathcal{Q}}$ to represent the orientation in the context.

To model the aerodynamic drag forces, we use the concept of linear drag formulation [39], which are proportional to the velocity of a rigid body. Specifically, the drag force acting on the robot is written as

$$\mathbf{a}_{\mathcal{L}} = -k_{\mathcal{L}}\mathbf{v}_{\mathcal{L}} \tag{29}$$

$$\mathbf{a}_{\mathcal{Q}} = -k_{\mathcal{Q}}\mathbf{v}_{\mathcal{Q}} \tag{30}$$

where $k_{\mathcal{L}}$ and $k_{\mathcal{Q}}$ are drag coefficients. $\mathbf{v}_{\mathcal{L}}$ and $\mathbf{v}_{\mathcal{Q}}$ are the velocities of the payload and the quadrotor, respectively. It is worth emphasizing that in our simulations and experiments, the robot's maximum speed does not exceed 3 m/s (see Section VI-C2). Therefore, it is reasonable to approximate the drag using a linear model. However, employing a quadratic model instead of (29), such as $\mathbf{a}_{\mathcal{L}} = -k_{\mathcal{L}}\|\mathbf{v}_{\mathcal{L}}\|^2 \frac{\mathbf{v}_{\mathcal{L}}}{\|\mathbf{v}_{\mathcal{L}}\|}$ [39], is more appropriate for the payload when tackling tasks that demand higher speeds. Equations (21)–(24) and (27) represent the joint system dynamics. $\dot{\mathbf{p}}$ and $\ddot{\mathbf{p}}$ are the first and second derivative of the direction vector $\mathbf{p}$ from the quadrotor to the payload.

The dynamics and performance of the suspended payload system can be significantly affected by aerodynamic drag forces, which depend on the system's geometric properties, motion states, and fluid properties. These forces can substantially impact the system's dynamics when the payload volume is large, or the quadrotor is flying at high speeds, causing deviations from the desired trajectory and reduced accuracy. It is essential to accurately model and incorporate the aerodynamic drag forces into the system dynamics to mitigate these effects and ensure accurate control.

B. Dual Motion Modes Prediction and Switching

In MPC, accurately predicting the system state, as shown in Fig. 6, is critical. To achieve this, we propose a hybrid mode prediction and switching algorithm based on a hybrid dynamics model that accounts for air drag (Algorithm 2). The algorithm iteratively predicts the system's motion mode at each time step within the prediction window, using the system's previous state and switch variable. The iterative calculations continue until the motion mode of the system is determined for all times within the prediction window. Finally, we minimize the objective function based on the sequence of system states to obtain the optimal control strategy.

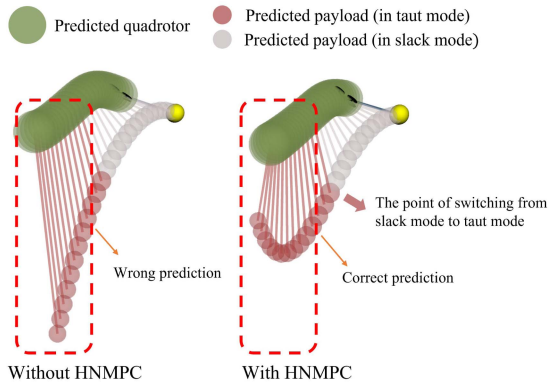


Fig. 6. Predicted quadrotor and payload position without or with hybrid mode switch, respectively, when the cable is slack. Without hybrid mode switch, the payload always obey the slack model and the distance between the quadrotor and payload l is greater than cable length l_0, which is infeasible. With hybrid mode switch, we can keep the prediction result satisfying $l \leq l_0$.

C. Hybrid Model Predictive Control Formulation

We construct a quadratic optimization problem based on traditional NMPC methods using the hybrid dynamics model and predicted motion mode. Our HNMPC is formulated as follows:

$$\min_{\mathbf{u}} \mathcal{J} = \frac{1}{2}\sum_{k=1}^{H}\left(\|\mathbf{x}(k) - \mathbf{x}_r(k)\|^2_{\mathbf{Q}_k} + \frac{1}{2}\|\mathbf{u}(k) - \mathbf{u}_r(k)\|^2_{\mathbf{R}_k}\right) + \frac{1}{2}\|\mathbf{x}(H) - \mathbf{x}_r(H)\|^2_{\mathbf{Q}_H} \tag{31a}$$

$$\text{s.t. } \mathbf{x}(k+1) = \mathrm{f}(\mathbf{x}(k), \mathbf{u}(k), S(k)) \tag{31b}$$

$$\mathbf{u}_{\min} \leqslant \mathbf{u} \leqslant \mathbf{u}_{\max} \tag{31c}$$

where H is the horizon of HNMPC and $\mathbf{x}_r(k)$ and $\mathbf{u}_r(k)$ represent reference state and control input, respectively, at k time step, which can be obtained from polynomial trajectories. In (31b), $\mathrm{f}(\mathbf{x}(k), \mathbf{u}(k), S(k))$ is the discretized version of the dynamic model. $\mathbf{x}(k)$ and $\mathbf{u}(k)$ are the system state and control input, and $S(k)$ is the switch variable obtained above. The boundary constraints of control input $\mathbf{u} = [f, \boldsymbol{\omega}_{\mathcal{Q}}]^T$ is defined as (31c) to limit the thrust f and body rate $\boldsymbol{\omega}_{\mathcal{Q}}$ of quadrotor in a physically feasible range.

Note that we apply exponential decay on the weight matrix $\mathbf{Q}$ and $\mathbf{R}$, written as (32), to improve the controller robustness and reduce the impact from the estimation noise. The $b_{\mathbf{x}}$ and $b_{\mathbf{u}}$ decide the decay speed

$$\mathbf{Q}_k = \exp\left(-\frac{k}{H}b_{\mathbf{x}}\right)\mathbf{Q},\ \mathbf{R}_k = \exp\left(-\frac{k}{H}b_{\mathbf{u}}\right)\mathbf{R}. \tag{32}$$

To ensure real-time execution of the HNMPC, we adopt a warm start-up strategy to reduce the computational load. Specifically, we derive an initial guess from the reference trajectory given by the planning module. After finding the feasible solution,

the previous results are used as the initial guess for the current control step.

By incorporating the hybrid model and motion mode predictions into the optimization problem, we can effectively solve for the optimal control strategy that accounts for the effects of air resistance and system motion modes. This approach enables us to achieve high-performance control of the system during complex flights, ensuring the safe and accurate operation of the system.

VI. Results

In this section, simulations and real-world experiments validate our proposed impact-aware planning and control methods for a quadrotor with a cable-suspended payload. First, we compare our impact-aware planning method to baseline algorithms, including the state-of-the-art (SOTA) method, and analyze the performance in terms of computational efficiency and solution accuracy (Section VI-B1). Then, we conduct ablation studies to verify that the HNMPC method can effectively reduce tracking errors (Section VI-B4). Finally, we designed two complex scenarios for the suspended payload system to perform agile flights in both simulations (Section VI-C1) and real-world (Section VI-C2).

To our knowledge, we are the first to conduct agile flying experiments with automatic mode-switching in the physical world. Videos of simulations and experiments can be found at https://sites.google.com/view/suspended-payload/.

A. System Setup

First, we describe our system setup, which comprises software and hardware. The computation and visualization of the trajectories are carried out on a laptop computer with Intel Core i7-10875H (CPU 2.3 GHz base speed), running Ubuntu 20.04 and ROS Noetic. Our method is implemented in C++11 and does not rely on any hardware acceleration. We utilize an open-source tool CasADi [40] with the Ipopt [9] solver to implement the comparison algorithms. The hardware for the suspended payload system is composed of several components (details in Fig. 16), including an NVIDIA Jetson Orin NX 16 GB as the onboard computer, a Kakute H7 autopilot hardware, a 250-mm carbon fiber frame, and a 3-D-printed payload. Table V shows the physical parameters of the robot used in this work.

Implementing our HNMPC code is based on the ACADO Toolkit [41], [42]. We utilize the real-time iteration (RTI) scheme provided by the ACADO toolkit for onboard computation. The solver that we choose is qpOASES. The HNMPC algorithm is executed at 100 Hz on the onboard computer, while the angular velocity loop operates at 400 Hz on the flight controller. The average solving time for the HNMPC on the onboard computer is approximately 6.77 ms, whereas on the laptop computer, it is around 2.65 ms.

