SAPIENZA
UNIVERSITÀ DI ROMA

Nonlinear predictive control of the continuum and hybrid dynamics of a suspended deformable cable for aerial pick and place

Department of Computer, Control, and Management Engineering "Antonio Ruberti"
Master of Science in Control Engineering

Antonio Rapuano
ID number: 2044902

Supervisor:
Prof. Antonio Franchi

Co-Supervisors:
Dr. Yaolei Shen
Dr. Federico Califano
Dr. Chiara Gabellieri

Academic Year: 2024-2025

Thesis defended on 18 July 2025
in front of a Board of Examiners composed of:

Prof. Alessandro De Luca (chairman)
Prof. Andrea Cristofaro
Prof. Leonardo Lanari
Prof. Domenico Lembo
Prof. Francesco Liberati
Prof. Mattia Mattioni
Prof. Marilena Vendittelli

Nonlinear predictive control of the continuum and hybrid dynamics of a suspended deformable cable for aerial pick and place
Master's thesis. Sapienza University of Rome.

This thesis has been typeset by LaTeX and the Sapthesis class.

Author's email: a.rapuano1999@gmail.com

To those who always believed in me.
To my family, who never stopped.

Acknowledgments

First and foremost, I would like to thank Prof. Antonio Franchi for supervising my thesis and for trusting me throughout the entire project. I am grateful for the opportunity to carry out my work within his research group. A sincere thanks goes to Dr. Yaolei Shen for his constant technical support and availability throughout every phase of the work. I also thank Dr. Federico Califano and Dr. Chiara Gabellieri for their valuable feedback and the genuine interest shown during our discussions.

I am deeply thankful to my family for being a solid and reassuring presence throughout my university journey. To my parents Carlo and Graziella, my sister Alessia, my grandmothers Ada and Angelina, and all my loved ones: thank you for the daily support and strength you have always provided.

A special thanks goes to Mariagiulia, who more than anyone else taught me to aim higher – reminding me that «non est ad astra mollis e terris via»– and constantly pushed me to become a better version of myself. Her radiant presence has guided me every step of the way.

I would like to thank my lifelong friends, starting with those from Benevento – especially Carmine, Giulio and Tiziano – for always being there for me despite the distance.

A huge thanks to Simone, who has shared nearly every aspect of my university experience, from our first exams to our graduation. Together with Dumitru, Francesco and Karim, they've been classmates, travel companions, and, above all, people I have always been able to count on – both academically and personally. I also thank my dear friend and former housemate Antonio, as well as Edoardo, Lorenzo, Mert, and all the other companions from "Cueli della menza", for all the conversations and laughs we had, and for the countless lunches together. Those moments have been just as important to me as any time spent studying.

I am grateful to Giulia, who has always been someone I could trust and who helped make Rome feel more familiar. I am also thankful to Verina, whose genuine kindness brightened everyday life at the Assisi residence. A special thought goes to Eleonora, whose support and the moments we shared gave me perspective on what truly matters.

Finally, I thank the members of the Sapienza Flight Team for our collaboration, which further fueled my passion and motivation throughout this path.

To all of you, once again, thank you.

Abstract

Aerial manipulation using unmanned aerial vehicles (UAVs) equipped with suspended flexible cables is an emerging technology that already underpins infrastructure inspection, agricultural servicing, logistics, search-and-rescue, and defence operations, while an even broader spectrum of applications continues to unfold. The cable introduces complex, highly nonlinear continuum dynamics, making precise modeling and real-time control challenging. This work presents a comprehensive framework to model and control the dynamics of a deformable, extensible cable suspended from a quadrotor UAV during aerial pick-and-place operations. First, a detailed nonlinear dynamic model based on partial differential equations is developed and discretized through a finite difference method to accurately represent the cable continuum. Given the high computational complexity, a data-driven model reduction strategy that uses proper orthogonal decomposition is proposed. The reduced-order model (ROM) significantly decreases computational effort while retaining critical dynamic behaviors essential for accurate control. Leveraging this ROM, a nonlinear predictive control module is implemented, combining model predictive control with an efficient trajectory optimization strategy tailored to the hybrid nature of the system. To support far-sighted decision making, a complementary offline planning scheme is developed that leverages the same model reduction method to generate dynamics-aware, constraint-compliant reference trajectories suitable for navigation in cluttered environments. This integration significantly enhances the robustness and practical applicability of the overall control framework in realistic aerial manipulation scenarios. Comprehensive simulation studies demonstrate the effectiveness of the overall approach. Specifically, simulations evaluating the reduced-order model verify its numerical stability, computational efficiency, accuracy, and robustness to parameter uncertainty. The results obtained with the proposed control strategy highlight precise and stable management of cable deformations, nonlinearities, and hybrid phenomena, such as payload attachment and detachment. Finally, numerical validation of the planning strategy in constrained environments shows its ability to generate dynamically feasible trajectories that respect state boundaries. In summary this thesis provides significant advancements in achieving real-time solutions for practical aerial manipulation tasks, demonstrating a validated approach capable of executing precise cable-and-payload handling maneuvers under realistic operational constraints.

Contents

List of symbols

A	Cable cross-sectional area
$\boldsymbol{a}$	Orthonormal mode coefficient vector
$\boldsymbol{\beta}$	Generalized force density
b_c	Cable drag coefficient
b_p	Payload drag coefficient
$\mathcal{C}$	HOSVD core tensor
$\boldsymbol{c_f}$	Propeller thrust coefficient
$\boldsymbol{c_\tau}$	Propeller torque coefficient
$\mathcal{D}$	Generalized spatial domain
D	Planner-trajectory segments
d	Spatial decimation factor
δ	Log-barrier extension threshold
δ_{jk}	Kronecker delta
$\mathcal{E}$	Set of transition edges
E	Cable Young's modulus
E_m	Mode relative energy
ϵ	Generalized ROM approximation error
$\boldsymbol{\varepsilon}$	Generalized tracking error
η_κ	Decay rate of κ
η_δ	Decay rate of δ
$\mathcal{F}_w$	World reference frame
$\mathcal{F}_b$	Quadrotor body reference frame
$\boldsymbol{f}$	Rotor thrust vector
$\boldsymbol{g}$	Continuous-time flow function
$\boldsymbol{G}$	Discrete-time flow function
g_0	Gravitational acceleration
γ	Inequality-constraint function
Γ	Set of infeasible continuous states
$\mathcal{H}_c$	Continuous-time hybrid automaton
$\mathcal{H}_d$	Discrete-time hybrid automaton
H	MPC prediction horizon
h_d	Spatial step size after decimation
h_k	MPC time step size
h_s	Spatial step size (FDM)
h_w	MPC iteration wall time
i	Cable node index
$\boldsymbol{J_b}$	Quadrotor inertia matrix
$\boldsymbol{K}$	Geometric controller gain matrix
k	MPC time step index
κ	Log-barrier severity parameter
L	Cable length
Λ	Guard set
ℓ	MPC running cost
ℓ_H	MPC terminal cost
$\bar{\ell}$	MPC barrier cost

m	Orthonormal mode index
m_b	Quadrotor mass
m_p	Payload mass
m_0	Total system mass
M	Spatial intervals after downsampling
$\boldsymbol{n}$	Cable contact force map
n	Generalized number of dimensions
N	Number of cable segments (FDM)
Ω	Subspace of functions vanishing at cable endpoints
O	Number of POD snapshots
$\boldsymbol{o}$	Quadrotor center of mass
$\boldsymbol{p}$	Generalized position
ϕ	Orthonormal mode function
$\boldsymbol{\phi}$	Vector of orthonormal mode samples
$\boldsymbol{\Phi}$	Matrix of orthonormal mode samples
$\boldsymbol{\Psi}$	Reset matrix of hybrid automaton
$\mathcal{Q}$	Set of discrete states
q	Discrete state of hybrid automaton
R	ROM order
$\boldsymbol{R_b}$	World-to-body rotation matrix
$\boldsymbol{\rho}$	Reset function
ρ_c	Cable density
$\boldsymbol{r}$	Cable position map
$\mathfrak{r}$	ROM cable position map
$\mathbb{r}$	Cable equilibrium position map
$\bar{\boldsymbol{r}}$	Cable reference configuration
$\tilde{\boldsymbol{r}}$	Cable deviation from reference configuration
$\tilde{\boldsymbol{\mathcal{R}}}$	Snapshot tensor of cable deviation
s	Curvilinear coordinate along cable
$\boldsymbol{S}$	MPC running state weight matrix
$\boldsymbol{S_H}$	MPC terminal state weight matrix
σ	Singular value
$\boldsymbol{\Sigma}$	Singular values diagonal matrix
$\boldsymbol{T}$	Hybrid automaton successor map
T_w	Simulation wall time
t	Time
Θ	Planner prediction horizon
$\boldsymbol{\tau}$	Rotor torque vector
$\boldsymbol{U}$	Left singular vectors matrix
$\boldsymbol{u}$	Left singular vector
Υ	Obstacle ellipsoidal region
$\mathcal{V}$	Space of control inputs
$\boldsymbol{V}$	Right singular vectors matrix
φ	Orthonormality weight function
$\boldsymbol{v}$	Control input of hybrid automaton
$\boldsymbol{v}_p$	Payload velocity in world frame
$\boldsymbol{W}$	MPC control weight matrix
w	Rotor angular rate
$\boldsymbol{w}$	Propeller angular-rate array
$\boldsymbol{x}$	Generalized physical field
$\boldsymbol{\xi}$	Generalized spatial coordinate
$\boldsymbol{\mathcal{X}}$	Snapshot tensor of physical field
$\mathcal{Z}$	Space of continuous states
$\boldsymbol{z}$	Continuous state of hybrid automaton

Part I

Introduction

Chapter 1

Context, objectives and outline

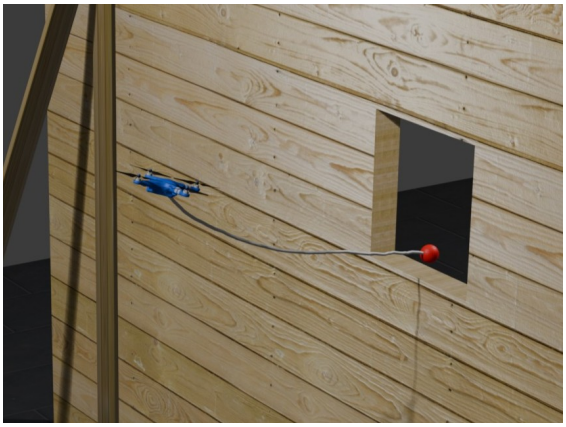


Figure 1.1. UAV carrying a slung payload through a narrow opening in a wall.

1.1 Motivation

Over the past decade, multirotor unmanned aerial vehicles (UAVs) have evolved from academic prototypes to indispensable workhorses across diverse domains such as infrastructure inspection, cinematography, precision agriculture, and – more recently – autonomous delivery and logistics. To perform physical interaction tasks, most commercial UAVs rely on simple rigid mechanisms such as grippers or winches. More advanced designs – equipped with dexterous robotic arms or multi-link appendages – have pushed the frontier of aerial manipulation further, yet they often introduce challenges in terms of payload, energy consumption, and system complexity [24].

A particularly promising yet under-explored approach is the use of flexible cables to interface with the environment. Tether-based aerial manipulation offers compelling advantages: it increases safety by distancing the vehicle from hazardous objects; allows for interaction in confined or cluttered spaces; and, if properly controlled, can even exploit passive dynamics to reduce energy consumption during maneuvers. Fig. 1.1 offers a glimpse of these opportunities. However, these benefits come at the cost of significantly increased modeling and control complexity.

Unlike rigid appendages, flexible cables exhibit continuum dynamics that are inherently infinite-dimensional, governed by partial differential equations (PDEs). These dynamics involve nonlinear coupling, distributed mass, elasticity, damping, and – when payloads are attached – hybrid mode transitions such as slack-to-taut switching. As a result, traditional control strategies often simplify the cable as a rigid pendulum, a valid approximation only under small-angle and taut-cable assumptions. While such simplifications enable real-time feasibility, they fail to capture high-frequency vibrations, slackness, or distributed deformations– features that become crucial in dynamic pick-and-place scenarios.

On the opposite end of the spectrum, high-fidelity PDE-based models offer accurate descriptions of cable motion, especially under aggressive UAV trajectories or during payload transitions. Yet these models are computationally intensive and impractical for real-time control due to the sheer number of degrees of freedom involved. Bridging this gap – achieving real-time performance without sacrificing physical fidelity – emerges as a critical challenge.

In summary, the motivation behind this thesis is twofold: to elevate the modeling of UAVs with flexible cables beyond simplified approximations, and to integrate these enhanced models within a computationally viable control architecture capable of handling real-time, high-precision manipulation tasks.

1.2 Objectives

At a high level, the problem to solve is to design a complete control framework for a quadrotor UAV equipped with a flexible, extensible cable capable of handling dynamic payload pick-and-place operations. The framework must achieve precision and agility in trajectory tracking, even under aggressive and rapid maneuvers, meanwhile featuring damping and limitation of cable oscillations and strain; also, it has to handle robustly discrete events (i.e. attachment and detachment of payloads). Moreover, ensuring real-time computational feasibility is an added bonus.

Achieving these objectives involves overcoming several interconnected challenges:

- Capturing the essential features of the nonlinear, distributed dynamics of the cable in a form that remains tractable for real-time control and planning.
- Dealing with the hybrid nature of the problem, in particular managing transitions between discrete operational modes in a predictable and physically consistent way.
- Integrating all of the above into a unified control architecture capable of operating effectively under realistic computational and environmental constraints.
- Designing and applying a validation strategy that rigorously assesses the proposed methodology through representative simulations and metrics.

1.3 Literature review

The literature on UAVs with flexible cables is vast, but it can be broadly categorized into three main approaches: (i) rigid pendulum models, (ii) slack/taut hybrid dynamics, and (iii) lumped representations of flexible continua. Each approach has its strengths

and weaknesses, and the choice of model often depends on the specific application and performance requirements. Please refer to Tab. 1.1 for a summary of this section.

Table 1.1. Summary of literature on UAVs carrying cable-suspended payloads

Reference	Modeling	Control	Planning
Palunko et al. (2012) [25]	Rigid pendulum	–	DP
Pizetta et al. (2015) [26]	Planar rigid pend.	Swing rejection	–
Foehn et al. (2017) [10]	Rigid p., RRP joint	–	MPCC
Chang et al. (2023) [7]	Damped elastic pend.	Payl. compensation	–
Goodman et al. (2023) [15]	Multiple elastic links	Geometric	–
Li et al. (2023) [21]	Multiple rigid links	NMPC	–
Sreenath et al. (2013) [31]	Hybrid	DF, geometric	DF
Tang et al. (2015) [33]	Hybrid	DF, geometric	DF, MIQP
Kotaru et al. (2017) [19]	Hybrid, elastic pend.	DF, geometric	DF
Zeng et al. (2019) [40]	Hybrid, pulley	DF, geometric	DF
Sarvaiya et al. (2024) [28]	Hybrid	HPA-MPC	–
Wang et al. (2024) [38]	Hybrid	DF, hybrid NMPC	DF, ONCC, ALM
Quisenberry et al. (2012) [27]	Serial thin rods	–	–
Goodarzi et al. (2015) [14]	Serial rigid pend.	Geometric	–
Liu et al. (2017) [22]	PDE	Lyapunov-based	–
Gabellieri et al. (2023) [12]	Serial elastic pend.	DF	–
Shen et al. (2024) [29]	PDE, ROM	NMPC, ROM	PSO, ROM

Swinging pendulum and link models Early work treated the cable and payload as a single rigid pendulum, that is a massless rigid link with a point-mass payload – the bob (see Fig. 1.2a). This greatly simplifies the dynamics and was sufficient for basic swing attenuation and trajectory planning.