B. Benchmark Comparisons

1) Obstacle Avoidance: We compare the efficiency and accuracy of our proposed impact-aware planning method with two algorithms to validate it. We denote the SOTA method proposed by Zeng et al. [5] as Ipopt+MP, the same method with safety distance rather than minimum-penetration constraints as Ipopt+SD. These methods construct an optimization problem containing complementarity constraints with discrete system states as decision variables and solve this optimization problem using a general-purpose solver, Ipopt. The difference between the Ipopt+SD and the Ipopt+MP is that the former uses the sign distance function as the obstacle avoidance constraint. The latter equivalently converts the sign distance function into a set of constraints by introducing dual variables. Details of this conversion method of introducing dual variables can be found in a pioneering work [43].

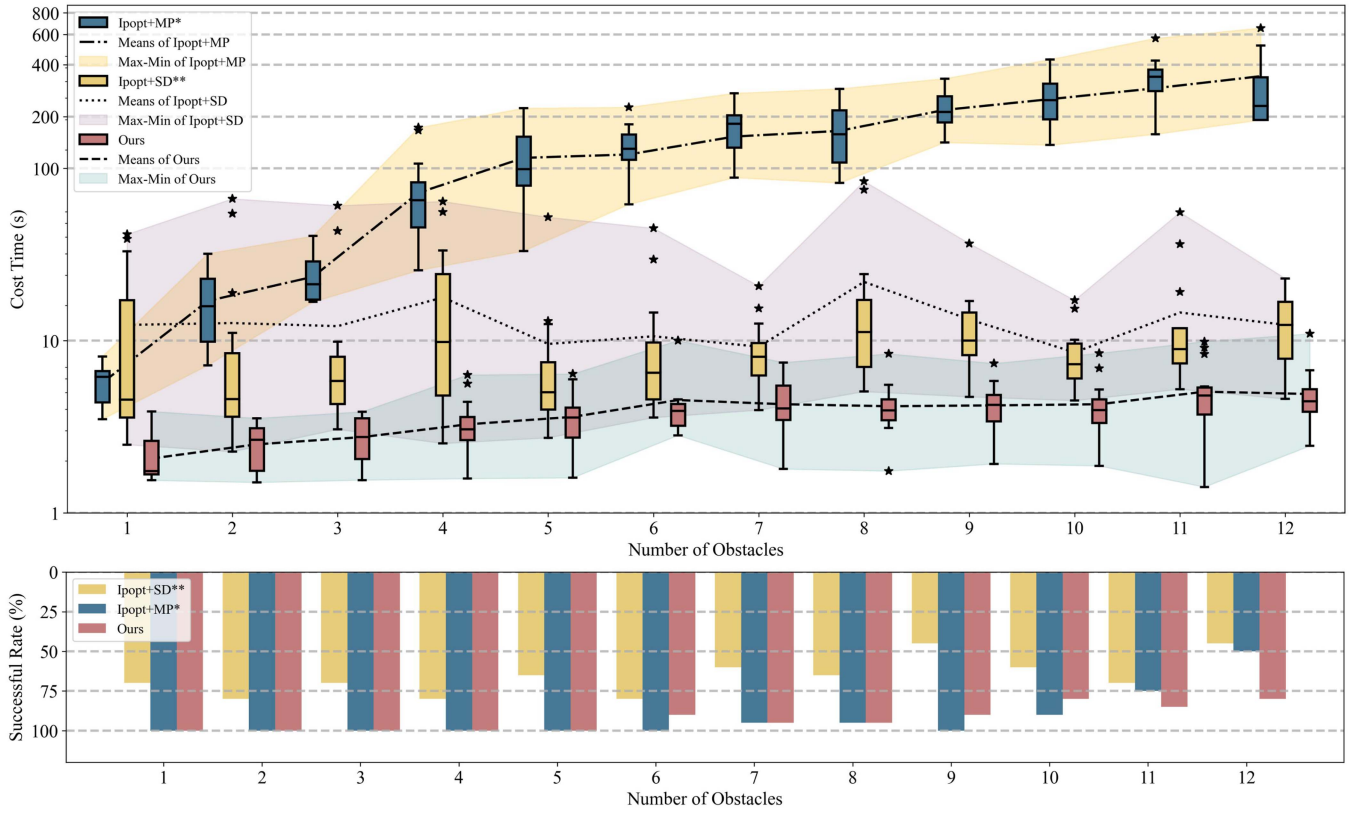
We conduct a total of 12 groups of experiments and compare the mean, maximum, and minimum time cost and the number of failed cases for each group, as shown in Fig. 7 and Table II . From the comparison of computational time, we can draw two conclusions.

1) Using polynomials instead of discrete system states as decision variables can reduce the mean time cost.
2) Utilizing robust unconstrained optimization algorithms can improve solving efficiency and yield stable results.

The first conclusion is evident because the mean time cost of both the Ipopt+MP and the Ipopt+SD is higher than the proposed method. Without considering the dual variables introduced in optimization, these two former methods use discrete system states as decision variables. To ensure the continuity of robot motion, they require a relatively dense sampling of trajectories (we set the number of sampling to 50, fewer points will cause discontinuous robot states). In contrast, our method utilizes polynomials to represent trajectories, which naturally guarantees the continuity of robot motion and significantly reduces the number of variables.

The second conclusion is based on the following observation. As shown in Fig. 7, the Ipopt-MP has better stability in solutions than the Ipopt-SD (with a smaller variance of computation time in the same group of simulations). This is because the Ipopt-MP, compared with the Ipopt-SD, converts nonsmooth safety distance constraints into smooth differentiable dual constraints. However, the number of dual variables is positively correlated with the number of obstacles, resulting in a significant increase in the average computation time of the Ipopt-MP. By utilizing robust unconstrained optimization algorithms (as detailed in the previous technical section), our proposed method achieves better stability than both baseline methods while maintaining high computational efficiency.

Comparing success rates can demonstrate the advantage of our algorithm in computation accuracy. First, as shown in both Fig. 7 and Table II, the Ipopt+SD has a significantly lower success rate than the other two methods. This is because the Ipopt+SD directly uses nonsmooth safety distance as collision avoidance constraints, which may cause the algorithm to fail to converge to a solution with the desired accuracy. Second, the Ipopt+MP utilizes smooth minimum penetration constraints to obtain accurate solutions, but its success rate significantly decreases when the number of obstacles is large. The Ipopt+MP



*Ipopt with minimum penetration constraints.
**Ipopt with safety distance constraints.

Fig. 7. Figure compares the optimization time required by three methods and the success rate for obstacle avoidance tasks. It shows 12 groups of simulations, each containing 20 random maps with varying numbers of obstacles. The blue boxplot in the top subplot represents the Ipopt+MP method proposed by J. Zeng et al. [5]. This method used Ipopt to solve the ONCC problem with minimum penetration constraints. It should be noted that the data for the Ipopt+MP were generated from the data distribution parameters provided in the original paper [5]. This is done for two reasons: first, because the method involves many optimization parameters and no open-source implementation is given, none of the results we reproduce are as good as those in the original paper; second, our method has an order of magnitude improvement in optimization time, so the differences due to the specific implementation can be disregarded. The yellow boxplot represents the Ipopt+SD method, which is the same as the Ipopt+MP method but uses safety distance constraints. This method is implemented using CasADi [40]. Moreover, the red boxplot illustrates the proposed method. The boundary of translucent color areas highlights the maximum and minimum optimization times for each method across all simulations. The bottom half of the figure shows the success rates of the three methods in different groups of simulations.

TABLE II
RESULTS OF OBSTACLE AVOIDANCE TASKS

# of obstacles		1	2	3	4	5	6	7	8	9	10	11	12
Ipopt with SD	Mean (s)	12.34	12.66	12.14	17.83	9.57	10.60	9.23	21.93	13.28	8.55	14.56	12.37
	Min (s)	2.49	2.27	3.06	2.53	2.73	3.58	3.95	5.07	4.72	4.49	5.23	4.59
	Max (s)	41.44	66.65	60.75	64.32	52.04	44.98	20.74	84.29	36.65	17.18	55.60	22.99
	Success rate (%)	70.0	80.0	70.0	80.0	65.0	80.0	60.0	65.0	45.0	60.0	70.0	45.0
Ipopt with MP	Mean (s)	5.7	17.0	23.5	72.1	115.2	120.5	153.1	165.2	218.7	252.3	292.1	342.9
	Min (s)	3.5	7.2	16.8	25.6	33.2	61.9	88.5	82.4	141.4	136.6	158.0	191.2
	Max (s)	8.1	32.0	40.5	172.3	223.5	226.4	272.8	289.9	332.0	428.9	569.1	652.7
	Success rate (%)	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**95.0**	**95.0**	**100.0**	**90.0**	75.0	50.0
Our method	Mean (s)	**2.07**	**2.50**	**2.75**	**3.27**	**3.61**	**4.12**	**4.27**	**4.16**	**4.21**	**4.27**	**5.05**	**4.90**
	Min (s)	1.54	1.50	1.55	1.58	1.60	1.77	1.80	1.75	1.93	1.88	1.42	2.44
	Max (s)	3.88	3.54	3.87	6.32	6.41	7.78	7.47	8.39	7.37	8.54	9.88	10.96
	Success rate (%)	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	90.0	**95.0**	**95.0**	90.0	80.0	**85.0**	**80.0**

The bold entities to indicate the lowest optimization time and the highest success rate.

cannot find feasible solutions because sparse sampling of discrete system states cannot guarantee the continuity of robot motion. While the Ipopt+MP's success rate significantly drops with more obstacles, our method still maintains a high success rate.

Furthermore, we can highlight the advantage of ALM over Ipopt in solving ONCC by comparing the success rates. Since both the Ipopt+SD method and our method were tested on the same maps, the main reason the Ipopt+SD method has a lower success rate is that it is more prone to stopping at nonoptimal

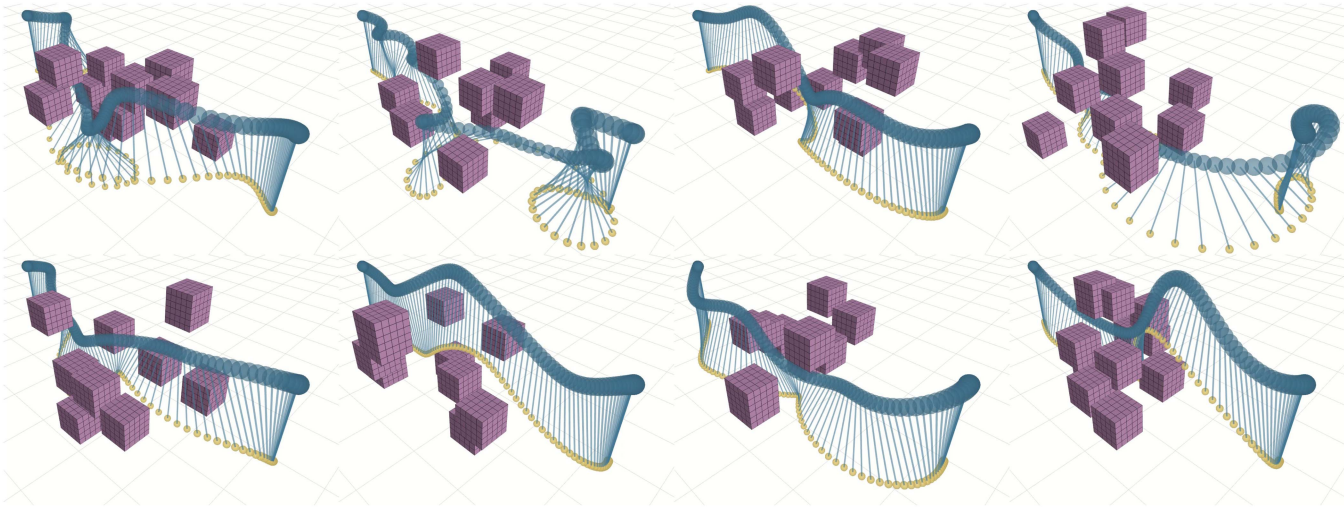


Fig. 8. Results of obstacle avoidance using our proposed trajectory planning method. Similar to crowded scenarios in previous work [5], we set the payload's starting position and target position to $\mathbf{x}_{\mathcal{L},0} = [0.0,\ -2.5,\ 1.0]^T$ and $\mathbf{x}_{\mathcal{L},T} = [0.0,\ 2.5,\ 1.0]^T$ in the world frame. The movement of the payload is limited within the space $\mathcal{P} = \{p = [x, y, z]^T, x \in [-1.5, 1.5], y \in [-1.5, 1.5], z \in [0.0, 2.0]\}$. We conducted a total of 12 groups of experiments, each with 20 trails. In each trail, obstacles were generated within the movement area of the payload $\mathcal{P}$. The obstacle was a cubic with a side length of 0.5 m. The subplots illustrate eight different scenarios with dense obstacles. Our approach can navigate the suspended payload system through crowded environments and achieve aggressive flight trajectories while avoiding collisions.

values when solving the ONCC problem. Our method also demonstrates a higher success rate in scenarios with many obstacles (for example, 11 and 12). The above results align with the findings in the classical research [7], where ALM and other solvers, including Ipopt, were tested on standard optimization test sets and reached similar conclusions.

By preserving trajectory continuity and improving solution accuracy through iterative optimization, our method can naturally generate feasible trajectories with large attitude changes in maps with dense obstacles (as shown in Fig. 8).

2) Optimization Ablation Study: We conducted ablation experiments to evaluate the contributions of different modules in our proposed trajectory planning algorithm on optimization efficiency and trajectory feasibility. We performed ablation experiments by removing specific algorithm modules or replacing certain modules with more basic algorithms in our optimization algorithm.

We focused on three critical components of our algorithm: 1) the initialization module, 2) the inner-loop optimization algorithm module, and 3) the outer-loop parameter update module. We tested the influence of the initial values by comparing warm-up strategies. For the inner-loop optimization module, we replaced the L-BFGS algorithm and the Lewis-Overton line search algorithm used in our method with the steepest gradient descent and backtracking line search algorithms, respectively, to compare the effects of different unconstrained optimization strategies on trajectory optimization. Finally, we recorded the optimization time and trajectory feasibility for different combinations of these modules under 1, 3, and 5 iterations of outer-loop parameter updates. We set a stringent convergence threshold to ensure that the outer-loop iterations were always greater than 5. We tested the trajectories optimized by different module combinations in Drake and recorded whether these trajectories were feasible. The results are presented in Table III. We took the optimization time required to complete trajectory optimization in Scene 1 using the method that includes all algorithm modules for five iterations as a benchmark. We recorded the corresponding proportion of optimization times for other different combinations of methods relative to the benchmark. In the table, "✗" and "✔" indicate whether the optimized trajectories can be executed in the simulation.

TABLE III
OPTIMIZATION EFFICIENCY AND TRAJECTORY FEASIBILITY ANALYSIS

Method	Iteration 1	Iteration 3	Iteration 5
No warm up	2.12, ✗	23.23, ✔	34.18, ✔
Backtracking line search	0.62, ✗	1.02, ✗	1.85, ✔
Steepest descent	184.37, ✗	890.75, ✗	2856.66, ✗
Our full method	0.31, ✗	0.50, ✔	1.00, ✔

Through analyzing the data, we draw several conclusions as follows.

1) An excellent initial value can significantly expedite the optimization process. Without using the warm-up strategy, although the final trajectories obtained after multiple iterations of the outer-loop optimization are feasible, the required optimization time increases significantly.
2) The method for solving the unconstrained optimization problem in the inner loop is crucial in determining both the efficiency and trajectory feasibility. Employing more efficient and effective gradient descent and line search methods can significantly enhance the optimization speed and trajectory feasibility.