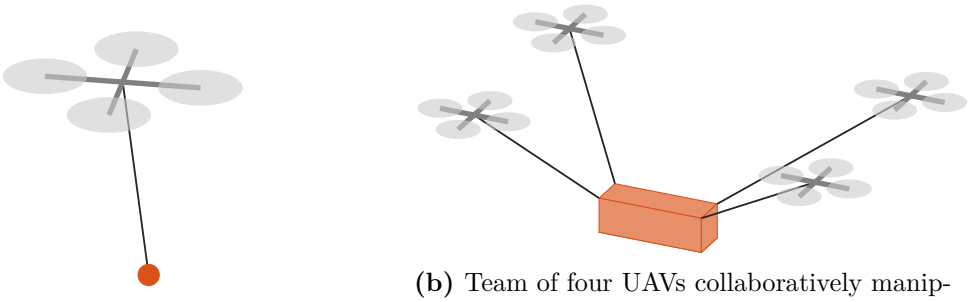
For instance, [25] modeled a quadrotor with a suspended payload as a spherical pendulum and used a dynamic programming approach to generate swing-free trajectories. Their planner could find aggressive maneuvers that transfer the payload with minimal residual oscillation. In contrast, the authors of [10] proposed a trajectory generation framework based on a model that treats the tethered payload as a passive link connected to the UAV through a pair of revolute joints and a prismatic joint. By imposing geometric constraints on the prismatic joint, they formulated a mathematical program with complementarity constraints (MPCC) that accounts for the reaction forces arising from the motion of the payload, without having to explicitly formulate a hybrid model.

In related work, the effects of the pendulum were sometimes treated as external disturbance to the UAV. For example, [26] applied feedback linearization to a planar UAV-pendulum model, effectively canceling the coupling so that the payload swing had little influence on the flight control of the vehicle. Alternatively, [7] designed an energy-based adaptive control law for real-time compensation of the effects of the tethered payload, represented as an elastic rope with damping. These studies validated (mostly in simulation) that relatively simple controllers can stabilize the payload or achieve tracking, provided the payload remains directly beneath the UAV and the cable stays taut.

In parallel, studies on multi-UAV payload transport successfully employed the rigid/elastic link model for effectively controlling a payload – as in Fig. 1.2b. For

example, [15] and [21] implemented cooperative controllers (respectively, a geometric controller and a nonlinear model predictive controller – or NMPC) for multiple drones carrying a single payload, still assuming taut cables and treating swing dynamics in a simplified manner.

These early works established baseline control and planning strategies but also highlighted limitations: the pendulum/link model cannot capture cable sag or oscillatory modes, and neglecting slackness means the approach fails if the cable goes loose or if high-frequency vibrations occur.



(a) Aerial system composed of a quadrotor carrying a payload with a rigid cable/link.

(b) Team of four UAVs collaboratively manipulating a bulky rigid body through cables that remain taut throughout the operation.

Figure 1.2. Conceptual representations of aerial systems based on the rigid-pendulum/link paradigm.

Slack/taut hybrid dynamics When a cable can go slack (e.g. if the UAV descends faster than the payload or in maneuvers that introduce slackness), the dynamics become hybrid – alternating between a taut-cable regime (where tension forces transmit between UAV and payload) and a slack regime (where the payload is in free flight, and the cable exerts no force until it re-tensions) (Fig. 1.3). Accounting for these mode changes is crucial for high-performance maneuvers or ensuring safety (no abrupt jerks when tension is regained).

[31] was an early example tackling this explicitly, as they treated the slack-taut transition as an inelastic collision and included a model of the impact at the moment of re-tensioning. The resulting model is a hybrid system with two modes: cable taut (payload slung under tension) vs. cable slack (payload in free-fall). Moreover, a significant advance came with the recognition that the system is differentially flat (DF) in every state, which allowed them to define flat outputs (like the payload position) for trajectory planning and geometric tracking control.
Similarly, [19] addressed slackness by introducing an elastic element in the cable model. They modeled the cable as a mass-damper spring, which inherently smooths out the shock of slack-to-taut transitions. Their geometric control law, the same as [31], could stabilize the payload even as the cable length oscillated slightly due to elasticity. These early notions of treating the slack-versus-taut situation as a mode-switching problem set the stage for later research on handling cable slackness explicitly.

Other researchers around that time further advanced trajectory generation within the hybrid-system paradigm. [33] formulated a mixed-integer quadratic program (MIQP) to generate payload trajectories, explicitly encoding the binary slack/taut state in the

optimization. Their MIQP planner could handle obstacles and other logical constraints by switching modes within the trajectory. Although the MIQP was solved offline, it demonstrated that one can formally optimize through discrete cable modes.

[40] further studied a variable-length cable (via a winch/pulley on the UAV) that could be intentionally paid out or reeled in. They considered both slack and taut phases and proved the entire UAV-pulley-payload system is differentially flat; using this, they designed a geometric controller capable of handling the transitions between slack and taut conditions. Their results indicated that careful trajectory planning (using flat outputs in each mode) combined with a switching controller can tackle multiple (alternative) objectives, such as payload attitude or payload position plus cable length control.

In the most recent years, researchers have started integrating such hybrid mode logic into complex real-time control frameworks. One notable example is the hybrid perception-aware model predictive control (HPA-MPC) in [28]. Here, the authors developed a nonlinear optimal controller that explicitly models the slack/taut dynamics of the cable as part of the system state. A mode variable (with appropriate constraints to ensure physical consistency) was included in the prediction model. In their formulation, when the predicted cable tension drops to zero, the MPC seamlessly switches to a ballistic payload model, and when tension is nonzero it uses the pendulum model. Importantly, they demonstrated this hybrid MPC on hardware, running entirely onboard a quadrotor, and showed stable payload tracking even through mode transitions. This result indicates that, with current computational power and careful coding, it is feasible to solve such a hybrid-dynamics optimal control problem in real time onboard a UAV. An added innovation in [28] is the inclusion of a perception-aware cost term – penalizing configurations where the payload moves out of the onboard camera field of view – which highlights a practical consideration when manipulating payloads: the controller must maintain line-of-sight for vision-based feedback. Their experiments confirmed that the hybrid MPC could keep the payload in view and stable despite a human operator inducing slack in the cable.

While such advanced hybrid controllers are promising, they come with challenges. One issue is how to detect and time the mode switches in practice. Simpler approaches use threshold logic on cable tension or geometry (e.g. they assume that the cable is taut whenever the distance between UAV and payload exceeds the maximum slack length). More sophisticated methods, like complementarity constraints or state estimators, have been proposed to handle this smoothly. For instance, [39] formulated the collaborative control of ground robots towing a cable-suspended payload as a hybrid trajectory optimization problem. Their approach explicitly handles cable slackness through complementarity constraints and enumerates cable modes (slack or taut) via binary decision variables, solving the nonlinear optimization problems stemming from all the possible combinations of discrete states for all the robots involved in the experiment.

Although originally applied to quadrupedal robots, this methodology, particularly the explicit handling of slack and taut cable conditions through complementarity constraints, is conceptually transferable to aerial vehicles. The work [38] is a notable example: according to the paper, the authors developed a complete impact-aware framework which includes a hybrid nonlinear MPC controller and an offline trajectory planner that solves an optimization problem with nonlinear complementarity constraints (ONCC) using an augmented Lagrangian method (ALM). Their approach was validated in simulation and through hardware experiments, demonstrating the ability of the controlled system to handle complex maneuvers such as passing through a narrow opening.

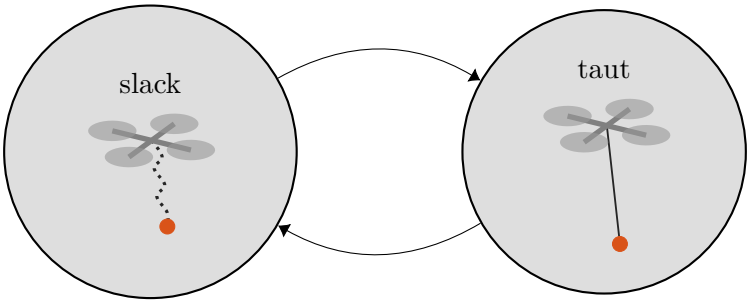


Figure 1.3. Scheme of the UAV-cable-payload system modeled as a hybrid system with two states: slack and taut. The slack state is active when cable tension is negligible, decoupling the payload from the UAV dynamics. Otherwise, the system enters the taut state, where the cable is treated as a rigid/elastic link.

Lumped representations of flexible continua The previous abstractions become inadequate when the cable is long, lightweight, or elastic, because it can support complex vibrational modes (transverse oscillations, longitudinal stretching) rather than remaining straight under tension. To capture these effects, later studies began modeling the cable as a flexible continuum – sometimes extensible as well. The highest-fidelity model treats the cable as a distributed parameter system (a string) described by PDEs. This model captures the nonlinear dynamics of a flexible extensible cable, including spatial deformations, slack formation, and wave propagation driven by internal tension and external forces. However, it introduces infinitely many degrees of freedom, which is intractable for real-time internal model control.

In the aerospace community, [22] modeled aerial refueling hoses as distributed-parameter systems enabling the simulation of hose whip and oscillations during tanker-receiver maneuvers. Through Lyapunov-based design, they managed to use the infinite-dimensional model to build a boundary controller for minimizing the displacement of the hose tip during refueling.

In order to achieve greater control authority over the cable continuum, researchers have looked for ways to discretize and reduce the infinite-dimensional model to a manageable size. One common approach is to approximate the cable by a finite number of lumped elements – as in Fig. 1.4a. [14] used a series of rigid links connected by pin joints: in their model, each link is characterized by its own mass and length, effectively dividing the cable into multiple pendulum segments. By increasing the number of segments, the model can approximate more complex cable shapes (approaching a continuous curve). The authors showed that the model was sufficient to design a geometric nonlinear controller that stabilizes the hanging equilibrium of the chain of links. Clearly, using a series-of-links model means that the number of links can be increased for higher fidelity, but it becomes more computationally demanding and still introduces some modeling error since a discrete link chain is only an approximation of a continuous rope.
An alternative discretization was explored in [27], which modeled the cable with many small distributed-mass elements, referred to as thin rods. This was used mostly for offline dynamic analysis rather than control, but the authors showed its superiority with respect to the multiple-pendula models.

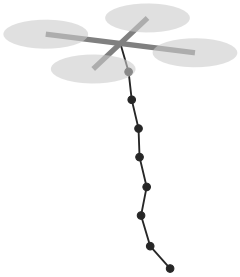
To bridge the gap between high-fidelity modeling and real-time control, researchers have adopted model order reduction techniques, and a prominent method in recent UAV

cable research is proper orthogonal decomposition (POD). In essence, POD empirically derives a set of orthonormal mode shapes (basis functions) from data (typically simulation or experimental data of the cable motion), and then projects the full model onto the subspace spanned by a few dominant modes (see Fig. 1.4b). The authors of [16] were the pioneers in using POD to reduce the state dimension of the model of soft robots, allowing to derive fast, but still physically meaningful, reduced-order models (ROMs) for real-time control.

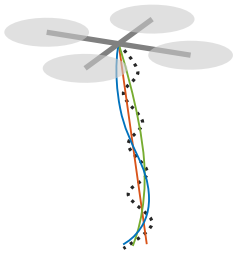
On the same thread, [29] applied POD to a flexible cable suspended from a quadrotor. They first derived the continuous equations for a cable (a string PDE with gravity and tension forces, plus suitable boundary conditions at the quadrotor attachment and the free end); next, since an analytical solution of the PDE is intractable in real time, they discretized it using a finite-difference method (FDM) into a large number of small segments. They then simulated this "full" model under various maneuvers to gather a dataset of cable shape trajectories and, by performing POD, they extracted a set of mode shapes. The authors report that the first few POD modes capture the vast majority of the motion of the cable; thus, by truncating to a small number of modes, they obtained a reduced-order model that still represents the most important flexible dynamics. Notably, these modes are not simple sinusoidal shapes; they are tailored to the dynamics induced by the UAV motions (e.g. one mode might capture the fundamental swing of the entire cable, another might capture a curvature with one node along the cable, etc.). By using this ROM in their controller, they achieved a dramatic reduction in state dimension with minimal loss of fidelity in the frequency range of interest. They verified that over short time horizons (on the order of a few seconds), the predicted cable shape matched the FDM truth model closely. The benefit for control is clear: the ROM state had on the order of a few variables (including the UAV state and a handful of mode amplitudes), which is tractable for MPC, whereas the full discretized cable had hundreds of states. In addition, it turns out that the ROM is also useful in trajectory generation, as the authors used it offline, together with particle swarm optimization (PSO), to find a reference sequence for the controller that could guide the system to traverse a narrow window without collisions. The question naturally arises: *how many modes or segments are enough?* [29] chose the number of POD modes based on an energy cutoff criterion (capturing most of the deformation energy). Typically, even a single elastic mode in the cables (modeled as a linear spring) was sufficient to design a stable controller, because the dominant disturbance was the collective swing of the cable. However, for more dynamic scenarios where higher-frequency whipping of the cable occurs, more modes are needed. In general, using too few modes can omit important dynamics (leading to residual oscillations that the controller cannot predict), whereas using too many modes makes the controller slow and can even introduce numerical ill-conditioning in the MPC optimization. There is no one-size-fits-all answer; instead, it is appropriate to choose the smallest model that achieves acceptable tracking accuracy. In summary, flexible cable modeling has advanced to the point where we can obtain good reduced models for control, but choosing the model granularity remains a design decision that balances performance and solvability.

When modeling flexible cables, researchers must also make assumptions about additional physical effects. One important factor is damping: real cables have internal material damping and experience aerodynamic drag, which help to dissipate oscillations over time. For example, the PDE model of [29] explicitly incorporated aerodynamic drag distributed along the cable length. Generally, a bit of damping in the model not only

makes it more realistic, but also improves numerical stability for simulation and control (preventing sustained oscillations that a real cable might naturally damp out).
Additionally, if the UAV or payload moves fast, aerodynamic forces on the cable can become non-negligible. Long cables can experience noticeable drag, effectively increasing damping and altering the tension profile (a blowing wind can even induce a catenary shape in a long tether). Most UAV payload papers either assume quasi-steady aerodynamic drag (proportional to cable segment velocity) and tune a drag coefficient, or they neglect it if the cable is short. Notably, [12] analyzed an aerial cable system composed of two UAVs holding a series of elastic links in a (possibly) viscous medium and demonstrated the effectiveness of their proposed control law (based on differential flatness) through simulations of scenarios where part of the cable is dragged on the surface of water.



(a) UAV-driven cable modeled as a series of lumped masses connected by rigid or elastic links.



(b) POD-based reduced-order model: the shape of the cable (dotted line) is approximated by the combination of multiple modes (solid lines), each representing a different oscillatory behavior.

Figure 1.4. Illustrations of the fundamental approaches to approximate the flexible cable continuum.

1.4 Proposed contributions

This thesis is rooted primarily in the framework introduced in [29], which models an aerial system composed solely of a UAV and a flexible cable – without a suspended payload – using PDEs to represent the cable dynamics with high fidelity. In their approach, a ROM is derived and employed as the prediction model within a nonlinear MPC scheme, enabling direct trajectory tracking of the shape and behavior of the cable. The main contribution of the present work is a significant and non-trivial extension of that framework to accommodate a hybrid system suitable for aerial pick-and-place manipulation.

This extension necessitates the formulation of a hybrid model, since the original does not account for the presence of a payload nor does it support discrete mode transitions such as attachment and detachment. While simplified pendulum or rigid-link models (e.g. [25, 10]) can describe payload dynamics to a limited extent, they lack accuracy due to their inability to handle complex oscillatory modes and/or abrupt changes in boundary conditions. On the other hand, more advanced slack/taut hybrid models (e.g. [31, 19, 38]) can handle discrete mode transitions but remain reliant on simplified assumptions that do not accurately predict the distributed cable deformation crucial in highly dynamic maneuvers. Consequently, neither approach individually suffices for tasks

that simultaneously require high accuracy, distributed deformation prediction, and hybrid mode-switching capabilities – limitations addressed by the methodology proposed here.

Among the implications of this, the design of the MPC controller becomes substantially more complex, primarily due to the hybrid nature of its internal predictive model. Therefore, a reduced-order internal model, analogous in spirit to the one proposed in [29], is obtained via POD. However, unlike the original formulation, this decomposition is adapted to account for the hybrid dynamics of the system and the increased relevance of the cable tip and payload in pick-and-place scenarios. The resulting reduced model retains the ability to capture critical features of the evolution of the cable across discrete mode transitions, which is essential for accurate prediction and control in manipulation tasks. To the best of the author's knowledge, this constitutes the first integration of a POD-based continuum tether model explicitly designed to handle hybrid mode-switching events, preserving modal coherence across discrete transitions.