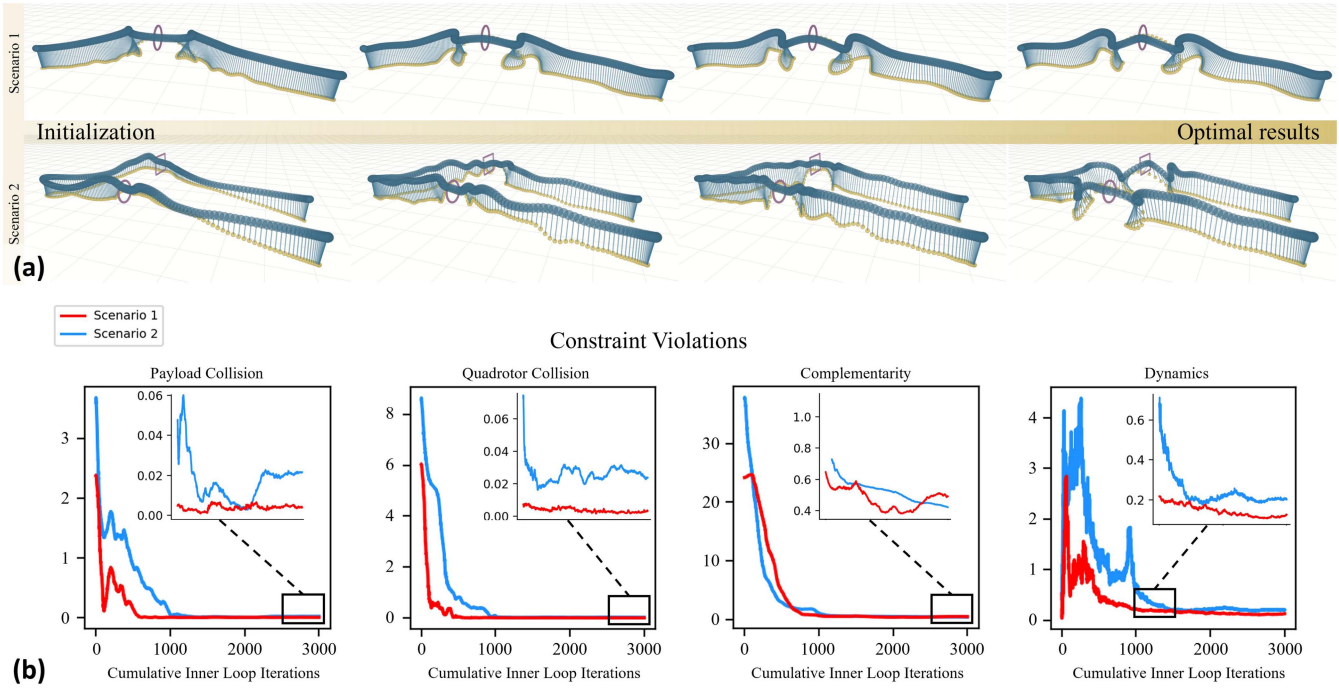


Fig. 9. Visualization of optimized trajectories in two complex scenarios. (a) Trajectory shapes during the optimization process. The first row displays the trajectory evolution in scenario 1, from initialization (left) to the optimal results (right), with two intermediate optimization states visualized. The second row shows the trajectory evolution in scenario 2. (b) Changes in four essential constraint violations during trajectory optimization. From left to right, the subplots represent the payload collision constraint, quadrotor collision constraint, complementarity constraint, and dynamical constraint. The red curves represent the constraint violation values in scenario 1, while the blue curves represent those in scenario 2. The constraint violation values near the end of the optimization process are magnified to show that all constraints are well satisfied, indicating the high precision of the optimization results.

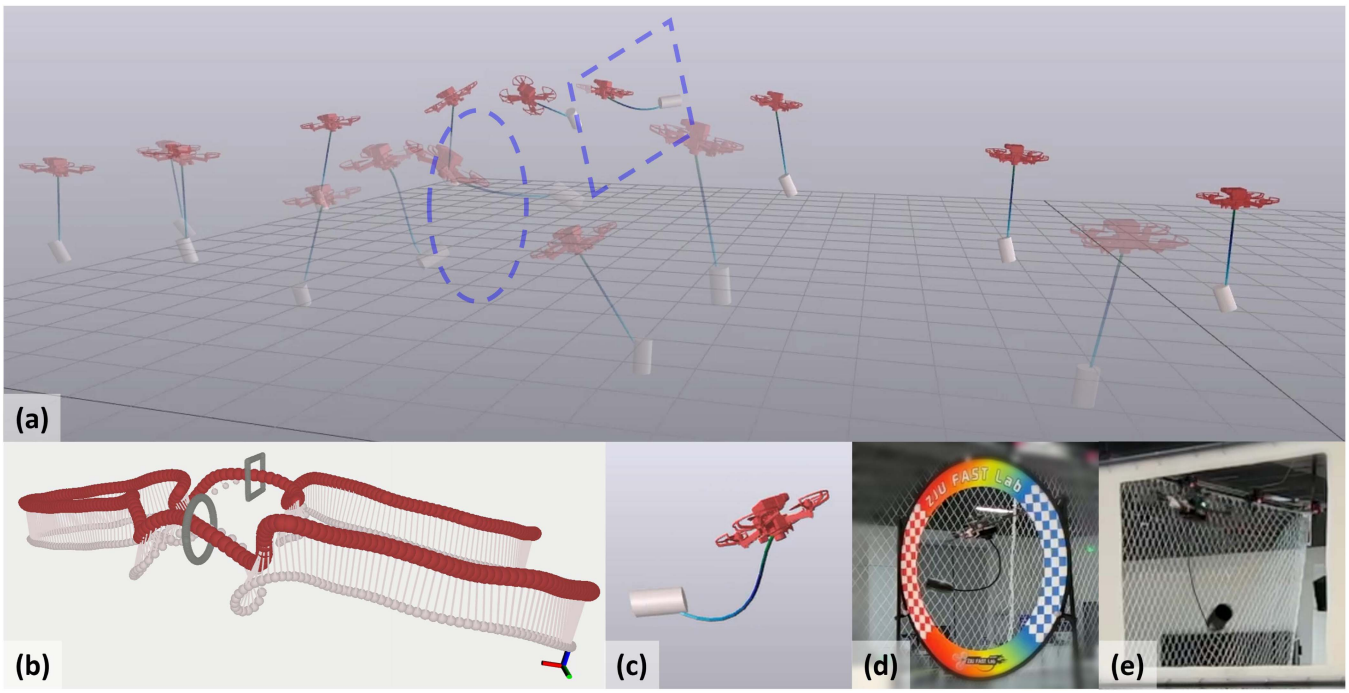


Fig. 10. Simulation of suspended payload system motion in scenario 2 using Drake. (a) Multiple system states at different time steps overlaid in a single scene, with transparency increasing from light to dark to represent the chronological sequence. The suspended payload system is in motion, demonstrating its dual modes under different cable tension conditions. The planned trajectories for the complex scenario 2 are shown in (b). We captured the robot passing through a ring gate in the trajectory and included snapshots of the simulation (c) and actual flight (d). The two snapshots show that the simulation results are similar to the actual results. In addition, we captured the actual effects of the suspended payload system passing through two gates on the trajectory, as shown in (d) and (e). The snapshots show that the robot is in a slack mode of motion.

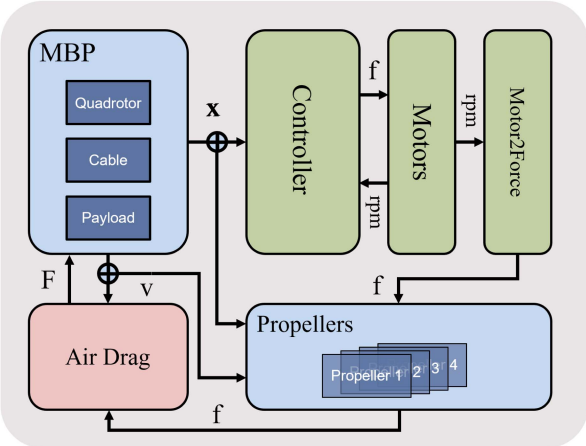


Fig. 11. Simulation diagram of the suspended payload system, which illustrates the relationship between different modules. The multi-body plant (MBP) module, consisting of a quadrotor, cable, and payload, is shown in the left-top part. The interfaces between modules are designed to be consistent with the actual robot interfaces, ensuring the fidelity of the simulation.

TABLE IV
COMPARISON OF CONTROL EFFECTS WITH AND WITHOUT HNMPC

Scenario	Tracking Error	w/o HNMPC	w HNMPC	Error Reduction (%)
#1	RMSE (m)	0.051	**0.046**	9.8 ↓
	Max (m)	0.121	**0.083**	31.4 ↓
	RMSE-XY (m)	0.040	**0.034**	15.0 ↓
	Max-XY (m)	0.112	**0.079**	29.5 ↓
#2	RMSE (m)	0.053	**0.048**	9.4 ↓
	Max (m)	0.164	**0.105**	36.0 ↓
	RMSE-XY (m)	0.037	**0.034**	8.1 ↓
	Max-XY (m)	0.114	**0.083**	27.2 ↓

The bold entities highlight indicators with more minor tracking errors.

TABLE V
CONFIGURATIONS OF SUSPENDED PAYLOAD SYSTEM

Notation	Value	Description
$m_{\mathcal{Q}}$	0.746 kg	Mass of the quadrotor
$m_{\mathcal{L}}$	0.054 kg	Mass of the payload
l_0	0.644 m	Length of cable
$r_{\mathcal{Q}}$	0.2 m	Safe margin radius of the quadrotor
$r_{\mathcal{L}}$	0.2 m	Safe margin radius of the payload

3) The multiple iterations of outer-loop parameter updates in ALM are essential in improving trajectory feasibility. Each update of ALM parameters leads to fewer constraint violations in the trajectory, thereby improving trajectory feasibility.

3) HNMPC Robustness Exploring: To explore the robustness of our proposed HNMPC method, we tested our controller under different cable lengths and payload masses. The experimental procedure is as follows: We set the system in a similar slack mode state as the initial state. During the simulation, the payload can freely fall under gravity, gradually straightening the cable. The target position for the quadrotor and its initial position are both set at the origin $(0, 0, 0)$ of the world frame. In contrast, the target position for the payload is at $(0, 0, -l_0)$, which is directly below the origin with a height difference equal to the cable length l_0. We record the states of the quadrotor and payload over a sufficiently long period and calculate their respective tracking errors and convergence times. The convergence time t_{cov} is the smallest value of t such that the tracking error remains below the threshold Δ for all times greater than t.

We tested 12 combinations formed by pairing four different payload masses (50, 100, 150, and 200 g) with three different cable lengths (50, 60, and 70 cm), and each combination was tested three times. The results are summarized in Figs. 12 and 13. Fig. 12 displays the tracking error curves for the different combinations, with a red horizontal dashed line indicating the threshold used for calculating the convergence time. We group the tracking error curves for the same cable length in each column subplot. For example, the first column subplot visualizes the tracking error curves for the combination with a cable length of 0.7, showing the tracking errors of the payload and quadrotor. In addition, we recorded and visualized the mean and standard deviation of the tracking error convergence times for different systems in Fig. 13.

Based on the data of tracking errors and their convergence times, we draw three conclusions as follows.

1) Under the same cable length, the convergence time of the controller has almost no difference, even with different payload masses.
2) The convergence time of the controller shows a very slight increase (on the order of hundreds of milliseconds) as the cable length increases.
3) Even when velocity discontinuities result from collision behavior, the controller can guide the system to converge to a stable state. The convergence time of the tracking error is hardly affected, even in situations with larger payload masses and more pronounced collision behavior.

4) HNMPC Evaluation in Simulations: To evaluate our proposed HNMPC method, we test control effects in two complex scenarios with and without the HNMPC method. In each scenario, we repeated an agile flight simulation five times with each of the two control strategies. Table IV shows the tracking errors' numerical results.

We notice a significant decrease in the maximum trajectory tracking error, 31.4% in scenario 1 and 36.0% in scenario 2, after using the hybrid predictive control method. Although the motion mode switch occurs in a small percentage of the overall flight time (see Fig. 15), the decrease in the track's root-mean-square error (RMSE) is still up to 9.8% and 9.4% in two scenarios, respectively. In addition, we also compared the RMSE and maximum value of tracking errors in the *X*-*Y* plane between the two control strategies. The value in Table IV shows that the HNMPC method significantly contributes to reducing the tracking errors on the *X*-*Y* plane. This indicates that the mode switching on the agile flight occurs in 3-D space rather than simply accelerating or decelerating in the *Z*-axis. More detailed procedures can be found in the supplementary video.[2]

[2][Online]. Available: https://youtu.be/6jo6N0TMt0o

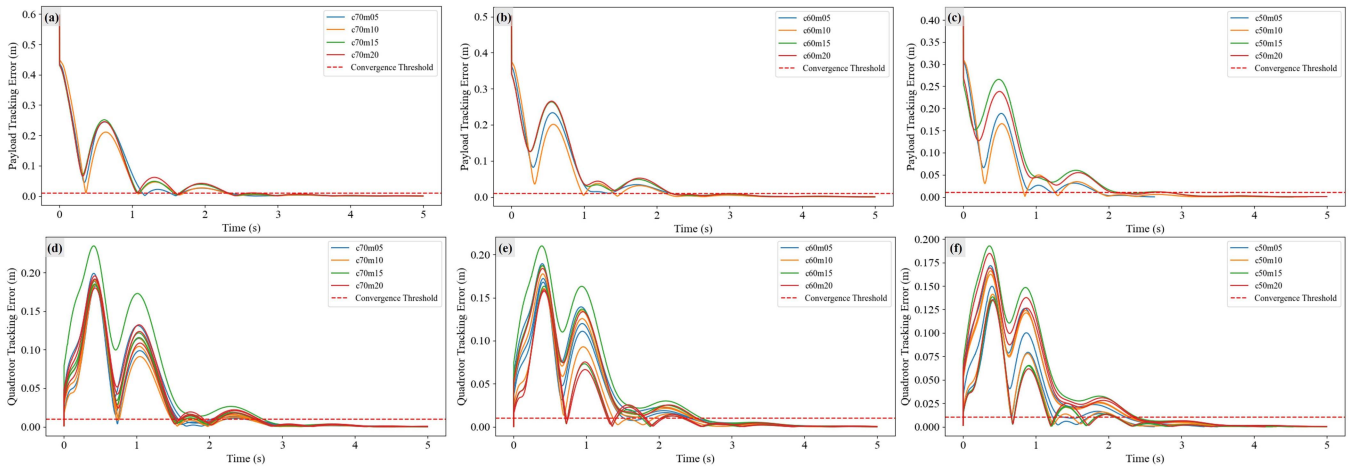


Fig. 12. Tracking error of different combinations of cable lengths and payload masses. The top row shows the tracking error for the payload, while the bottom row displays the tracking error for the quadrotor. The left column corresponds to a cable length of 70 cm, the middle to 60 cm, and the right to 50 cm. Each subplot represents the tracking error curves for different payload weights with the same rope length, depicted in different colors. The red dashed horizontal line represents the convergence threshold, set at 0.01 cm in our experiments.

C. Flying Through Narrow Gates

We design two complex scenarios to verify that our planning and control algorithms can ensure: 1) natural mode switching can occur when the robot passes through narrow corridors; 2) the robot can complete long-distance movements while retaining the ability of natural mode switching. These two key points are critical to ensuring the practical application of the suspended payload system.

1) Agile Flight Simulation: We generate point cloud maps based on the scenario layouts and test our planning algorithm within them. We visualize several iteration results in the algorithm process, as shown in Fig. 9(a). We can observe that the method can obtain a feasible trajectory after a limited number of iterations (the computational time required for the optimization to converge within 1 min) from an initially completely infeasible trajectory. In Fig. 9(b), we present four essential constraints that are prone to violations (the ordinate represents the sum of the absolute constraint violation values on the trajectory). We can clearly see that all constraint values are low when the trajectory approaches the optimum. This indicates that our optimization algorithm can provide sufficiently accurate solutions within a short computation time, ensuring the planned flight is feasible.

A simulation environment that closely resembles reality can significantly accelerate the process of deploying our methods in practical applications. We build a diagram of the suspended payload system using Drake [44], as shown in Fig. 10. In order to be consistent with the actual situation, we implement different leaf systems and keep each module's interface consistent with the actual robot's interface, as shown in Fig. 11. Specifically, we simulate four motor signal inputs with first-order response functions and the thrust model of individual motors with propeller modules. We add an air drag module to simulate the velocity-dependent resistance on the rigid body. Detailed demonstrations can be found in our supplementary videos.[3,4] The implementation of the simulation system will also be included in our open-source code.