Once a control module capable of handling both the flexible cable and the possible payload – including across hybrid transitions – has been developed, the framework is further extended to fully exploit its potential (e.g. in cluttered environments) through the design of a dedicated trajectory planner. As in [29], this planner leverages the ROM to enable efficient trajectory generation. However, it incorporates several key modifications to accommodate the increased complexity of the optimisation landscape, which arises from the presence of hybrid dynamics, environmental constraints, and the added requirements linked to the task.

Finally, it is important to emphasise that the entire development of this work has been guided by the objective of ensuring practical implementability, with particular attention paid to the computational efficiency of the proposed methods. This foresight ensures that the framework remains suitable not only for simulation studies but also for potential deployment in real-time applications.

1.5 Thesis outline

This thesis is structured into four main parts, each contributing to guiding the reader through the progression from fundamental knowledge to detailed methodological developments and their validation.

Part I: Introduction Part I provides essential context, motivation, and the objectives driving this research.

In Ch. 1, the reader is informed with the necessity and benefits of advancing aerial manipulation capabilities of UAVs equipped with flexible cables (Sec. 1.1). Then, the research problem is stated formally (Sec. 1.2), with a clear definition of the system under study – a quadrotor with a flexible, extensible cable capable of picking up or releasing payloads. Following this, an extensive literature review assesses existing approaches (Sec. 1.3), setting the stage for the novel contributions of this work. These contributions are then summarized (Sec. 1.4), clarifying how the presented methodology advances the state of the art.

Next, Ch. 2 introduces the theoretical and technical background necessary for understanding the subsequent developments. This chapter covers essential concepts such as hybrid systems modeling and their predictive control (Sec. 2.1), POD-based

model-order reduction methods for continuum systems (Sec. 2.2), partial differential equation modeling for flexible cables (Sec. 2.3), and fundamental aspects of quadrotor dynamics and control (Sec. 2.4).

Part II: Methodology Part II delves into the core of the thesis, detailing the novel methodological contributions proposed.

Initially, Ch. 3 develops a general and systematic approach for reducing the complexity of models involving flexible cables (Sec. 3.1), adapting the introduced POD-based model reduction technique to the case of extensible strings, possibly with dynamic boundary conditions. This approach is fundamental for computational efficiency while maintaining accuracy. Then, the specific aerial system at hand is defined (Sec. 3.2), outlining assumptions, configurations, and key parameters. A rigorous hybrid PDE model, including the conditions regulating hybrid transitions, represents the dynamics (Sec. 3.3). Also, it fathers a finite-dimensional model (Sec. 3.4) – whose equations are derived via FDM – that ensures computational tractability of the PDE model with minimal loss of fidelity.

Building upon this, Ch. 4 formulates a comprehensive hybrid control and planning framework that utilizes the reduced-order model within for the purposes of nonlinear predictive control. Special attention is given to the MPC architecture (Sec. 4.1), notably its internal hybrid model capable of handling discrete transitions associated with attaching or detaching payloads. Serving as a bridge between the MPC and the drone, developments on the low-level quadrotor controller are addressed (Sec. 4.2), tackling position and attitude regulation critical for the effectiveness of the entire hierarchy. Finally, a segmented trajectory planner is proposed (Sec. 4.3) to handle the nonconvexity of a variety of complex aerial manipulation tasks – especially those involving constraints.

Part III: Numerical validation Part III presents a thorough evaluation of the proposed methodology, demonstrating their effectiveness and applicability.

Ch. 5 outlines the experimental setup, clearly defining the metrics (Sec. 5.1), system configurations (Sec. 5.2), and ROM construction process (Sec. 5.3). The chapter also describes the simulation environment and the numerical tools used for validation, ensuring reproducibility and clarity in the experimental design.

Ch. 6 systematically addresses the proposed model reduction technique, assessing its performance – both in terms of accuracy and computational efficiency –, with respect to the reduction order (Sec. 6.1) and the payload mass (Sec. 6.2). An analysis on robustness against uncertainty in critical parameters of the cable and the payload is conducted as well (Sec. 6.3). The chapter features comparisons between the proposed reduction approach and the state of the art in model reduction.

Finally, Ch. 7 validates the proposed hierarchical control framework across various representative scenarios. It features an experiment regarding the regulation of the shape of the cable (Sec. 7.1), including a comparison with a naive MPC-less control strategy; then, tracking of highly-dynamic online and offline hybrid trajectories is tackled (Sec. 7.2); finally, the full power of the proposed framework is demonstrated in an articulated pick-and-place scenario involving obstacle avoidance (Sec. 7.3). These experiments highlight both the practical applicability and the level of performance achieved by the proposed framework.

Part IV: Conclusions At last, Part IV concludes the thesis by consolidating the main findings and reflecting on their significance.

Ch. 8 first provides a synthesis of the key contributions (Sec. 8.1), highlighting how each component of the proposed framework – ranging from the hybrid model formulation to the reduced-order controller and planner – effectively addresses the objectives posed at the starting line. The chapter then turns to future perspectives (Sec. 8.2), identifying limitations encountered throughout the development and validation phases, and outlining potential extensions. These perspectives open the way for further developments in real-time aerial manipulation using flexible, hybrid-interacting tether systems.

Chapter 2

Theoretical and technical background

2.1 Hybrid modeling and predictive control

Hybrid automata A hybrid system is characterized by the interplay of continuous dynamics and logical decision making, typically modeled by differential equations and discrete transitions, respectively. A rigorous formalism for such systems is provided by the notion of a hybrid automaton.

The components for a possible definition of a (controlled) hybrid automaton tailored for the purposes of this work are introduced below [13]. One needs to define:

- a finite set of discrete states $\mathcal{Q}$;
- a set of edges $\mathcal{E} \subseteq \mathcal{Q} \times \mathcal{Q}$, representing transitions between discrete states;
- a space of continuous states $\mathcal{Z}$, possibly infinite-dimensional;
- a control space $\mathcal{V}$;

and, in addition:

- the flow functions $\boldsymbol{g^q} : \mathcal{Z} \times \mathcal{V} \to T\mathcal{Z}, \forall q \in \mathcal{Q}$, which associate a distinct continuous dynamics to each discrete state;
- the guard sets $\Lambda^e \subset \mathcal{Z}, \forall e \in \mathcal{E}$, each defining a subset of $\mathcal{Z}$ for a transition across the corresponding edge;
- the reset functions $\boldsymbol{\rho^e} : \Lambda^e \to \mathcal{Z}, \forall e \in \mathcal{E}$, which assign a new continuous state after a transition along a given edge;

with $T\mathcal{Z}$ denoting the tangent bundle of $\mathcal{Z}$.
The hybrid automaton is then defined as the tuple:

$$\mathcal{H}_c \coloneqq (\mathcal{Q}, \mathcal{Z}, \mathcal{V}, \mathcal{E}, \{\boldsymbol{g^q}\}_{q\in\mathcal{Q}}, \{\Lambda^e\}_{e\in\mathcal{E}}, \{\boldsymbol{\rho^e}\}_{e\in\mathcal{E}}) \,. \tag{2.1}$$

Essentially, given an initial state in $\mathcal{Q} \times \mathcal{Z}$, the system evolves according to $\boldsymbol{g^q}$ until the continuous state enters a guard set Λ^e for some edge $e \in \mathcal{E}$. At that point, the

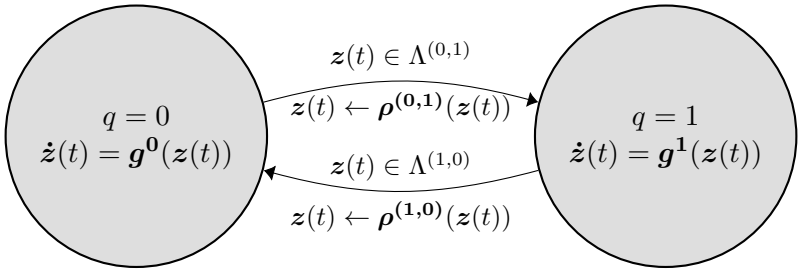


Figure 2.1. State transition diagram of a dummy continuous-time hybrid system akin to (2.1) with state variable $\boldsymbol{z}(t) \in \mathcal{Z}$ and two discrete states corresponding to $q = 0$ and $q = 1$.

system must transition to a new discrete state through the reset map $\boldsymbol{\rho}^{\boldsymbol{e}}$, and the process continues. Fig. 2.1 illustrates a basic example of a hybrid automaton according to the adopted notation.

In discrete time, the flow maps $\boldsymbol{g}^{\boldsymbol{q}}$ are replaced by their discrete-time counterparts $\boldsymbol{G}^{\boldsymbol{q}} : \mathcal{Z} \times \mathcal{V} \to \mathcal{Z}$:

$$\mathcal{H}_d \coloneqq (\mathcal{Q}, \mathcal{Z}, \mathcal{V}, \mathcal{E}, \{\boldsymbol{G}^{\boldsymbol{q}}\}_{q\in\mathcal{Q}}, \{\Lambda^e\}_{e\in\mathcal{E}}, \{\boldsymbol{\rho}^{\boldsymbol{e}}\}_{e\in\mathcal{E}}) \,. \tag{2.2}$$

Hybrid model predictive control Model predictive control is particularly well-suited for systems exhibiting nonlinear, hybrid, or constrained dynamics. In this proceeding, the predictive nature of hybrid MPC enables the anticipation of complex behaviors of the cable such as swing, vibrations and tip boundary transitions[1].

Throughout this section, the plant dynamics internal to the optimizer are represented by the discrete-time hybrid automaton (2.2). For the sake of compactness, with time step index $k \in \mathbb{N}$, the successor map $T : \mathcal{Q} \times \mathcal{Z} \times \mathcal{V} \to \mathcal{Q} \times \mathcal{Z}$ is introduced:

$$(q(k+1), \boldsymbol{z}(k+1)) = \boldsymbol{T}(q(k), \boldsymbol{z}(k), \boldsymbol{v}(k)), \tag{2.3}$$

At step k, $q(k) \in \mathcal{Q}$ is a variable that represents the discrete state of the system, $\boldsymbol{z}(k) \in \mathcal{Z}$ is the continuous state, and $\boldsymbol{v}(k) \in \mathcal{V}$ is the control input. $\boldsymbol{T}$ returns the deterministic successor generated by the automaton, i.e. it encapsulates flow, edge selection, guard and reset.

Given initial conditions $q(0)$ and $\boldsymbol{z}(0)$, corresponding to the real-time system state, the optimal control sequence is generally determined at each controller call by solving the following optimization problem with prediction horizon of length $H \in \mathbb{N}$:

$$\begin{aligned} \min_{\{\boldsymbol{v}(k)\}_{k=0}^{H-1}} & \left(\sum_{k=0}^{H-1} \ell^{q(k)}\left(\boldsymbol{z}(k), \boldsymbol{v}(k)\right) + \ell_H^{q(H)}\left(\boldsymbol{z}(H)\right) \right) \\ \text{s.t.} \quad & (2.3), \quad k = 0, \ldots, H-1 \end{aligned} \tag{2.4}$$

Here, at some discrete state $q(k)$, $\ell^{q(k)}(\boldsymbol{z}(k), \boldsymbol{v}(k))$ is the running cost for the state-control pair $(\boldsymbol{z}(k), \boldsymbol{v}(k))$; similarly, $\ell_H^{q(H)}(\boldsymbol{z}(H))$ is the terminal cost at the end of the horizon. Commonly, only the first control input $\boldsymbol{v}(0)$ is actually sent to the plant, while the rest of the sequence is discarded.

[1] MPC is inherently able to handle hard input and state constraints, but for the scope of this study they are omitted.

Problem (2.4) is mixed-integer and nonlinear: the Boolean mode decisions encoded in (2.3) make the cost landscape non-convex, while the flow and reset maps introduce nonlinear equalities. Off-the-shelf quadratic-programme engines therefore cannot deliver a solution within sufficiently short time. Practical hybrid MPC must rely on a solver that is able to handle and exploit the piecewise-smooth structure of the dynamics – for instance, hybrid iterative linear quadratic regulator (HiLQR, Appx. A.1).

2.2 Model order reduction for continuum systems

Proper orthogonal decomposition[2] is a widely recognized technique for reducing the complexity of a physical field by extracting a finite number of its most significant modes with minimal L^2-error [32]. It is a data-driven method, since it uses data collected from the field to train a reduced-order representation.

According to [16, 32], a key step in POD is to project the target function onto a reduced-order functional basis. A particularly effective strategy involves decoupling time and space by combining the purely-spatial orthonormal modes via time-dependent coefficients. By targeting a (possibly vector-valued) function $\boldsymbol{x}(\boldsymbol{\xi}, t) \in \mathbb{R}^n, n \in \mathbb{N}$ over a physical domain $\mathcal{D} \subset \mathbb{R}^3$, where $\boldsymbol{\xi} \in \mathcal{D}$ is a spatial coordinate and $t \in [0, \infty)$ is time, one can write:

$$\boldsymbol{x}(\boldsymbol{\xi}, t) \approx \sum_m \boldsymbol{a_m}(t)\phi_m(\boldsymbol{\xi}). \tag{2.5}$$

Here, $\boldsymbol{a_m}(t) \in \mathbb{R}^n$ are the time-dependent coefficients and $\phi_m(\boldsymbol{\xi}) \in \mathbb{R}$ are the spatial basis functions – with $m \in \mathbb{N}$.

The modes may be determined out of a set of observations acquired at discrete times, or snapshots, of $\boldsymbol{x}(\boldsymbol{\xi}, t)$, representative of the intended behavior of the system. These can be captured by sampling $\boldsymbol{x}(\boldsymbol{\xi}, t)$ both in space and time at discrete points.
If one constructs a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{n\times(M+1)\times(O+1)}$ where each page is a snapshot:

$$\boldsymbol{\mathcal{X}}_{::j} \coloneqq \begin{bmatrix} \boldsymbol{x}(\boldsymbol{\xi_1}, t_j) & \boldsymbol{x}(\boldsymbol{\xi_2}, t_j) & \dots & \boldsymbol{x}(\boldsymbol{\xi_{M+1}}, t_j) \end{bmatrix}, \quad \forall j = 1, \dots, O+1,$$

with $\boldsymbol{\xi_j} \in \mathcal{D}, \forall j = 1, \dots, M+1$ being the spatial sampling points, higher-order singular value decomposition (HOSVD) [8] breaks it down as:

$$\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{C}} \times_1 \boldsymbol{U_1} \times_2 \boldsymbol{U_2} \times_3 \boldsymbol{U_3},$$

where $\boldsymbol{\mathcal{C}} \in \mathbb{R}^{n\times(M+1)\times(O+1)}$ is the core tensor and $\boldsymbol{U_j}, \forall j = 1, 2, 3$ are unitary matrices. Mind that these matrices could also be found applying standard singular value decomposition to the mode-j unfolding of $\boldsymbol{\mathcal{X}}$:

$$\boldsymbol{\mathcal{X}}_{(j)} = \boldsymbol{U_j}\boldsymbol{\Sigma_j}\boldsymbol{V_j}^H, \quad \forall j = 1, 2, 3.$$

Here, $\boldsymbol{V_j}^H$ is the Hermitian transpose of $\boldsymbol{V_j}$, and the singular values in $\boldsymbol{\Sigma_j}$ are assumed to be placed in descending order.
The matrix that serves the purpose is $\boldsymbol{U_2} \in \mathbb{R}^{(M+1)\times(M+1)}$:

$$\boldsymbol{U_2} = \begin{bmatrix} \boldsymbol{u_1} & \boldsymbol{u_2} & \dots & \boldsymbol{u_m} & \dots & \boldsymbol{u_{M+1}} \end{bmatrix}.$$

[2] In statistics and machine learning, POD is also referred to as principal component analysis – PCA in short.