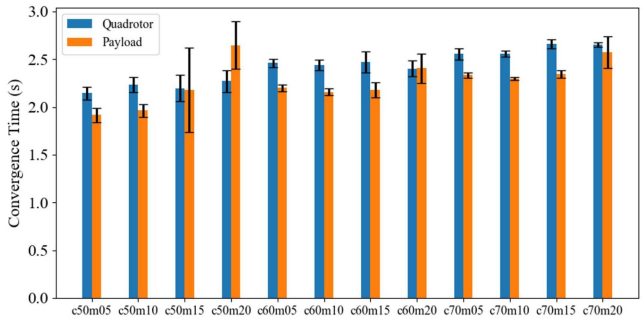


Fig. 13. Convergence time for stable hovering of different combinations of cable lengths and payload masses. In the figure, we visualize the convergence time for stable hovering at the target position across 12 combinations of three cable lengths and four payload masses. The blue bars represent the quadrotor's convergence time, while the orange bars represent the payload's convergence time. The x-axis corresponds to these different combinations, and the y-axis represents the convergence time in seconds.

2) Real-World Deployment: We further validate the feasibility of our method in the real world, as shown in Fig. 14. We reproduce the two complex scenarios mentioned above and test whether the robot system could complete the expected agile flights. The specific configuration of the robot hardware is mentioned in Fig. 16. In addition, we used a high-precision motion capture system to obtain real-time motion states of the quadrotor and payload.

We recorded the motion data of the robot in two scenarios. In Scenario 1, the average speed and acceleration of the robot are 0.78 m/s and 2.74 m/s^2, respectively, with maximum speed and acceleration of 2.32 m/s and 10.42 m/s^2, respectively. In scenario 2, the average speed and acceleration of the robot are 0.93 m/s and 2.55 m/s^2, respectively, with maximum speed and acceleration of 2.57 m/s and 15.52 m/s^2, respectively. We find that the overall speed of the robot is not high. However, the robot

[3][Online]: Available: https://youtu.be/wgSglmOJz0Y
[4][Online]: Available: https://youtu.be/mPx3neVR-Bc

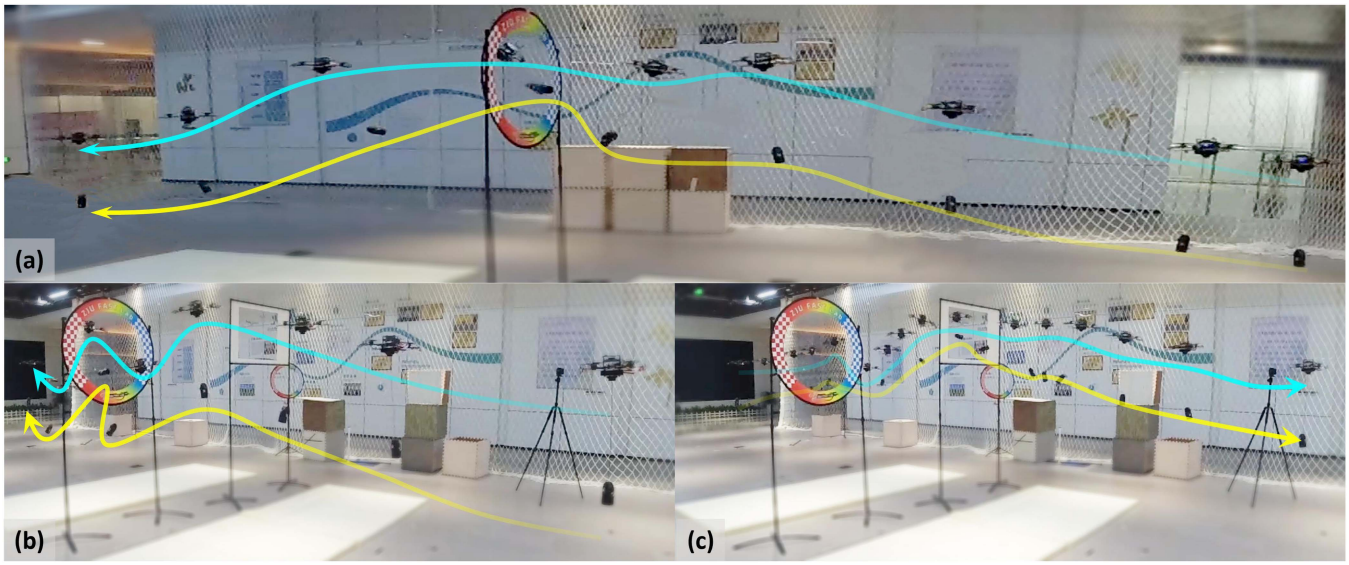


Fig. 14. Screenshots of our robot navigating through two complex scenarios. In the first scenario, the robot travels from right to left through a narrow ring-shaped gate. In the second scenario, the robot moves to the trajectory's center by passing through the ring gate and then crosses the rectangular gate to reach the end. The screenshots demonstrate the effectiveness of our trajectory planning method in handling complex environments and navigating through narrow passages. In addition, they showcase the real-time performance of our methods.

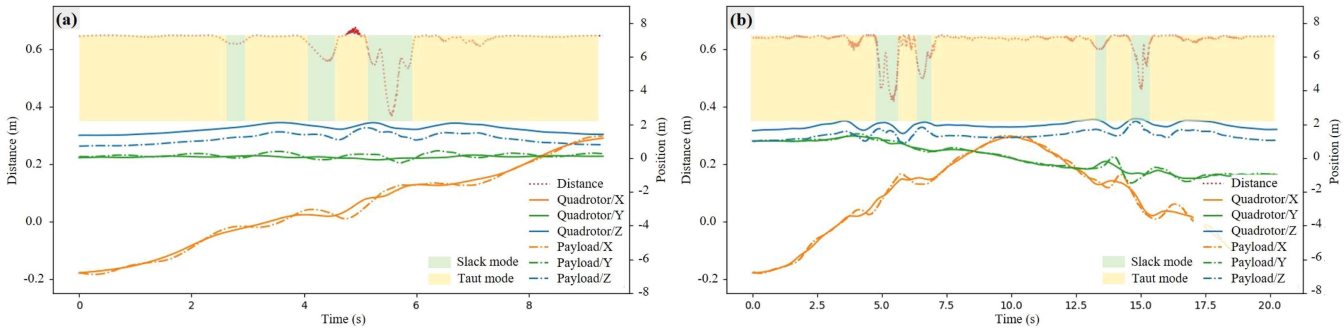


Fig. 15. System trajectories and motion modes in scenarios 1 (left) and 2 (right). (a) Three types of information: solid and dotted lines represent the position of the quadrotor and the payload along the x, y, and z axes, respectively. The dashed line depicts the distance changes between the quadrotor and the payload. The time intervals corresponding to the taut and slack modes are marked in shades of orange and light green, respectively. (b) Corresponding information for scenario 2.

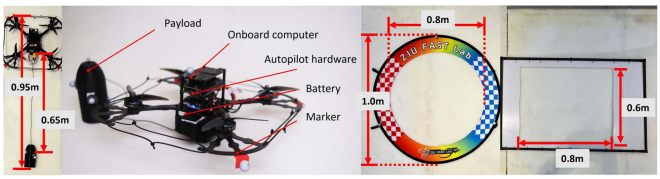


Fig. 16. Suspended payload system's structure in the upper part and two narrow passageways used in complex scenarios in the lower part. Notably, the suspended payload system's size with the cable under tension exceeds the maximum diameters of both gates, rendering the passage difficult.

needs considerable acceleration to reduce the tracking error due to the switching of motion modes.

It should be noted that the HNMPC method proposed by us is crucial for the robot to complete flights. Although using the classical NMPC method in simulation causes significant control errors, the simulated robot can still barely complete the task. In actual experiments, the control errors cannot be ignored and will cause the robot to crash. We conduct multiple flight tests, and the suspended payload system is able to complete the tasks well. This indicates that our proposed HNMPC method produces more minor control errors, ensuring feasible agile flights.

VII. Conclusion

A. Limitations and Extensions

Despite our efforts to construct a compact optimization model and use appropriate optimization methods to obtain the optimal trajectories, our work has some limitations. Specifically, we do not consider the challenge of using limited onboard sensors for environment perception and robot state estimation during high-speed flight. In this study, we relied on a high-precision motion capture system for robot localization, which may not be

feasible in outdoor environments. We will step toward the fully autonomous suspended payload system in future work.

B. Conclusion

In this work, we stress consistency and compactness in an agile flight of a suspended payload system. Based on these two essential features, an impact-aware planning algorithm is proposed to solve an optimization problem with nonlinear complementarity constraints. An impact-aware control method is proposed to address the natural model mismatch issue in flight with hybrid modes. Our methods systematically solve the problem of generating feasible and practical agile flight with hybrid motion modes for the suspended payload system. Adequate simulations and experiments validate the feasibility and effectiveness of our proposed methods.

Appendix A
System Dynamics

In this Appendix, we illustrate how to compute the system's state using the lat output, taking angular velocity as an example.

When the system is in slack mode, (8) is equivalent to [34, (3)]. In this case, we can directly follow the steps outlined in [34, Sec. III] to compute the quadrotor's angular velocity. When the system is in taut mode, according to [1, Lemma 1], we can compute a quantity $f_T \cdot \mathbf{p}$ using the following equation:

$$f_T \cdot \mathbf{p} = m_{\mathcal{L}}(\ddot{\mathbf{x}}_{\mathcal{L}} + g\mathbf{e}_z).$$

Then, based on (5) and (7), we can calculate the quadrotor's position and its higher order derivatives. At this point, subsequent steps to compute the required system state are the same as the steps in [34].

Here, we provide the formula to calculate the angular velocity directly for readers' reference. The body rate given the flat output is calculated by

$$\begin{aligned}
\boldsymbol{\omega}(0) &= \dot{\mathbf{z}}_{\mathcal{B}}(0) \cdot \sin(\psi) - \dot{\mathbf{z}}_{\mathcal{B}}(1) \cdot \cos(\psi) - \\
&\quad \dot{\mathbf{z}}_{\mathcal{B}}(2) \cdot \frac{\mathbf{z}_{\mathcal{B}}(0) \cdot \sin(\psi) - \mathbf{z}_{\mathcal{B}}(1) \cdot \cos(\psi)}{1 + \mathbf{z}_{\mathcal{B}}(2)} \\
\boldsymbol{\omega}(1) &= \dot{\mathbf{z}}_{\mathcal{B}}(0) \cdot \cos(\psi) + \dot{\mathbf{z}}_{\mathcal{B}}(1) \cdot \sin(\psi) - \\
&\quad \dot{\mathbf{z}}_{\mathcal{B}}(2) \cdot \frac{\mathbf{z}_{\mathcal{B}}(0) \cdot \cos(\psi) + \mathbf{z}_{\mathcal{B}}(1) \cdot \sin(\psi)}{1 + \mathbf{z}_{\mathcal{B}}(2)} \\
\boldsymbol{\omega}(2) &= \dot{\mathbf{z}}_{\mathcal{B}}(2) \cdot \dot{\mathbf{z}}_{B}(1) - \frac{\mathbf{z}_{B}(1) \cdot \dot{\mathbf{z}}_{\mathcal{B}}(2)}{1 + \mathbf{z}_{\mathcal{B}}(3)} + \dot{\psi}
\end{aligned}$$

where $\dot{\mathbf{z}}_{\mathcal{B}}$ is the first derivative of the z-axis of the quadrotor given by

$$\dot{\mathbf{z}}_{\mathcal{B}} = \frac{1}{\|m_{\mathcal{Q}}(\ddot{\mathbf{x}}_{\mathcal{Q}} + g\mathbf{e}_z) - f_T\mathbf{p}\|} \cdot (m_{\mathcal{Q}}\dddot{\mathbf{x}}_{\mathcal{Q}} + m_{\mathcal{L}}\dddot{\mathbf{x}}_{\mathcal{L}}) \cdot \left(\mathbf{I} - \frac{(m_{\mathcal{Q}}(\ddot{\mathbf{x}}_{\mathcal{Q}} + g\mathbf{e}_z) - f_T\mathbf{p})(m_{\mathcal{Q}}(\ddot{\mathbf{x}}_{\mathcal{Q}} + g\mathbf{e}_z) - f_T\mathbf{p})^T}{(m_{\mathcal{Q}}(\ddot{\mathbf{x}}_{\mathcal{Q}} + g\mathbf{e}_z) - f_T\mathbf{p})^T(m_{\mathcal{Q}}(\ddot{\mathbf{x}}_{\mathcal{Q}} + g\mathbf{e}_z) - f_T\mathbf{p})}\right).$$

It should be noted that the value of $\dot{\mathbf{z}}_B$ is obtained based on the third derivatives of $\mathbf{x}_{\mathcal{Q}}$ and $\mathbf{x}_{\mathcal{L}}$. Therefore, to ensure the smoothness of angle rates, it is necessary to ensure that the $\ddot{\mathbf{x}}_{\mathcal{Q}}$ and $\ddot{\mathbf{x}}_{\mathcal{L}}$ are smooth. In our implementation, we employ a seventh-order polynomial function to represent the trajectories of $\mathbf{x}_{\mathcal{Q}}$ and $\mathbf{x}_{\mathcal{L}}$.

Appendix B
Practical ALM Formation

The practical ALM formation is given by

$$L_{\rho^k}\left(\mathbf{x}^k, \bar{\lambda}^k\right) = w_f f\left(\mathbf{x}^k\right) + \sum_{i \in \mathcal{E} \cup \mathcal{I}_k} P_{\rho^k}\left(w_{c_i} c_i\left(\mathbf{x}^k\right), \bar{\lambda}_i^k\right) \tag{33}$$

where

$$P_{\rho^k}\left(w_{c_i} c_i\left(\mathbf{x}^k\right), \bar{\lambda}_i^k\right) = \begin{cases} w_{c_i} c_i\left(\mathbf{x}^k\right)\left(\bar{\lambda}_i^k + \frac{1}{2}\rho w_{c_i} c_i\left(\mathbf{x}^k\right)\right), & \text{if } c_i \in \mathcal{H} \cup \mathcal{G}_k \\ -\frac{1}{2}\left(\bar{\lambda}_i^k\right)^2 / \rho^k, & \text{otherwise.} \end{cases} \tag{34}$$

Scaling objective function and constraints

$$w_f = \frac{1}{\max\left\{1, \|\nabla f\left(\mathbf{x}^0\right)\|_\infty\right\}}$$

$$w_{c_j} = \frac{1}{\max\left\{1, \|\nabla c_j\left(\mathbf{x}^0\right)\|_\infty\right\}},\ j = 1, \ldots, m.$$

Dual variables update

$$\lambda_j^{k+1} = \begin{cases} \bar{\lambda}_j^k + \rho_k w_{c_j} c_j\left(\mathbf{x}^k\right) & \text{if } j \in E \text{ or } \rho_k w_{c_j} c_j\left(\mathbf{x}^k\right) \geq \bar{\lambda}_j^k \\ 0 & \text{otherwise.} \end{cases}$$

References

[1] K. Sreenath, N. Michael, and V. Kumar, "Trajectory generation and control of a quadrotor with a cable-suspended load - A differentially-flat hybrid system," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2013, pp. 4888–4895.

[2] 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.

[3] 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.

[4] P. Foehn, D. Falanga, N. Kuppuswamy, R. Tedrake, and D. Scaramuzza, "Fast trajectory optimization for agile quadrotor maneuvers with a cable-suspended payload," in *Proc. Robot.: Sci. Syst. XIII, Massachusetts Inst. Technol.*, Cambridge, MA, USA, 2017, pp. 1–10.

[5] 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.

[6] T. A. Howell, K. Tracy, S. Le Cleac'h, and Z. Manchester, "CALIPSO: A differentiable solver for trajectory optimization with conic and complementarity constraints," in *Proc. Int. Symp. Robot. Res.*, 2023, pp. 504–521.

[7] A. F. Izmailov, M. V. Solodov, and E. Uskov, "Global convergence of augmented Lagrangian methods applied to optimization problems with degenerate constraints, including problems with complementarity constraints," *SIAM J. Optim.*, vol. 22, no. 4, pp. 1579–1606, 2012.

[8] P. E. Gill, W. Murray, and M. A. Saunders, "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM Rev.*, vol. 47, no. 1, pp. 99–131, 2005.

[9] A. Wächter 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, 2006.

[10] M. R. Hestenes, "Multiplier and gradient methods," *J. Optim. Theory Appl.*, vol. 4, no. 5, pp. 303–320, 1969.

[11] M. J. Powell, "A method for nonlinear constraints in minimization problems," *Optimization*, vol. 14, pp. 283–298, 1969.