The samples of the spatial basis functions are obtained by scaling the columns of $\boldsymbol{U_2}$. With uniform spatial grid, one has:

$$\boldsymbol{\phi_m} = \begin{bmatrix} \phi_m(\boldsymbol{\xi_1}) & \phi_m(\boldsymbol{\xi_2}) & \dots & \phi_m(\boldsymbol{\xi_{M+1}}) \end{bmatrix}^\top := \varphi \boldsymbol{u_m}, \quad \forall m = 1, \dots, M+1,$$

where $\varphi \in \mathbb{R}$ is a weight introduced to take spatial sampling into account, ensuring orthonormality[3] according to:

$$\delta_{jk} = \int_{\mathcal{D}} \phi_j(\boldsymbol{\xi}) \phi_k(\boldsymbol{\xi}) \, \mathrm{d}\mathcal{D}, \quad \forall j, k = 1, \dots, M+1. \tag{2.6}$$

It follows that the maximum number of independent modes that can be extracted is $\min\{M+1, n(O+1)\}$, and that it is appropriate to choose the number of snapshots $O+1 \geq \frac{M+1}{n}$ in accordance to that[4]. It means that (2.5) can be reformulated as:

$$\boldsymbol{x}(\boldsymbol{\xi}, t) \approx \sum_{m=1}^{M+1} \boldsymbol{a_m}(t) \phi_m(\boldsymbol{\xi}). \tag{2.7}$$

Vice versa, the coefficients of the linear combination are found via the exploitation of the orthonormality condition (2.6):

$$\boldsymbol{a}_m(t) = \int_{\mathcal{D}} \boldsymbol{x}(\boldsymbol{\xi}, t) \phi_m(\boldsymbol{\xi}) \, \mathrm{d}\mathcal{D}.$$

The extension to the time derivatives, used to obtain the reduced-order dynamics, is straightforward, as one only needs to differentiate with respect to t.

2.3 PDE modeling of flexible cables[5]

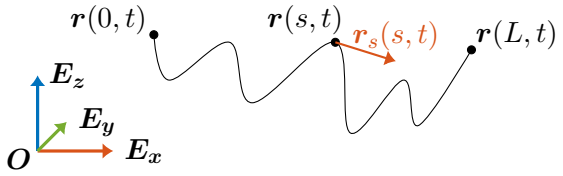


Figure 2.2. Extensible string in the Euclidean space.

String equations of motion In the Euclidean 3-space $\mathbb{E}^3$, consider a right-handed inertial frame of reference $\mathcal{F}_w = \{\boldsymbol{O}, \boldsymbol{E_x}, \boldsymbol{E_y}, \boldsymbol{E_z}\}$. A *string* is defined as a set of material points capable of occupying curves in $\mathbb{E}^3$ with the mechanical property of perfect flexibility – namely, no torque is generated when the string is bending or twisting [2].

[3]The orthonormality condition is written in its continuous form because the POD modes should ultimately be interpreted as functions over the continuous spatial domain. A precise correspondence between these continuous functions and their discrete samples will be established in Sec. 3.1.

[4]However, an even larger number increases the probability of obtaining a good representation of the original system.

[5]It is assumed that all functions of space and time appearing in the equations of this section are continuous on the interiors of their domains.

Identifying with $s \in [0, L]$ the curvilinear coordinate that parametrizes the material points of the string and with t time, the configuration of the string in the world frame is described by the map $\boldsymbol{r}(s,t) \in \mathbb{R}^3$ (please refer to Fig. 2.2). Note that $\boldsymbol{r}$ should be C^2 with respect to s and t, with C^2 denoting the space of twice-continuously-differentiable functions.
Let $\boldsymbol{r_t}(s,t) := \frac{\partial \boldsymbol{r}}{\partial t}(s,t)$ and $\boldsymbol{r_{tt}}(s,t) := \frac{\partial^2 \boldsymbol{r}}{\partial t^2}(s,t)$ represent the first and second time derivatives of $\boldsymbol{r}(s,t)$, respectively. This means that $\boldsymbol{r}(0,t)$, $\boldsymbol{r_t}(0,t)$ and $\boldsymbol{r_{tt}}(0,t)$ are the position, velocity and acceleration of the first boundary point, while $\boldsymbol{r}(L,t)$, $\boldsymbol{r_t}(L,t)$ and $\boldsymbol{r_{tt}}(L,t)$ are those of the distal boundary.
In a similar manner, $\boldsymbol{r_s}(s,t) := \frac{\partial \boldsymbol{r}}{\partial s}(s,t)$ is the tangent vector to the string at the point $\boldsymbol{r}(s,t)$.

Consider a string material segment (s_0, s_1), with $0 < s_0 < s_1 < L$, subject to the contact forces (exerted by the rest of the string) $-\boldsymbol{n}(s_0,t)$ and $\boldsymbol{n}(s_1,t)$ and to the body force $\int_{s_0}^{s_1} \boldsymbol{\beta}(s,t)ds$. Below, the integral equation expresses the conservation of linear momentum for the material segment. Specifically, the time derivative of its total momentum equals the net force acting on it[6]:

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{s_0}^{s_1} \rho_c A \boldsymbol{r_t}(s,t)\mathrm{d}s = \int_{s_0}^{s_1} \boldsymbol{\beta}(s,t)\mathrm{d}s - \boldsymbol{n}(s_0,t) + \boldsymbol{n}(s_1,t), \quad \forall (s_0,s_1) \subset (0,L), \quad \forall t \geq 0, \tag{2.8}$$

where ρ_c is the density of the string and A is its cross-sectional area, both positive scalars. Differentiation returns the classical form:

$$\rho_c A \boldsymbol{r_{tt}}(s,t) = \boldsymbol{\beta}(s,t) + \boldsymbol{n_s}(s,t), \quad \forall s \in (0,L), \quad \forall t \geq 0. \tag{2.9}$$

Because of the assumption of perfect flexibility, the constitutive equation links its deformation to the contact forces through:

$$\boldsymbol{n}(s,t) = \|\boldsymbol{n}(s,t)\| \frac{\boldsymbol{r_s}(s,t)}{\|\boldsymbol{r_s}(s,t)\|},$$

with $\|\cdot\|$ denoting the L^2-norm (equivalent to the Euclidean norm for vectors).
Here, $\|\boldsymbol{n}(s,t)\|$ is the string tension (in units of force), which, according to [36], is:

$$\|\boldsymbol{n}(s,t)\| = EA(\|\boldsymbol{r_s}(s,t)\| - 1),$$

where E is the Young modulus of the string and the term $\|\boldsymbol{r_s}(s,t)\| - 1$ is called the strain.
Altogether, the expression for the contact force becomes:

$$\boldsymbol{n}(s,t) = EA\left(1 - \frac{1}{\|\boldsymbol{r_s}(s,t)\|}\right)\boldsymbol{r_s}(s,t). \tag{2.10}$$

To guarantee the well-posedness of the previous problem — meaning that the problem has a unique solution whose behavior changes continuously with the initial conditions — appropriate boundary conditions are necessary. For instance, dynamic boundary conditions involve motion or forces acting on the boundary, as in:

$$m_b \boldsymbol{r_{tt}}(0,t) = \boldsymbol{n}(0,t) \quad \text{or} \quad m_p \boldsymbol{r_{tt}}(L,t) = -\boldsymbol{n}(L,t), \quad m_b, m_p > 0.$$

These are crucial for modeling realistic interactions with other objects at the endpoints of the string.

[6]This formulation corresponds to the integral form of Newton's second law for a one-dimensional continuum.

Finite-difference discretization An effective approach to solve (and simulate) the string PDEs is to use a finite-difference method [23]. The idea is to discretize the domain $[0, L]$ in N intervals of length $h_s \coloneqq \frac{L}{N}$, hence describing the motion of the string with the position, velocity and acceleration of $N+1$ *nodes*, referred to as $\boldsymbol{r^i}(t) \coloneqq \boldsymbol{r}(ih_s, t)$, $\boldsymbol{r_t^i}(t) \coloneqq \boldsymbol{r_t}(ih_s, t)$ and $\boldsymbol{r_{tt}^i}(t) \coloneqq \boldsymbol{r_{tt}}(ih_s, t)$, respectively, $\forall i = 0, \dots, N$.
With analogous notation, the spatial derivatives are approximated using the central differencing scheme:

$$\boldsymbol{r_s^i}(t) \coloneqq \boldsymbol{r_s}(ih_s, t) \approx \frac{\boldsymbol{r^{i+1}}(t) - \boldsymbol{r^{i-1}}(t)}{2h_s}, \quad \forall i = 1, \dots, N-1,$$
$$\boldsymbol{r_{ss}^i}(t) \coloneqq \boldsymbol{r_{ss}}(ih_s, t) \approx \frac{\boldsymbol{r^{i+1}}(t) - 2\boldsymbol{r^i}(t) + \boldsymbol{r^{i-1}}(t)}{h_s^2}, \quad \forall i = 1, \dots, N-1.$$

Similarly, given $\boldsymbol{n^i}(t) \coloneqq \boldsymbol{n}(ih_s, t), \forall i = 1, \dots, N-1$, the derivative $\boldsymbol{n_s}(s, t)$ can be approximated as:

$$\boldsymbol{n_s^i}(t) \coloneqq \boldsymbol{n_s}(ih_s, t) \approx EA\Bigg(\frac{\boldsymbol{r^{i+1}}(t) - 2\boldsymbol{r^i}(t) + \boldsymbol{r^{i-1}}(t)}{h_s^2} + \\ + \frac{1}{h_s}\frac{\boldsymbol{r^i}(t) - \boldsymbol{r^{i-1}}(t)}{\|\boldsymbol{r^i}(t) - \boldsymbol{r^{i-1}}(t)\|} - \frac{1}{h_s}\frac{\boldsymbol{r^{i+1}}(t) - \boldsymbol{r^i}(t)}{\|\boldsymbol{r^{i+1}}(t) - \boldsymbol{r^i}(t)\|}\Bigg), \quad \forall i = 1, \dots, N-1.$$

On the other hand, spatial integrals can be approximated using the trapezoidal rule [3]:

$$\int_{N_0 h_s}^{N_1 h_s} \beta(s)\mathrm{d}s \approx \frac{h_s}{2} \sum_{i=N_0+1}^{N_1} \left(\beta((i-1)h_s) + \beta(ih_s)\right), \\ \forall N_0, N_1 \in \mathbb{N} \text{ s.t. } 0 \leq N_0 < N_1 \leq N.$$

Targeting (2.9), the result is a system of $3(N+1)$ second-order ordinary differential equations that can be solved using standard numerical integration methods, such as Runge-Kutta's. On the interior of the domain, the discretized version of the equations of motion reads:

$$\rho_c A \boldsymbol{r_{tt}^i}(t) = \boldsymbol{\beta}^i(t) + \boldsymbol{n_s^i}(t), \quad \forall i = 1, \dots, N-1,$$

with $\boldsymbol{\beta}^i(t) \coloneqq \boldsymbol{\beta}(ih_s, t), \forall i = 1, \dots, N-1$.
At the boundaries, on the other hand, the equations depend on the specific boundary conditions imposed.

2.4 Quadrotor platform

Rigid-body dynamics To understand and control the motion of a quadrotor, it is essential to model the vehicle as a rigid body subject to external forces and torques. This thesis adopts the Newton-Euler framework for rigid body dynamics, which is widely used in robotics and multibody systems [30, 37].

At all times, the pose of the quadrotor is uniquely determined by the position of its center of mass and its orientation, both with respect to the inertial frame $\mathcal{F}_w$. Let a body-fixed frame $\mathcal{F}_b = \{\boldsymbol{o}, \boldsymbol{e_x}, \boldsymbol{e_y}, \boldsymbol{e_z}\}$ be associated with the quadrotor; the origin $\boldsymbol{o}$ shall

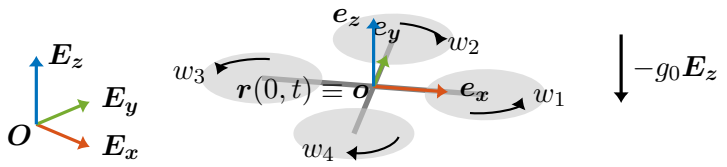


Figure 2.3. Quadrotor – equipped with its body frame $\mathcal{F}_b$ – and gravity vector in the Euclidean space.

coincide with its center of mass (see Fig. 2.3).
For simplicity, it is assumed that the center of mass is connected to the first end of the cable at all times, therefore for the motion of the UAV in the inertial frame one can adopt the notation of position $\boldsymbol{r^0}(t)$, velocity $\boldsymbol{r_t^0}(t)$, and acceleration $\boldsymbol{r_{tt}^0}(t)$.
Correspondingly, at all times the orientation of the robot is described by the rotation matrix $\boldsymbol{R_b}(t) \in SO(3)$, allowing to avoid the singularities introduced by the Euler/Cardan angles. Its angular velocity, expressed in the body frame, is represented by $\boldsymbol{\omega_b}(t) \in \mathbb{R}^3$, and it satisfies:

$$\dot{\boldsymbol{R}}_{\boldsymbol{b}}(t) \coloneqq \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{R_b}(t) = \boldsymbol{R_b}(t)\boldsymbol{S}(\boldsymbol{\omega_b}(t)),$$

where $\boldsymbol{S} : \mathbb{R}^3 \to \mathfrak{so}(3)$ is the well-known skew-symmetric map [9]. $SO(3)$ and $\mathfrak{so}(3)$ are the special orthogonal group and its Lie algebra, respectively.

For the translation dynamics, one has:

$$m_b \boldsymbol{r_{tt}^0}(t) = \boldsymbol{R_b}(t) \sum_{j=1}^{4} \boldsymbol{f_j}(t) - m_b g_0 \boldsymbol{E_z}, \tag{2.11}$$

meaning that the quadrotor, of mass m_b, translates as result of the thrust force generated by the four propellers and the gravity force, with $g_0 = 9.81\,\mathrm{m\,s^{-2}}$. The j-th propeller produces a thrust $\boldsymbol{f_j}(t) = c_f w_j^2(t)\boldsymbol{e_z}$, rotating with angular rate $w_j(t)$ and with thrust coefficient c_f. Note that drag is neglected because it is supposed to be countered by the control action. Obviously, $\boldsymbol{e_z} = \begin{bmatrix}0 & 0 & 1\end{bmatrix}^\top$ in the body frame and $\boldsymbol{E_z} = \begin{bmatrix}0 & 0 & 1\end{bmatrix}^\top$ in the world frame.
On the same note, Euler's rotation equation reads:

$$\boldsymbol{J_b}\dot{\boldsymbol{\omega}}_{\boldsymbol{b}}(t) + \boldsymbol{\omega_b}(t) \times \boldsymbol{J_b}\boldsymbol{\omega_b}(t) = \sum_{j=1}^{4} \boldsymbol{\tau_j}(t), \tag{2.12}$$

with $\dot{\boldsymbol{\omega}}_{\boldsymbol{b}}(t) \coloneqq \frac{\mathrm{d}\boldsymbol{\omega_b}}{\mathrm{d}t}(t)$ and $\boldsymbol{J_b}$ being the inertia matrix of the quadrotor. Here, $\boldsymbol{\tau_j}(t)$ is the moment generated by the j-th propeller, given by $\boldsymbol{\tau_j}(t) = \boldsymbol{p_j} \times \boldsymbol{f_j}(t) + (-1)^j c_\tau w_j^2(t)\boldsymbol{e_z}$, which gathers both the contribution due to the thrust force and the one due to the drag force, with $\boldsymbol{p_j}$ being the position of the j-th propeller in body coordinates and c_τ being its torque coefficient.

Geometric control on SE(3) The following implementation – whose block scheme is drawn in Fig. 2.4 – is based on [20] and [37].