[12] R. T. Rockafellar, "Augmented Lagrange multiplier functions and duality in nonconvex programming," *SIAM J. Control*, vol. 12, no. 2, pp. 268–285, 1974.
[13] A. S. Lewis and M. L. Overton, "Nonsmooth optimization via quasi-newton methods," *Math. Program.*, vol. 141, no. 1, pp. 135–163, 2013.
[14] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Math. Program.*, vol. 45, no. 1, pp. 503–528, 1989.
[15] G. Li, A. Tunchez, and G. Loianno, "PCMPC: Perception-constrained model predictive control for quadrotors with suspended loads using a single camera and IMU," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 2012–2018.
[16] 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. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 5665–5671.
[17] 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.
[18] A. Romero, S. Sun, P. Foehn, and D. Scaramuzza, "Model predictive contouring control for time-optimal quadrotor flight," *IEEE Trans. Robot.*, vol. 38, no. 6, pp. 3340–3356, Dec. 2022.
[19] Y. Song and D. Scaramuzza, "Policy search for model predictive control with application to agile drone flight," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2114–2130, Aug. 2022.
[20] P. Drews, G. Williams, B. Goldfain, E. A. Theodorou, and J. M. Rehg, "Aggressive deep driving: Combining convolutional neural networks and model predictive control," in *Proc. Conf. Robot Learn.*, 2017, pp. 133–142.
[21] M. Posa, C. Cantu, and R. Tedrake, "A direct method for trajectory optimization of rigid bodies through contact," *Int. J. Robot. Res.*, vol. 33, no. 1, pp. 69–81, 2014.
[22] A. Aydinoglu, V. M. Preciado, and M. Posa, "Contact-aware controller design for complementarity systems," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 1525–1531.
[23] A. Aydinoglu, P. Sieg, V. M. Preciado, and M. Posa, "Stabilization of complementarity systems via contact-aware controllers," *IEEE Trans. Robot.*, vol. 38, no. 3, pp. 1735–1754, Jun. 2022.
[24] M. Posa, S. Kuindersma, and R. Tedrake, "Optimization and stabilization of trajectories for constrained dynamical systems," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 1366–1373.
[25] E. G. Birgin and J. M. Martínez, *Practical Augmented Lagrangian Methods for Constrained Optimization*. Philadelphia, PA, USA: SIAM, 2014.
[26] D. Fernández and M. V. Solodov, "Local convergence of exact and inexact augmented Lagrangian methods under the second-order sufficient optimality condition," *SIAM J. Optim.*, vol. 22, no. 2, pp. 384–407, 2012.
[27] 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.
[28] M. E. Guerrero, D. A. Mercado, R. Lozano, and C. D. García, "Passivity based control for a quadrotor UAV transporting a cable-suspended payload with minimum swing," in *Proc. IEEE 54th Conf. Decis. Control*, 2015, pp. 6718–6723.
[29] M. E. Guerrero-Sánchez, D. A. Mercado-Ravell, R. Lozano, and C. D. García-Beltrán, "Swing-attenuation for a quadrotor transporting a cable-suspended payload," *ISA Trans.*, vol. 68, pp. 433–449, 2017.
[30] I. Palunko, R. Fierro, and P. Cruz, "Trajectory generation for swing-free maneuvers of a quadrotor with suspended payload: A dynamic programming approach," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2012, pp. 2691–2697.
[31] F. A. Goodarzi, D. Lee, and T. Lee, "Geometric stabilization of a quadrotor UAV with a payload connected by flexible cable," in *Proc. Amer. Control Conf.*, 2014, pp. 4925–4930.
[32] C. De Crousaz, F. Farshidian, and J. Buchli, "Aggressive optimal control for agile flight with a slung load," in *Proc. Workshop Mach. Learn. Plan.Control Robot Motion*, 2014, Art. no. 7.
[33] C. de Crousaz, F. Farshidian, M. Neunert, and J. Buchli, "Unified motion control for dynamic quadrotor maneuvers demonstrated on slung load and rotor failure tasks," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2015, pp. 2223–2229.
[34] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2011, pp. 2520–2525.
[35] Y.-H. Dai and C.-X. Kou, "A nonlinear conjugate gradient algorithm with an optimal property and an improved Wolfe line search," *SIAM J. Optim.*, vol. 23, no. 1, pp. 296–320, 2013.
[36] A. Alhawarat and Z. Salleh, "Modification of nonlinear conjugate gradient method with weak Wolfe–Powell line search," in *Proc. Abstr. Appl. Anal.*, 2017, pp. 1–6.
[37] M. P. Kelly, "Transcription methods for trajectory optimization: A beginners tutorial," 2017, *arXiv:1707.00284*.
[38] Z. Wang, C. Xu, and F. Gao, "Robust trajectory planning for spatial-temporal multi-drone coordination in large scenes," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 12182–12188.
[39] N. D. Potdar, G. C. H. E. de Croon, and J. Alonso-Mora, "Online trajectory planning and control of a MAV payload system in dynamic environments," *Auton. Robots*, vol. 44, no. 6, pp. 1065–1089, 2020.
[40] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADI: A software framework for nonlinear optimization and optimal control," *Math. Program. Comput.*, vol. 11, pp. 1–36, 2019.
[41] B. Houska, H. Ferreau, and M. Diehl, "ACADO toolkit–An open source framework for automatic control and dynamic optimization," *Optimal Control Appl. Methods*, vol. 32, pp. 298–312, 2011.
[42] B. Houska, H. J. Ferreau, and M. Diehl, "An auto-generated real-time iteration algorithm for nonlinear MPC in the microsecond range," *Automatica*, vol. 47, no. 10, pp. 2279–2285, 2011.
[43] X. Zhang, A. Liniger, and F. Borrelli, "Optimization-based collision avoidance," *IEEE Trans. Control Syst. Technol.*, vol. 29, no. 3, pp. 972–983, May 2020.
[44] R. Tedrake, Drake development the team, "Drake: Model-based design and verification for robotics," 2019. [Online]. Available: https://drake.mit.edu

Haokun Wang (Graduate Student Member, IEEE) received the B.Eng. degree in computer science and engineering from the Southern University of Science and Technology, Shenzhen, China, in 2019. He is currently working toward the Ph.D. degree in robotics and autonomous systems with the Aerial Robotics Group, the Hong Kong University of Science and Technology, Hong Kong, China.

His research interests include motion planning and control for robotic systems.

Haojia Li (Graduate Student Member, IEEE) received the B.Eng. degree in robotics engineering from Northeastern University, Shenyang, China, in 2021. He is currently working toward the Ph.D. degree in electronic and computer engineering with Aerial Robotics Group, Hong Kong University of Science and Technology, Hong Kong, China.

His research interests include motion planning, control, and autonomous navigation for mobile robots.

Boyu Zhou received the B.Eng. degree in mechanical engineering from Shanghai Jiao Tong University, Shanghai, China, in 2018, and the Ph.D. degree in electronic and computer engineering with the Hong Kong University of Science and Technology, Hong Kong, in 2022.

He is currently an Assistant Professor with the School of Artificial Intelligence, Sun Yat-sen University, Guangzhou, China, where he directs the Smart Autonomous Robotics Group. His research interests include aerial robots, autonomous navigation, motion planning, 3-D reconstruction, exploration, and swarm.

Fei Gao (Member, IEEE) received the Ph.D. degree in electronic and computer engineering from the Hong Kong University of Science and Technology, Hong Kong, in 2019.

He is currently a tenured Associate Professor with the Department of Control Science and Engineering, Zhejiang University, Hangzhou, China, where he leads the Flying Autonomous Robotics group affiliated with the Field Autonomous System and Computing Laboratory. His research interests include aerial robots, autonomous navigation, motion planning, optimization, and localization and mapping.

Shaojie Shen received the B.Eng. degree in electronic engineering from the Hong Kong University of Science and Technology (HKUST), Hong Kong, in 2009, the M.S. degree in robotics and the Ph.D. degree in electrical and systems engineering from the University of Pennsylvania, Philadelphia, PA, USA, in 2011 and 2014, respectively.

He joined the Department of Electronic and Computer Engineering, HKUST, in 2014, as an Assistant Professor, and was promoted to Associate Professor in 2020. His research interests include robotics and unmanned aerial vehicles, with focus on state estimation, sensor fusion, computer vision, localization and mapping, and autonomous navigation in complex environments.