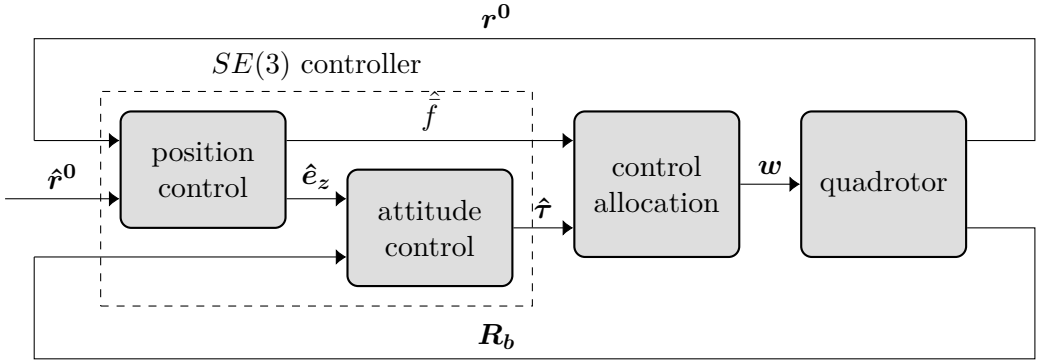


Figure 2.4. Three-layer control architecture for full-pose quadrotor regulation. The desired heading direction is fixed (see Fn. 7).

Defining the quantities:

$$\bar{f}(t) \coloneqq c_f \sum_{j=1}^{4} w_j^2(t),$$

$$\boldsymbol{\tau}(t) \coloneqq \sum_{j=1}^{4} \left(\boldsymbol{p_j} \times c_f w_j^2(t) \boldsymbol{e_z} + (-1)^j c_\tau w_j^2(t) \boldsymbol{e_z} \right)$$

that appear in the quadrotor dynamics in Sec. 2.4, assuming no transients for the propeller angular velocities, it holds that:

$$\begin{bmatrix} \bar{f}(t) \\ \boldsymbol{\tau}(t) \end{bmatrix} = \begin{bmatrix} c_f & c_f & c_f & c_f \\ p_{1,y} c_f & p_{2,y} c_f & p_{3,y} c_f & p_{4,y} c_f \\ -p_{1,x} c_f & -p_{2,x} c_f & -p_{3,x} c_f & -p_{4,x} c_f \\ -c_\tau & c_\tau & -c_\tau & c_\tau \end{bmatrix} \begin{bmatrix} w_1^2(t) \\ w_2^2(t) \\ w_3^2(t) \\ w_4^2(t) \end{bmatrix}. \tag{2.13}$$

Here, it is assumed that the rotors are placed in the transverse plane of the vehicle, implying that $\boldsymbol{p_j} = \begin{bmatrix} p_{j,x} & p_{j,y} & 0 \end{bmatrix}^\top, \forall j = 1, \ldots, 4$.

Considering the effect of the motion of the cable as a disturbance, one would solve the tracking problem compensating for gravity and correcting the position tracking error $\boldsymbol{\varepsilon_p^0}(t) \coloneqq \boldsymbol{r^0}(t) - \hat{\boldsymbol{r}}^{\boldsymbol{0}}(t)$ and the velocity tracking error $\boldsymbol{\varepsilon_v^0}(t) \coloneqq \boldsymbol{r_t^0}(t) - \hat{\boldsymbol{r}}_{\boldsymbol{t}}^{\boldsymbol{0}}(t)$, possibly with a feedforward term:

$$\bar{f}(t) \boldsymbol{R_b}(t) \boldsymbol{e_z} \to m_0 \left(g_0 \boldsymbol{E_z} - \boldsymbol{K_p} \boldsymbol{\varepsilon_p^0}(t) - \boldsymbol{K_v} \boldsymbol{\varepsilon_v^0}(t) + \hat{\boldsymbol{r}}_{\boldsymbol{tt}}^{\boldsymbol{0}}(t) \right) =: \hat{\boldsymbol{f}}(t), \tag{2.14}$$

where $\boldsymbol{K_p}$ and $\boldsymbol{K_v}$ are positive definite gain matrices and $\hat{\boldsymbol{r}}^0(t)$ is the desired trajectory, with its time derivatives $\hat{\boldsymbol{r}}_{\boldsymbol{t}}^{\boldsymbol{0}}(t)$ and $\hat{\boldsymbol{r}}_{\boldsymbol{tt}}^{\boldsymbol{0}}(t)$. Notice that $\hat{\boldsymbol{f}}(t)$ is proportional to the mass of the whole system, that is $m_0 > 0$. For a stand-alone quadrotor, $m_0 = m_b$.
The thrust-level control is enough only when the vertical axis of the body frame is aligned with the vector on the right hand side of the equation; the best that can be done for a generic orientation is to make an orthogonal projection:

$$\hat{\bar{f}}(t) \coloneqq \left\langle \hat{\boldsymbol{f}}(t), \boldsymbol{R_b}(t) \boldsymbol{e_z} \right\rangle, \tag{2.15}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product in $\mathbb{R}^3$.

The control law is completed by imparting a torque that makes the vertical axis of the quadrotor align with the direction of the desired thrust $\hat{\boldsymbol{e}}_{\boldsymbol{z}}(t) = \frac{\hat{\boldsymbol{f}}(t)}{\|\hat{\boldsymbol{f}}(t)\|}$ quickly enough. The desired orientation is encoded in the rotation matrix $\hat{\boldsymbol{R}}_{\boldsymbol{b}}(t) \in SO(3)$, namely:

$$\hat{\boldsymbol{R}}_{\boldsymbol{b}}(t) \coloneqq \begin{bmatrix} \frac{\hat{\boldsymbol{e}}_{\boldsymbol{y}}(t) \times \hat{\boldsymbol{e}}_{\boldsymbol{z}}(t)}{\|\hat{\boldsymbol{e}}_{\boldsymbol{y}}(t) \times \hat{\boldsymbol{e}}_{\boldsymbol{z}}(t)\|} & \hat{\boldsymbol{e}}_{\boldsymbol{y}}(t) & \hat{\boldsymbol{e}}_{\boldsymbol{z}}(t) \end{bmatrix}. \tag{2.16}$$

Keep in mind that $\hat{\boldsymbol{e}}_{\boldsymbol{y}}(t)$ must be chosen in such a way that $\boldsymbol{R}_{\boldsymbol{b}}(t)$ is orthonormal. Given a desired heading direction[7] $\hat{\boldsymbol{e}}_{\boldsymbol{x}}(t)$ – expressed in the world frame, just like the other vectors in the columns of $\boldsymbol{R}_{\boldsymbol{b}}(t)$ –, one could pick:

$$\hat{\boldsymbol{e}}_{\boldsymbol{y}}(t) \coloneqq \frac{\hat{\boldsymbol{e}}_{\boldsymbol{z}}(t) \times \hat{\boldsymbol{e}}_{\boldsymbol{x}}(t)}{\|\hat{\boldsymbol{e}}_{\boldsymbol{z}}(t) \times \hat{\boldsymbol{e}}_{\boldsymbol{x}}(t)\|}. \tag{2.17}$$

The attitude control term is chosen such that the gyroscopic effect is compensated and the desired orientation is reached by correcting the attitude error $\boldsymbol{\varepsilon}_{\boldsymbol{R}}(t) \coloneqq \frac{1}{2} S^{-1}(\hat{\boldsymbol{R}}_{\boldsymbol{b}}^{\top}(t)\boldsymbol{R}_{\boldsymbol{b}}(t) - \boldsymbol{R}_{\boldsymbol{b}}^{\top}(t)\hat{\boldsymbol{R}}_{\boldsymbol{b}}(t))$ and the angular velocity error $\boldsymbol{\varepsilon}_{\boldsymbol{\omega}}(t) \coloneqq \boldsymbol{\omega}_{\boldsymbol{b}}(t) - \boldsymbol{R}_{\boldsymbol{b}}^{\top}(t)\hat{\boldsymbol{R}}_{\boldsymbol{b}}(t)\hat{\boldsymbol{\omega}}_{\boldsymbol{b}}(t)$ (where $\hat{\boldsymbol{\omega}}_b(t)$ is the desired angular velocity):

$$\begin{aligned} \hat{\boldsymbol{\tau}}(t) \coloneqq \boldsymbol{\omega}_{\boldsymbol{b}}(t) \times \boldsymbol{J}_{\boldsymbol{b}}\boldsymbol{\omega}_{\boldsymbol{b}}(t) + \boldsymbol{J}_{\boldsymbol{b}}(\boldsymbol{R}_{\boldsymbol{b}}^{\top}(t)\hat{\boldsymbol{R}}_{\boldsymbol{b}}(t)\dot{\hat{\boldsymbol{\omega}}}_{\boldsymbol{b}}(t) - S(\boldsymbol{\omega}_{\boldsymbol{b}}(t))\boldsymbol{R}_{\boldsymbol{b}}^{\top}(t)\hat{\boldsymbol{R}}_{\boldsymbol{b}}(t)\hat{\boldsymbol{\omega}}_{\boldsymbol{b}}(t) + \\ - \boldsymbol{K}_{\boldsymbol{R}}\boldsymbol{\varepsilon}_{\boldsymbol{R}}(t) - \boldsymbol{K}_{\boldsymbol{\omega}}\boldsymbol{\varepsilon}_{\boldsymbol{\omega}}(t)), \end{aligned} \tag{2.18}$$

with $\dot{\hat{\boldsymbol{\omega}}}_{\boldsymbol{b}}(t) \coloneqq \frac{d\hat{\boldsymbol{\omega}}_{\boldsymbol{b}}(t)}{dt}$, and $\boldsymbol{K}_{\boldsymbol{R}}$ and $\boldsymbol{K}_{\boldsymbol{\omega}}$ being positive definite gain matrices.

Once all the quantities at stake have been derived, the complete control law can be stated based on the inversion of the control allocation matrix introduced in (2.13), using the control force (2.15) with (2.14), and the control torque (2.18) with (2.16) and (2.17):

$$\boldsymbol{w}(t) = \begin{bmatrix} w_1(t) \\ w_2(t) \\ w_3(t) \\ w_4(t) \end{bmatrix} = \sqrt{\begin{bmatrix} c_f & c_f & c_f & c_f \\ p_{1,y}c_f & p_{2,y}c_f & p_{3,y}c_f & p_{4,y}c_f \\ -p_{1,x}c_f & -p_{2,x}c_f & -p_{3,x}c_f & -p_{4,x}c_f \\ -c_\tau & c_\tau & -c_\tau & c_\tau \end{bmatrix}^{-1} \begin{bmatrix} \hat{\bar{f}}(t) \\ \hat{\boldsymbol{\tau}}(t) \end{bmatrix}}.$$

In the latter, the square root operator is employed component-wise.

[7] In this work, this extra degree of freedom is resolved by fixing $\hat{\boldsymbol{e}}_{\boldsymbol{x}}(t) = [1\ 0\ 0]^{\top}$, which corresponds to a constant desired heading aligned with the world-frame x-axis.

Part II

Methodology

Chapter 3

Modeling and reduction of the aerial system

3.1 String model order reduction[1]

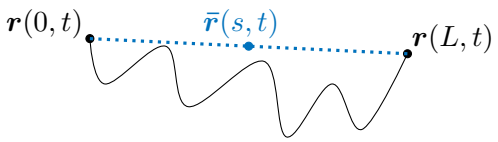


Figure 3.1. Decomposition of the string position map on which the proposed model-order reduction method is based.

Boundary-constrained reduced basis Consider the string PDEs in Sec. 2.3 with appropriate boundary conditions[2]. There is a need to realize a decomposition of the functional space $C^2([0,L],\mathbb{R}^3)$, enabling the isolation of the cable endpoint dynamics. This requires finding a linear subspace of $C^2([0,L],\mathbb{R}^3)$, say Ω, characterized by functions with the property of vanishing at the boundaries of the domain. Hence, the original functional space is decomposed in Ω and its algebraic complement.

With only the knowledge of the solution at the boundaries (i.e. $\boldsymbol{r}(0,t)$ and $\boldsymbol{r}(L,t)$), naturally one would place each material point $s \in [0,L]$ on the fictitious line that connects the two extremities (see Fig. 3.1). This configuration is referred to as the *reference configuration*.
The corresponding map $\bar{\boldsymbol{r}}$ is such that:

$$\bar{\boldsymbol{r}}(s,t) \coloneqq \boldsymbol{r}(0,t) + \frac{s}{L}(\boldsymbol{r}(L,t) - \boldsymbol{r}(0,t)) = \left(1 - \frac{s}{L}\right)\boldsymbol{r}(0,t) + \frac{s}{L}\boldsymbol{r}(L,t), \quad \forall s \in [0,L]. \tag{3.1}$$

The plan is to decompose the position map according to:

$$\tilde{\boldsymbol{r}}(s,t) \coloneqq \boldsymbol{r}(s,t) - \bar{\boldsymbol{r}}(s,t) \implies \boldsymbol{r}(s,t) = \bar{\boldsymbol{r}}(s,t) + \tilde{\boldsymbol{r}}(s,t), \tag{3.2}$$

[1] Before building a mathematical model for the specific aerial system under study, this section presents the general reduced-order modeling technique for a class of continuum systems based on the string PDE formulation introduced in Sec. 2.3.

[2] To emphasise the general applicability of the proposed method, note that it is not restricted to dynamic boundary conditions; it can also be employed under Neumann, Dirichlet, or Robin boundary conditions.

with $\tilde{\boldsymbol{r}}(t,\cdot) \in \Omega$ being the residual term, namely the deviation of the material point s from the reference configuration.
The essential feature of this decomposition is that the residual component satisfies the homogeneous boundary condition, i.e. it vanishes at the boundaries as required:

$$\tilde{\boldsymbol{r}}(0,t) = \tilde{\boldsymbol{r}}(L,t) = \boldsymbol{0}, \quad \forall t \geq 0. \tag{3.3}$$

The part that follows is closely related to the introduction on model-order reduction in Sec. 2.2, therefore the reader is referred to it for a more detailed explanation of the steps involved.
Adopting the usual notation $\tilde{\boldsymbol{r}}^{\boldsymbol{i}}(t) \coloneqq \tilde{\boldsymbol{r}}(ih_s,t)$, construct a tensor of snapshots $\tilde{\boldsymbol{\mathcal{R}}} \in \mathbb{R}^{3\times(M+1)\times(O+1)}$ such that:

$$\tilde{\boldsymbol{\mathcal{R}}}_{::\boldsymbol{j}} \coloneqq \begin{bmatrix} \tilde{\boldsymbol{r}}^{\boldsymbol{0}}(t_j) & \tilde{\boldsymbol{r}}^{\boldsymbol{d}}(t_j) & \tilde{\boldsymbol{r}}^{\boldsymbol{2d}}(t_j) & \dots & \tilde{\boldsymbol{r}}^{\boldsymbol{N}}(t_j) \end{bmatrix}, \quad \forall j = 0,\dots,O,$$

with $d \coloneqq \frac{N}{M} \in \mathbb{N}$ serving as a factor for uniform spatial decimation[3].
Following the POD procedure, the HOSVD of the snapshot tensor returns the unitary matrix:

$$\boldsymbol{U_2} = \begin{bmatrix} \boldsymbol{u_1} & \boldsymbol{u_2} & \dots & \boldsymbol{u_m} & \dots & \boldsymbol{u_{M+1}} \end{bmatrix}.$$

Notice that, because of (3.3), the first and last components of $\boldsymbol{u_m}$ are always null.
The actual modes are obtained by scaling the columns of $\boldsymbol{U_2}$:

$$\boldsymbol{\phi_m} = \begin{bmatrix} \phi_m^0 & \phi_m^d & \dots & \phi_m^N \end{bmatrix}^\top = \varphi \boldsymbol{u_m}, \quad \forall m = 1,\dots,M+1.$$

The weight φ is chosen to ensure orthonormality of the modes while accounting for the spatial sampling. For this, using trapezoidal integration in (2.6) over $[0,L]$ with step $h_d = h_s d$, the objective is to satisfy the identity:

$$\delta_{jk} = \int_0^L \phi_j(s)\phi_k(s)ds \approx h_d \boldsymbol{\phi_j}^\top \boldsymbol{\phi_k} = h_d \varphi^2 \boldsymbol{u_j}^\top \boldsymbol{u_k} = h_d \varphi^2 \delta_{jk} \iff \varphi = \sqrt{\frac{1}{h_d}}. \tag{3.4}$$

Note that condition (3.3) further limits the maximum number of modes that can be found to $\min\{M-1, 3(O+1)\}$.

Projection onto the reduced subspace The idea is to project the finite-dimensional model obtained via FDM onto the space spanned by the subset of $R \leq M-1$ orthonormal components whose energy, measured in terms of the magnitude of their corresponding singular values, is the largest.

Using the linear combination formula (2.7), in the (undersampled) lower-dimensional space, (3.2) and its time derivatives can be approximated as:

$$\begin{aligned}
\boldsymbol{r^i}(t) &\approx \bar{\boldsymbol{r}}^{\boldsymbol{i}}(t) + \sum_{m=1}^{R} \boldsymbol{a}_m(t)\phi_m^i, \quad \forall i = d, 2d, \dots, N-d, \\
\boldsymbol{r_t^i}(t) &\approx \bar{\boldsymbol{r}}_{\boldsymbol{t}}^{\boldsymbol{i}}(t) + \sum_{m=1}^{R} \dot{\boldsymbol{a}}_m(t)\phi_m^i, \quad \forall i = d, 2d, \dots, N-d, \\
\boldsymbol{r_{tt}^i}(t) &\approx \bar{\boldsymbol{r}}_{\boldsymbol{tt}}^{\boldsymbol{i}}(t) + \sum_{m=1}^{R} \ddot{\boldsymbol{a}}_m(t)\phi_m^i, \quad \forall i = d, 2d, \dots, N-d,
\end{aligned} \tag{3.5}$$

[3]This assumes that M is chosen such that N is divisible by M.

where $\bar{\boldsymbol{r}}^{\boldsymbol{i}}(t) \coloneqq \bar{\boldsymbol{r}}(ih_s, t)$, $\bar{\boldsymbol{r}}^{\boldsymbol{i}}_{\boldsymbol{t}}(t) \coloneqq \bar{\boldsymbol{r}}_{\boldsymbol{t}}(ih_s, t)$, and $\bar{\boldsymbol{r}}^{\boldsymbol{i}}_{\boldsymbol{tt}}(t) \coloneqq \bar{\boldsymbol{r}}_{\boldsymbol{tt}}(ih_s, t)$.
The two endpoint positions $\boldsymbol{r^0}(t)$ and $\boldsymbol{r^N}(t)$ and their derivatives, whose computation is crucial because they appear in $\bar{\boldsymbol{r}}^{\boldsymbol{i}}(t)$ and their time derivatives according to (3.1), live in their original subspaces and are not affected by projection.

To obtain the reduced-order dynamics, $\tilde{\boldsymbol{r}}(s,t)$ and its derivatives are projected onto the chosen R-dimensional subspace of Ω taking advantage of the orthonormality condition, employing the trapezoidal rule in accordance with (3.4):

$$\begin{aligned}
\boldsymbol{a}_m(t) &= \int_0^L (\boldsymbol{r}(s,t) - \bar{\boldsymbol{r}}(s,t))\phi_m(s)ds \approx h_d \sum_{\substack{i=d \\ \text{step } d}}^{N-d} (\boldsymbol{r}^{\boldsymbol{i}}(t) - \bar{\boldsymbol{r}}^{\boldsymbol{i}}(t))\phi^i_m, \\
\dot{\boldsymbol{a}}_m(t) &= \int_0^L (\boldsymbol{r_t}(s,t) - \bar{\boldsymbol{r}}_{\boldsymbol{t}}(s,t))\phi_m(s)ds \approx h_d \sum_{\substack{i=d \\ \text{step } d}}^{N-d} (\boldsymbol{r}^{\boldsymbol{i}}_{\boldsymbol{t}}(t) - \bar{\boldsymbol{r}}^{\boldsymbol{i}}_{\boldsymbol{t}}(t))\phi^i_m, \\
\ddot{\boldsymbol{a}}_m(t) &= \int_0^L (\boldsymbol{r_{tt}}(s,t) - \bar{\boldsymbol{r}}_{\boldsymbol{tt}}(s,t))\phi_m(s)ds \approx h_d \sum_{\substack{i=d \\ \text{step } d}}^{N-d} (\boldsymbol{r}^{\boldsymbol{i}}_{\boldsymbol{tt}}(t) - \bar{\boldsymbol{r}}^{\boldsymbol{i}}_{\boldsymbol{tt}}(t))\phi^i_m.
\end{aligned} \tag{3.6}$$

In the last equation, the accelerations of the string nodes $\boldsymbol{r}^{\boldsymbol{i}}_{\boldsymbol{tt}}(t), \forall i = d, \dots, N-d$ are taken from the equations of motion in Sec. 2.3, and the left-hand terms $\ddot{\boldsymbol{a}}_{\boldsymbol{m}}(t)$ are integrated into the lower-order derivatives $\dot{\boldsymbol{a}}_{\boldsymbol{m}}(t)$ and $\boldsymbol{a}_{\boldsymbol{m}}(t)$, which compose the original maps through (3.5).

It should also be stressed that, in a more compact form, (3.5) can be used to express the matrix relationship:

$$\begin{bmatrix} \tilde{\boldsymbol{r}}^{\boldsymbol{0}}(t) \\ \tilde{\boldsymbol{r}}^{\boldsymbol{d}}(t) \\ \vdots \\ \tilde{\boldsymbol{r}}^{\boldsymbol{N-d}}(t) \\ \tilde{\boldsymbol{r}}^{\boldsymbol{N}}(t) \end{bmatrix} = \boldsymbol{\Phi} \begin{bmatrix} \boldsymbol{a_1}(t) \\ \boldsymbol{a_2}(t) \\ \vdots \\ \boldsymbol{a_R}(t) \end{bmatrix}, \quad \boldsymbol{\Phi} \coloneqq \begin{bmatrix} \boldsymbol{\phi_1} & \boldsymbol{\phi_2} & \dots & \boldsymbol{\phi_R} \end{bmatrix} \otimes \boldsymbol{I_3}, \tag{3.7}$$

where $\otimes$ denotes the Kronecker product and $\boldsymbol{I_n}, \forall n \in \mathbb{N}$ is the $n \times n$ identity matrix.
Conversely, the projection onto the reduced subspace can be performed using the Moore-Penrose pseudoinverse $\boldsymbol{\Phi}^+$ of $\boldsymbol{\Phi}$:

$$\begin{bmatrix} \boldsymbol{a_1}(t) \\ \boldsymbol{a_2}(t) \\ \vdots \\ \boldsymbol{a_R}(t) \end{bmatrix} = \boldsymbol{\Phi}^+ \begin{bmatrix} \tilde{\boldsymbol{r}}^{\boldsymbol{0}}(t) \\ \tilde{\boldsymbol{r}}^{\boldsymbol{d}}(t) \\ \vdots \\ \tilde{\boldsymbol{r}}^{\boldsymbol{N-d}}(t) \\ \tilde{\boldsymbol{r}}^{\boldsymbol{N}}(t) \end{bmatrix}.$$

A straightforward implication is that, if one marks with the symbol $'$ an alternative base, whose elements are represented through the matrix $\boldsymbol{\Phi'}$ and where the reduced-order dynamics evolves through the coefficients $\boldsymbol{a'_m}(t)$, $\dot{\boldsymbol{a}}'_{\boldsymbol{m}}(t)$, $\ddot{\boldsymbol{a}}'_{\boldsymbol{m}}(t)$, then it is possible to

identify the corresponding coordinate-transformation map:

$$\begin{bmatrix} \boldsymbol{a_1}(t) \\ \boldsymbol{a_2}(t) \\ \vdots \\ \boldsymbol{a_R}(t) \end{bmatrix} = \boldsymbol{\Phi}^{+}\boldsymbol{\Phi'} \begin{bmatrix} \boldsymbol{a'_1}(t) \\ \boldsymbol{a'_2}(t) \\ \vdots \\ \boldsymbol{a'_R}(t) \end{bmatrix}. \tag{3.8}$$

The extension to the time derivatives is straightforward, since the mode matrices $\boldsymbol{\Phi}$ and $\boldsymbol{\Phi'}$ are constant in time. This is crucial for hybrid transitions between different reduced-order bases – for instance, in Sec. 4.1.

In conclusion, with the proposed method one moves from the original FDM state space $\mathbb{R}^{6(N+1)}$ to the lower-dimensional space $\mathbb{R}^{6(R+2)}$ – in fact, there are 2 boundary points and R modal coefficients. The higher R, the more accurate the approximation, but the larger the state dimension.

3.2 System description and assumptions

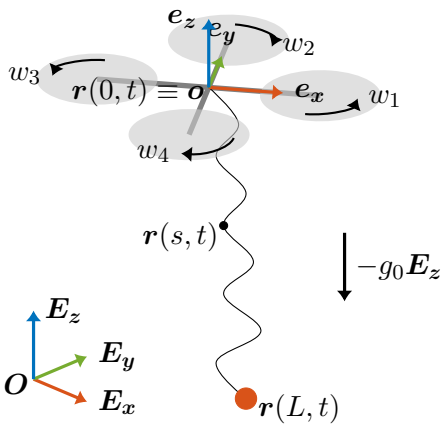


Figure 3.2. Aerial system composed of a quadrotor with a flexible cable with a payload attached to the distal end.

Kinematics As anticipated, the aerial system is composed of a quadrotor connected to a flexible cable. At the distal end, the cable can either be free or attached to a payload. Please refer to Fig. 3.2 for a visual representation of the system.
For the model of the flexible cable, the reader should consider the same setting described in Sec. 2.3. Also reprise the quadrotor model introduced in Sec. 2.4, recalling that the origin $\boldsymbol{o}$ of frame $\mathcal{F}_b$, i.e. the center of mass of the UAV, corresponds identically to either end of the cable. The other major point of interest is the distal end, namely the tip, which can be either free or attached to the payload. The payload is treated as a point mass, eliminating the need for additional degrees of freedom or a dedicated reference frame. Its position, velocity and acceleration in $\mathcal{F}_w$ are $\boldsymbol{r}(L,t)$, $\boldsymbol{r_t}(L,t)$, and $\boldsymbol{r_{tt}}(L,t)$, respectively.

Modeling assumptions The following lays out the main assumptions made in this thesis regarding the aerial system[4]:

- The air is assumed to be homogeneous with constant properties (e.g. density). It is considered static, and all the aerodynamic disturbances (e.g. downwash due to the propellers) are neglected except for a simplified drag force.
- Propeller and motor dynamics are assumed to be instantaneous.
- As anticipated in Sec. 2.3, the cable is modeled as an extensible string with perfect flexibility, with no internal damping nor self-interaction (e.g. self-collisions).
- The influence of the end-effector in the pick-and-place operation is neglected. Instead, the transition is modeled as an instantaneous inelastic collision, abstracting away any physical mechanisms (e.g. gripper).

3.3 Infinite-dimensional model

Cable During its motion, every infinitesimal element of the cable is subject to gravitational and aerodynamic drag force densities:

$$\boldsymbol{\beta}(s,t) = -b_c \|\boldsymbol{r_t}(s,t)\| \boldsymbol{r_t}(s,t) - \rho_c A g_0 \boldsymbol{E_z}, \tag{3.9}$$

where b_c is the drag coefficient of the cable.
Combining the classical equations of motion (2.9) and (3.9) yields the complete system of PDEs for the cable dynamics:

$$\rho_c A \boldsymbol{r_{tt}}(s,t) = -\rho_c A g_0 \boldsymbol{E_z} - b_c \|\boldsymbol{r_t}(s,t)\| \boldsymbol{r_t}(s,t) + \boldsymbol{n_s}(s,t), \quad \forall s \in (0,L). \tag{3.10}$$

Regarding the two boundary conditions, while the extremity at $s = 0$ is attached to the quadrotor, the other at $s = L$ can be either free or connected to the payload.

Quadrotor The quadrotor dynamics, already dealt with in Sec. 2.4, shall influence and be influenced by the cable dynamics.

An external force term equal to the contact force of the attached cable end is introduced in Newton's law (2.11), resulting in:

$$m_b \boldsymbol{r_{tt}}(0,t) = \boldsymbol{R_b}(t) \sum_{j=1}^{4} \boldsymbol{f_j}(t) - m_b g_0 \boldsymbol{E_z} + \boldsymbol{n}(0,t), \tag{3.11}$$

In order to ensure mathematical rigor and foster finer numerical integration in the FDM-based model in Sec. 2.3, computing $\boldsymbol{n}(s,t)$ directly at the boundaries of the domain of s using (2.10) should be avoided (because of $\boldsymbol{r_s}(s,t)$). Considering the integral equation (2.8) with $s_0 \to 0^+$, because of the balance of linear momentum one can write (for continuity):

$$\boldsymbol{n}(0,t) = -\rho_c A s_1 g_0 \boldsymbol{E_z} - \int_0^{s_1} \left(b_c \|\boldsymbol{r_t}(s,t)\| \boldsymbol{r_t}(s,t) + \rho_c A \boldsymbol{r_{tt}}(s,t)\right) ds + \boldsymbol{n}(s_1,t), \quad \forall s_1 \in (0,L) \tag{3.12}$$

[4]Some of these were already taken into account previously, either explicitly or implicitly, but they are reproposed here for clarity.

Because of the assumption of perfect flexibility, the cable does not exert torques on the quadrotor, and its attitude evolves independently according to (2.12).

Tip/payload When nothing is attached to the other end of the cable, the classical form of the dynamics (2.9) is extended to $s = L$, with the constraint that in that point the strain is null [29]:

$$\|\boldsymbol{r_s}(L,t)\| = 1 \implies \boldsymbol{n}(L,t) = 0, \tag{3.13}$$

which enforces an effective boundary constraint that propagates through the internal strain field by continuity.

On the other hand, having a mass hooked to the cable tip implies the assumption of null strain is no longer valid. Instead, similarly to what was done for the quadrotor, the boundary condition at $s = L$ is influenced by the dynamics of the payload, which is described along the lines of (3.11), (obviously) without any thrust contribution and with the addition of a drag term:

$$m_p \boldsymbol{r_{tt}}(L,t) = -\boldsymbol{n}(L,t) - m_p g_0 \boldsymbol{E_z} - b_p \|\boldsymbol{r_t}(L,t)\| \boldsymbol{r_t}(L,t), \tag{3.14}$$

where m_p is the mass of the payload and b_p is its drag coefficient.
Finally, one can take advantage once again of (2.8) with $s_1 \to L^-$ and write:

$$\boldsymbol{n}(L,t) = \boldsymbol{n}(s_0,t) + \rho_c A (L - s_0) g_0 \boldsymbol{E_z} + \int_{s_0}^{L} \left(b_c \|\boldsymbol{r_t}(s,t)\| \boldsymbol{r_t}(s,t) + \rho_c A \boldsymbol{r_{tt}}(s,t)\right) ds, \quad \forall s_0 \in (0,L). \tag{3.15}$$

Partial-differential hybrid automaton Allowing the payload to be hooked and unhooked arbitrarily makes the system at hand a hybrid system (see Sec. 2.1) whose state evolves in the infinite-dimensional space $SO(3) \times \mathfrak{so}(3) \times C^2([0,L],\mathbb{R}^3) \times C^2([0,L],\mathbb{R}^3)$. Therefore, it is approriate to model it using a partial differential hybrid automaton (PDHA) [4].

The PDHA, represented in Fig. 3.3, is composed of two discrete states, each corresponding to one of the two alternative boundary conditions at $s = L$ outlined in Sec. 3.3. The flow map, which takes the rotor angular speeds as control inputs, is defined according to the following equations:

- state $q = 0$ describes the case in which the cable is unloaded according to:

$$\begin{cases} \dot{\boldsymbol{\omega}}_{\boldsymbol{b}}(t) = \boldsymbol{J_b}^{-1} \left(\sum_{j=1}^{4} \boldsymbol{\tau_j}(t) - \boldsymbol{\omega_b}(t) \times \boldsymbol{J_b} \boldsymbol{\omega_b}(t) \right) \\ \boldsymbol{r_{tt}}(0,t) = \frac{1}{m_b} \left(\boldsymbol{R_b}(t) \sum_{j=1}^{4} \boldsymbol{f_j}(t) - m_b g_0 \boldsymbol{E_z} + \boldsymbol{n}(0,t) \right) \\ \boldsymbol{r_{tt}}(s,t) = -g_0 \boldsymbol{E_z} + \frac{1}{\rho_c A} \left(-b_c \|\boldsymbol{r_t}(s,t)\| \boldsymbol{r_t}(s,t) + \boldsymbol{n_s}(s,t) \right), \quad \forall s \in (0,L] \\ \|\boldsymbol{r_s}(L,t)\| = 1 \end{cases} ; \tag{3.16}$$

- state $q = 1$ is the case of the payload attached to the cable, described by:

$$\begin{cases} \dot{\boldsymbol{\omega}}_{\boldsymbol{b}}(t) = \boldsymbol{J}_{\boldsymbol{b}}^{-1}\left(\sum_{j=1}^{4}\boldsymbol{\tau}_{\boldsymbol{j}}(t) - \boldsymbol{\omega}_{\boldsymbol{b}}(t)\times \boldsymbol{J}_{\boldsymbol{b}}\boldsymbol{\omega}_{\boldsymbol{b}}(t)\right) \\ \boldsymbol{r}_{\boldsymbol{tt}}(0,t) = \frac{1}{m_b}\left(\boldsymbol{R}_{\boldsymbol{b}}(t)\sum_{j=1}^{4}\boldsymbol{f}_{\boldsymbol{j}}(t) - m_b g_0 \boldsymbol{E}_{\boldsymbol{z}} + \boldsymbol{n}(0,t)\right) \\ \boldsymbol{r}_{\boldsymbol{tt}}(s,t) = -g_0\boldsymbol{E}_{\boldsymbol{z}} + \frac{1}{\rho_c A}\left(-b_c\|\boldsymbol{r}_{\boldsymbol{t}}(s,t)\|\boldsymbol{r}_{\boldsymbol{t}}(s,t) + \boldsymbol{n}_{\boldsymbol{s}}(s,t)\right), \quad \forall s\in(0,L) \\ \boldsymbol{r}_{\boldsymbol{tt}}(L,t) = \frac{1}{m_p}\left(-\boldsymbol{n}(L,t) - m_p g_0\boldsymbol{E}_{\boldsymbol{z}} - b_p\|\boldsymbol{r}_{\boldsymbol{t}}(L,t)\|\boldsymbol{r}_{\boldsymbol{t}}(L,t)\right) \end{cases} . \tag{3.17}$$

The sources for these are (2.12), (3.10), (3.11), (3.13) and (3.14).
Transitions arise from two possible events:

- **Attachment**. Along edge $(0,1)$, the attachment of the payload to the cable tip is triggered by a geometric condition[5] on the cable tip position $\boldsymbol{r}(L,t)\in\Lambda^{(0,1)}\subset\mathbb{R}^3$. An inelastic collision between the payload and the tip of the cable, because of conservation of linear momentum, yields:

$$\int_0^L \rho_c A\boldsymbol{r}_{\boldsymbol{t}}(s,t^-)\mathrm{d}s + m_p\boldsymbol{v}_{\boldsymbol{p}}(t^-) = \int_0^L \rho_c A\boldsymbol{r}_{\boldsymbol{t}}(s,t^+)\mathrm{d}s + m_p\boldsymbol{v}_{\boldsymbol{p}}(t^+),$$

  with $\boldsymbol{v}_{\boldsymbol{p}}(t)$ being the payload velocity at time t, and with t^- and t^+ being the time instants immediately before and after the transition, respectively.
  It is reasonable to assume that the payload is stationary before the collision, i.e. $\boldsymbol{v}_{\boldsymbol{p}}(t^-) = \boldsymbol{0}$. Afterward, on the other hand, its position coincides with that of the cable tip, therefore – assuming no further switches –, one has $\boldsymbol{v}_{\boldsymbol{p}}(t) = \boldsymbol{r}_{\boldsymbol{t}}(L,t), \forall t \geq t^+$. Since the collision is idealized to occurr at a single point (no spatial extent), it holds that:

$$\int_0^L \rho_c A\boldsymbol{r}_{\boldsymbol{t}}(s,t^-)\mathrm{d}s = \int_0^L \rho_c A\boldsymbol{r}_{\boldsymbol{t}}(s,t^+)\mathrm{d}s.$$

  The conclusion is that this event enforces a velocity reset at the tip:

$$\boldsymbol{r}_{\boldsymbol{t}}(L,t) \leftarrow \boldsymbol{0}. \tag{3.18}$$

- **Detachment**. Vice versa, along edge $(1,0)$, the payload is detached when $\boldsymbol{r}(L,t)\in\Lambda^{(1,0)}\subset\mathbb{R}^3$ without resetting the continuous state.

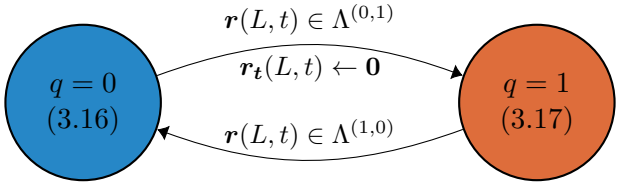


Figure 3.3. State transition diagram of the PDHA that models the UAV-cable-payload system using the infinite-dimensional model.

[5]The treatment of more intricate guard conditions – for instance, the closure of a gripper end-effector – goes beyond the scope of this work, hence it is left as a topic for future research.

3.4 Finite-dimensional model

Cable There should not be any hassle in converting the cable dynamics (3.10) using the finite difference method proposed in Sec. 2.3. After some minor rearrangements, the corresponding set of ordinary differential equations reads:

$$\rho_c A \boldsymbol{r_{tt}^i}(t) = -b_c \|\boldsymbol{r_t^i}(t)\| \boldsymbol{r_t^i}(t) - \rho_c A g_0 \boldsymbol{E_z} + \boldsymbol{n_s^i}(t), \quad \forall i = 1, \dots, N-1. \tag{3.19}$$

Quadrotor Making use of the trapezoidal rule, after choosing $s_0 = h_s$ for ease of computation, it is possible to approximate the integral in (3.12) as:

$$\int_0^{h_s} (b_c \|\boldsymbol{r_t}(s,t)\| \boldsymbol{r_t}(s,t) + \rho_c A \boldsymbol{r_{tt}}(s,t))\, ds \approx \frac{h_s}{2} [b_c \left(\|\boldsymbol{r_t^0}(t)\| \boldsymbol{r_t^0}(t) + \|\boldsymbol{r_t^1}(t)\| \boldsymbol{r_t^1}(t) \right) + \\ + \rho_c A (\boldsymbol{r_{tt}^0}(t) + \boldsymbol{r_{tt}^1}(t))],$$

implying that the boundary condition (3.11), after some rearrangements, can be rewritten as:

$$\left(m_b + \frac{\rho_c A h_s}{2} \right) \boldsymbol{r_{tt}^0}(t) = \boldsymbol{R_b}(t) \sum_{j=1}^{4} \boldsymbol{f_j}(t) - (m_b + \rho_c A h_s) g_0 \boldsymbol{E_z} + \\ - \frac{h_s}{2} [b_c \left(\|\boldsymbol{r_t^0}(t)\| \boldsymbol{r_t^0}(t) + \|\boldsymbol{r_t^1}(t)\| \boldsymbol{r_t^1}(t) \right) + \rho_c A \boldsymbol{r_{tt}^1}(t)] + \boldsymbol{n^1}(t). \tag{3.20}$$

Tip/payload When the payload is not attached, the employment of the central differencing scheme requires that an additional *ghost* node $N+1$ is created to impose the null strain boundary condition (3.13), satisfying:

$$\boldsymbol{r_s^N}(t) = \frac{\boldsymbol{r^{N+1}}(t) - \boldsymbol{r^{N-1}}(t)}{2h_s} = \frac{\boldsymbol{r^N}(t) - \boldsymbol{r^{N-1}}(t)}{\|\boldsymbol{r^N}(t) - \boldsymbol{r^{N-1}}(t)\|} \iff \\ \iff \boldsymbol{r^{N+1}}(t) = \boldsymbol{r^{N-1}}(t) + 2h_s \frac{\boldsymbol{r^N}(t) - \boldsymbol{r^{N-1}}(t)}{\|\boldsymbol{r^N}(t) - \boldsymbol{r^{N-1}}(t)\|}. \tag{3.21}$$

The latter plays a role in the extension of the cable dynamics (3.19) at the free end, i.e. for $i = N$.

The same procedure seen for the quadrotor applies to the slung payload state as well, selecting $s_0 = (N-1)h_s$ while discretizing (3.15), yielding:

$$\left(m_p + \frac{\rho_c A h_s}{2} \right) \boldsymbol{r_{tt}^N}(t) = -\boldsymbol{n^{N-1}}(t) - (m_p + \rho_c A h_s) g_0 \boldsymbol{E_z} + \\ - \frac{h_s}{2} (b_c \|\boldsymbol{r_t^{N-1}}(t)\| \boldsymbol{r_t^{N-1}}(t) + \rho_c A \boldsymbol{r_{tt}^{N-1}}) - \left(\frac{h_s b_c}{2} + b_p \right) \|\boldsymbol{r_t^N}(t)\| \boldsymbol{r_t^N}(t). \tag{3.22}$$

Cards on the table, the workarounds (3.12) and (3.15) have been introduced in order to calculate the derivatives $\boldsymbol{r_s}(s,t)$ consistently using the central differencing scheme (only applicable at points s internal to the domain), and, perhaps more importantly, since this way it becomes possible to rearrange the terms to explicitly consider the coupled dynamics of the UAV/payload and the portion of the cable at the boundary.

Hybrid automaton Once each discrete state of the infinite-dimensional system has been approximated by an equivalent ODE system, the PDHA degenerates into a classical hybrid automaton (Fig. 3.4). The finite-dimensional discrete states are:

- $q = 0$, with the dynamics:

$$\begin{cases} \dot{\boldsymbol{\omega}}_{\boldsymbol{b}}(t) = \boldsymbol{J_b}^{-1}\left(\sum_{j=1}^{4}\boldsymbol{\tau_j}(t) - \boldsymbol{\omega_b}(t)\times \boldsymbol{J_b}\boldsymbol{\omega_b}(t)\right) \\ \boldsymbol{r_{tt}^0}(t) = \frac{1}{m_b + \frac{\rho_c A h_s}{2}}\Big\{\boldsymbol{R_b}(t)\sum_{j=1}^{4}\boldsymbol{f_j}(t) - (m_b+\rho_c A h_s)g_0\boldsymbol{E_z} + \\ \qquad -\frac{h_s}{2}\left[b_c\left(\|\boldsymbol{r_t^0}(t)\|\boldsymbol{r_t^0}(t) + \|\boldsymbol{r_t^1}(t)\|\boldsymbol{r_t^1}(t)\right) + \rho_c A\boldsymbol{r_{tt}^1}(t)\right] + \boldsymbol{n^1}(t)\Big\} \\ \boldsymbol{r_{tt}^i}(t) = -g_0\boldsymbol{E_z} + \frac{1}{\rho_c A}\left(-b_c\|\boldsymbol{r_t^i}(t)\|\boldsymbol{r_t^i}(t) + \boldsymbol{n_s^i}(t)\right), \quad \forall i = 1,\dots,N \\ \boldsymbol{r^{N+1}}(t) = \boldsymbol{r^{N-1}}(t) + 2h_s\frac{\boldsymbol{r^N}(t) - \boldsymbol{r^{N-1}}(t)}{\|\boldsymbol{r^N}(t) - \boldsymbol{r^{N-1}}(t)\|} \end{cases} ; \tag{3.23}$$

- $q = 1$, described by:

$$\begin{cases} \dot{\boldsymbol{\omega}}_{\boldsymbol{b}}(t) = \boldsymbol{J_b}^{-1}\left(\sum_{j=1}^{4}\boldsymbol{\tau_j}(t) - \boldsymbol{\omega_b}(t)\times \boldsymbol{J_b}\boldsymbol{\omega_b}(t)\right) \\ \boldsymbol{r_{tt}^0}(t) = \frac{1}{m_b + \frac{\rho_c A h_s}{2}}\Big\{\boldsymbol{R_b}(t)\sum_{j=1}^{4}\boldsymbol{f_j}(t) - (m_b+\rho_c A h_s)g_0\boldsymbol{E_z} + \\ \qquad -\frac{h_s}{2}\left[b_c\left(\|\boldsymbol{r_t^0}(t)\|\boldsymbol{r_t^0}(t) + \|\boldsymbol{r_t^1}(t)\|\boldsymbol{r_t^1}(t)\right) + \rho_c A\boldsymbol{r_{tt}^1}(t)\right] + \boldsymbol{n^1}(t)\Big\} \\ \boldsymbol{r_{tt}^i}(t) = -g_0\boldsymbol{E_z} + \frac{1}{\rho_c A}\left(-b_c\|\boldsymbol{r_t^i}(t)\|\boldsymbol{r_t^i}(t) + \boldsymbol{n_s^i}(t)\right), \quad \forall i = 1,\dots,N-1 \\ \boldsymbol{r_{tt}^N}(t) = \frac{1}{m_p + \frac{\rho_c A h_s}{2}}\Big\{-\boldsymbol{n^{N-1}}(t) - (m_p+\rho_c A h_s)g_0\boldsymbol{E_z} + \\ \qquad -\frac{h_s}{2}\left(b_c\|\boldsymbol{r_t^{N-1}}(t)\|\boldsymbol{r_t^{N-1}}(t) + \rho_c A\boldsymbol{r_{tt}^{N-1}}\right) - \left(\frac{h_s b_c}{2} + b_p\right)\|\boldsymbol{r_t^N}(t)\|\boldsymbol{r_t^N}(t)\Big\} \end{cases} \tag{3.24}$$

The previous expressions come from (3.19), (3.20), (3.21) and (3.22).

Just like in the previous case, the transitions are dictated by the following events:

- **Attachment**. The attachment of the payload happens when $\boldsymbol{r^N}(t) \in \Lambda^{(0,1)}$, resetting the velocity of the cable tip in order to satisfy the momentum conservation law for the inelastic collision between the payload and the last node of the cable – whose lumped mass is $\frac{\rho_c A h_s}{2}$ –, i.e.:

$$m_p\boldsymbol{v_p}(t^-) + \frac{\rho_c A h_s}{2}\boldsymbol{r_t^N}(t^-) = m_p\boldsymbol{v_p}(t^+) + \frac{\rho_c A h_s}{2}\boldsymbol{r_t^N}(t^+).$$

According the same reasoning and assumptions used for deriving (3.18), one gets[6]:

$$\frac{\rho_c A h_s}{2} \boldsymbol{r_t^N}(t^-) = \left(m_p + \frac{\rho_c A h_s}{2} \right) \boldsymbol{r_t^N}(t^+),$$

and therefore the velocity reset at the cable tip reads:

$$\boldsymbol{r_t^N}(t) \leftarrow \frac{1}{1 + \frac{2m_p}{\rho_c A h_s}} \boldsymbol{r_t^N}(t) \tag{3.25}$$

- **Detachment**. The payload is released as soon as $\boldsymbol{r^N}(t) \in \Lambda^{(1,0)}$.

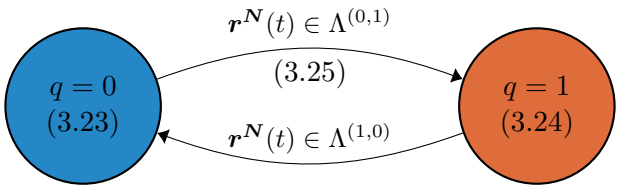


Figure 3.4. State transition diagram of the hybrid automaton that models the UAV-cable-payload system using the finite-dimensional model.

[6]Although the notation differs, (3.25) reduces to (3.18) in the limit $h_s \to 0$; this confirms that the two expressions represent the same event.

Chapter 4

Control and planning architecture

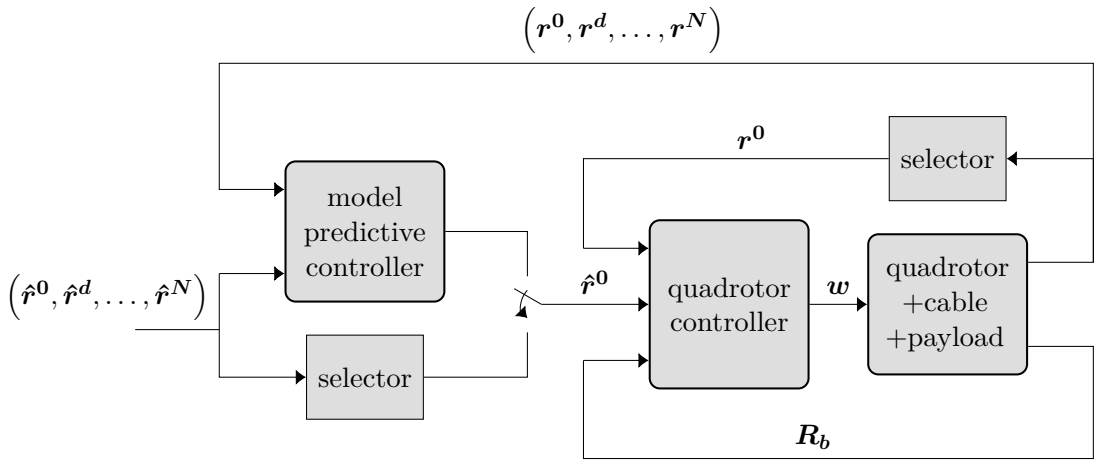


Figure 4.1. Proposed control architecture. As reference input, the quadrotor controller on $SE(3)$ can take either the output of the MPC or an external pre-computed trajectory.

The control hierarchy – illustrated in Fig. 4.1 – is organized as a cascade architecture. A high-level predictive controller generates reference trajectories for the cable shape and the payload position. These references, defined with respect to the first point of the cable, are tracked by the quadrotor through a low-level geometric controller. At any time, the reference input of low-level controller can be switched to an externally computed trajectory to bypass load control.

4.1 Hybrid reduced-order MPC

Internal model It is more important for the MPC that the model be able to capture the main dynamics of the system over the short term, rather than being an extremely detailed representation of every aspect of the process. Therefore, with the advantage of reducing the computational complexity of the control law, this work uses the proposed model-order reduction method (see Sec. 3.1) to build the controller internal model.

The internal model is a ROM-based hybrid automaton (Fig. 4.2), whose discrete-time

successor map, in analogy with (2.3), can be expressed with the notation:

$$(q(k+1), \boldsymbol{z}(k+1)) = \boldsymbol{T}(q(k), \boldsymbol{z}(k), \boldsymbol{v}(k)). \tag{4.1}$$

The latter uses, at step k, the Boolean variable $q(k) \in \{0, 1\}$ to express, respectively, whether the cable is free or the payload is attached to it, $\boldsymbol{z}(k) \in \mathbb{R}^{6(R+2)}$ as continuous state variable, and $\boldsymbol{v}(k) \in \mathbb{R}^3$ as input.
Importantly, to ensure optimal performance in both states of the system, two reduced bases are constructed. They correspond to the projection matrices $\boldsymbol{\Phi^0}$ and $\boldsymbol{\Phi^1}$ – whose derivation is detailed in (3.7). This means that the reset map of the automaton should also include a basis change according to (3.8), i.e.:

$$\boldsymbol{\rho}^e(\boldsymbol{z}(k)) = \begin{bmatrix} \boldsymbol{\Psi}_p^e & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Psi}_v^e \end{bmatrix} \boldsymbol{z}(k), \quad \forall e \in \{(0,1),(1,0)\}, \tag{4.2}$$

where, with $\boldsymbol{\Phi^{q+}}$ being the pseudo-inverse of the projection matrices $\boldsymbol{\Phi^q}, \forall q = 0, 1$, one has:

$$\boldsymbol{\Psi}_p^{(0,1)} := \begin{bmatrix} \boldsymbol{I_3} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Phi^{1+}\Phi^0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I_3} \end{bmatrix}, \quad \boldsymbol{\Psi}_v^{(0,1)} := \begin{bmatrix} \boldsymbol{I_{3(R+1)}} & \boldsymbol{0} \\ \boldsymbol{0} & \frac{\boldsymbol{I_3}}{1+\frac{2m_p}{\rho_c A h_d}} \end{bmatrix} \boldsymbol{\Psi}_p^{(0,1)},$$

$$\boldsymbol{\Psi}_p^{(1,0)} := \begin{bmatrix} \boldsymbol{I_3} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Phi^{0+}\Phi^1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I_3} \end{bmatrix}, \quad \boldsymbol{\Psi}_v^{(1,0)} := \boldsymbol{\Psi}_p^{(1,0)}.$$

In particular, the tip velocity rescaling in $\boldsymbol{\Psi}_v^{(0,1)}$ follows from the conservation of linear momentum in the inelastic attachment event, as discussed in Sec. 3.4.

At time t, based on the current state of the real system, one can assign the initial values $q(0)$ and $\boldsymbol{z}(0)$, with the latter being:

$$\begin{aligned} \boldsymbol{z}^\top(0) &= \begin{bmatrix} \boldsymbol{z_1}^\top(0) & \boldsymbol{z_2}^\top(0) & \dots & \boldsymbol{z_{2R+3}}^\top(0) & \boldsymbol{z_{2R+4}}^\top(0) \end{bmatrix}^\top \\ &\leftarrow \begin{bmatrix} \boldsymbol{r^{0\top}}(t) & \boldsymbol{a_1}^\top(t) & \dots & \boldsymbol{a_R}^\top(t) & \boldsymbol{r^{N\top}}(t) & \boldsymbol{r_t^{0\top}}(t) & \boldsymbol{\dot{a}_1}^\top(t) & \dots & \boldsymbol{\dot{a}_R}^\top(t) & \boldsymbol{r_t^{N\top}}(t) \end{bmatrix}^\top \end{aligned} \tag{4.3}$$

with $\boldsymbol{z_j}(0) \in \mathbb{R}^3, \forall j = 1, \dots, 2R+4$, and with $\boldsymbol{a}_m(t)$ and $\boldsymbol{\dot{a}}_m(t), \forall m = 1, \dots, R$ coming from the projection equations (3.6). Be aware that the previous assignment depends on the specific basis used in the projection, hence on $q(0)$.
The successor map of this hybrid ROM is integrated – employing the fourth-order Runge-Kutta method (RK4) [3] with step time $h_k \in \mathbb{R}$ – by reiterating over k the following algorithm[1].

1. Lift $\boldsymbol{z}(k)$ back to the full FDM coordinate space via the POD equations (3.5) depending on $q(k)$, obtaining the corresponding positions and velocities of the cable nodes, denoted here with underscores:

$$\boldsymbol{z}(k) \xrightarrow{(3.5)} \left(\underline{\boldsymbol{r}}^{\boldsymbol{i}}(k), \underline{\boldsymbol{r}}_{\boldsymbol{t}}^{\boldsymbol{i}}(k)\right), \quad \forall i = 0, d, \dots, N-d, N.$$

[1] In this context, the adaptation of the notation regarding the system dynamics is not explicit, but it is intuitive when considering (4.3).

2. Compute the accelerations of the nodes $\underline{\boldsymbol{r}}_{\boldsymbol{tt}}^{\boldsymbol{i}}(k), \forall i = d, \ldots, N-d, N$ according to the dynamics described in the FDM model in Sec. 3.4. The quadrotor acceleration $\underline{\boldsymbol{r}}_{\boldsymbol{tt}}^{\boldsymbol{0}}(k)$ is the input $\boldsymbol{v}(k)$.

3. Project the accelerations back onto the reduced-order finite-dimensional space using the projection rules (3.6). With analogous notation:

$$\left(\underline{\boldsymbol{r}}_{\boldsymbol{tt}}^{\boldsymbol{0}}(k), \underline{\boldsymbol{r}}_{\boldsymbol{tt}}^{\boldsymbol{d}}(k), \ldots, \underline{\boldsymbol{r}}_{\boldsymbol{tt}}^{\boldsymbol{N-d}}(k), \underline{\boldsymbol{r}}_{\boldsymbol{tt}}^{\boldsymbol{N}}(k)\right) \xrightarrow{(3.6)} \left(\underline{\boldsymbol{r}}_{\boldsymbol{tt}}^{\boldsymbol{0}}(k), \underline{\ddot{\boldsymbol{a}}}_{\boldsymbol{1}}(k), \ldots, \underline{\ddot{\boldsymbol{a}}}_{\boldsymbol{R}}(k), \underline{\boldsymbol{r}}_{\boldsymbol{tt}}^{\boldsymbol{N}}(k)\right).$$

4. Update the positions by integrating the velocities:

$$\begin{aligned}
(\boldsymbol{z}_{\boldsymbol{1}}(k), \boldsymbol{z}_{\boldsymbol{R+3}}(k)) &\xrightarrow{RK4} \boldsymbol{z}_{\boldsymbol{1}}(k+1) \\
(\boldsymbol{z}_{\boldsymbol{2}}(k), \boldsymbol{z}_{\boldsymbol{R+4}}(k)) &\xrightarrow{RK4} \boldsymbol{z}_{\boldsymbol{2}}(k+1) \\
&\vdots \\
(\boldsymbol{z}_{\boldsymbol{R+2}}(k), \boldsymbol{z}_{\boldsymbol{2R+4}}(k)) &\xrightarrow{RK4} \boldsymbol{z}_{\boldsymbol{R+2}}(k+1)
\end{aligned}.$$

5. Update the velocities by integrating the accelerations:

$$\begin{aligned}
\left(\boldsymbol{z}_{\boldsymbol{R+3}}(k), \underline{\boldsymbol{r}}_{\boldsymbol{tt}}^{\boldsymbol{0}}(k)\right) &\xrightarrow{RK4} \boldsymbol{z}_{\boldsymbol{R+3}}(k+1) \\
(\boldsymbol{z}_{\boldsymbol{R+4}}(k), \underline{\ddot{\boldsymbol{a}}}_{\boldsymbol{1}}(k)) &\xrightarrow{RK4} \boldsymbol{z}_{\boldsymbol{R+4}}(k+1) \\
&\vdots \\
\left(\boldsymbol{z}_{\boldsymbol{2R+4}}(k), \underline{\boldsymbol{r}}_{\boldsymbol{tt}}^{\boldsymbol{N}}(k)\right) &\xrightarrow{RK4} \boldsymbol{z}_{\boldsymbol{2R+4}}(k+1)
\end{aligned}.$$

6. Put together the next states:

$$\begin{aligned}
q(k+1) &= q(k), \\
\boldsymbol{z}^{\top}(k+1) &= \left[\boldsymbol{z}_{\boldsymbol{1}}^{\top}(k+1) \quad \boldsymbol{z}_{\boldsymbol{2}}^{\top}(k+1) \quad \ldots \quad \boldsymbol{z}_{\boldsymbol{2R+3}}^{\top}(k+1) \quad \boldsymbol{z}_{\boldsymbol{2R+4}}^{\top}(k+1)\right]^{\top}.
\end{aligned}$$

7. If $\boldsymbol{z}(k+1)$ satisfies a guard condition that triggers a hybrid transition – meaning that the position of the last node of the cable belongs to the guard set of an edge e –, first update $q(k+1)$ accordingly and then apply the reset map (4.2), i.e.:

$$\boldsymbol{z}(k+1) \leftarrow \begin{bmatrix} \boldsymbol{\Psi}_{\boldsymbol{p}}^{\boldsymbol{e}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Psi}_{\boldsymbol{v}}^{\boldsymbol{e}} \end{bmatrix} \boldsymbol{z}(k+1).$$

The last point implies that even though the transition guard condition may be met at any instant within integration interval of the internal model, it is, for the sake of simplicity, always evaluated at the end of it [17]. This simplification introduces an error that decreases as the step size shrinks.

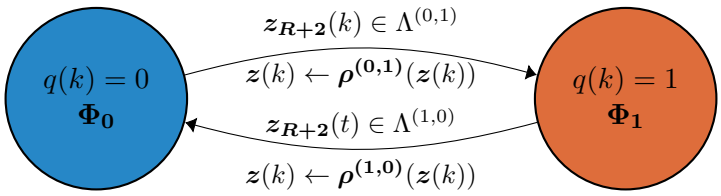


Figure 4.2. State transition diagram of the hybrid automaton that models the reduced-order internal model of the MPC. Here, instead of the flow map, each state shows the projection matrix of the associated basis.

Receding-horizon optimization The MPC is constructed from (2.4), and the optimal control problem uses (4.3) to set the initial conditions at $k = 0$ and (4.1) for the dynamic constraints. The result is the following optimization problem, solved at every call of the MPC:

$$\min_{\{\boldsymbol{v}(k)\}_{k=0}^{H-1}} \left(\sum_{k=0}^{H-1} \ell^{q(k),\hat{q}(k)} \left(\boldsymbol{z}(k), \hat{\boldsymbol{z}}(k), \boldsymbol{v}(k), \hat{\boldsymbol{v}}(k)\right) + \ell_H^{q(H),\hat{q}(H)} \left(\boldsymbol{z}(H), \hat{\boldsymbol{z}}(H)\right) \right) \quad \text{s.t.} \quad \begin{cases} (4.3) \\ (4.1), \quad k = 0, \dots, H-1 \end{cases} . \tag{4.4}$$

With respect to Sec. 2.1, the running and terminal cost functions also include the reference state and control input trajectories, respectively $\hat{\boldsymbol{z}}(k), \forall k = 0, \dots, H$ and $\hat{\boldsymbol{v}}(k), \forall k = 0, \dots, H-1$. Their explicit expressions satisfy:

$$\ell^{q,\hat{q}}(\boldsymbol{z}, \hat{\boldsymbol{z}}, \boldsymbol{v}, \hat{\boldsymbol{v}}) := \left(\boldsymbol{P^q z} - \boldsymbol{P^{\hat{q}}} \hat{\boldsymbol{z}}\right)^\top \boldsymbol{S} \left(\boldsymbol{P^q z} - \boldsymbol{P^{\hat{q}}} \hat{\boldsymbol{z}}\right) + (\boldsymbol{v} - \hat{\boldsymbol{v}})^\top \boldsymbol{W} (\boldsymbol{v} - \hat{\boldsymbol{v}}) + \bar{\ell}^q(\boldsymbol{z}),$$
$$\ell_H^{q,\hat{q}}(\boldsymbol{z}, \hat{\boldsymbol{z}}) = \left(\boldsymbol{P^q z} - \boldsymbol{P^{\hat{q}}} \hat{\boldsymbol{z}}\right)^\top \boldsymbol{S_H} \left(\boldsymbol{P^q z} - \boldsymbol{P^{\hat{q}}} \hat{\boldsymbol{z}}\right) + \bar{\ell}^q(\boldsymbol{z}).$$

The symmetric matrices $\boldsymbol{S}, \boldsymbol{S_H} \geq 0$ and $\boldsymbol{W} > 0$ are weights chosen to penalize the state and control tracking errors, while $\boldsymbol{P^q}$ and $\boldsymbol{P^{\hat{q}}} \in \mathbb{R}^{6(N+1)\times 6(R+2)}$ are projection matrices in charge of mapping the states of the reduced model back into the original state space.

These come from the following:

$$\begin{bmatrix} \boldsymbol{r^0}(t) \\ \boldsymbol{r^d}(t) \\ \vdots \\ \boldsymbol{r^{N-d}}(t) \\ \boldsymbol{r^N}(t) \\ \boldsymbol{r_t^0}(t) \\ \boldsymbol{r_t^d}(t) \\ \vdots \\ \boldsymbol{r_t^{N-d}}(t) \\ \boldsymbol{r_t^N}(t) \end{bmatrix} \approx \boldsymbol{P^q} \begin{bmatrix} \boldsymbol{r^0}(t) \\ \boldsymbol{a_1}(t) \\ \vdots \\ \boldsymbol{a_R}(t) \\ \boldsymbol{r^N}(t) \\ \boldsymbol{r_t^0}(t) \\ \dot{\boldsymbol{a}}_{\boldsymbol{1}}(t) - \dot{\hat{\boldsymbol{a}}}_{\boldsymbol{1}}(t) \\ \vdots \\ \dot{\boldsymbol{a}}_{\boldsymbol{R}}(t) - \dot{\hat{\boldsymbol{a}}}_{\boldsymbol{R}}(t) \\ \boldsymbol{r_t^N}(t) \end{bmatrix},$$

$$\boldsymbol{P^q} \coloneqq \boldsymbol{I_2} \otimes \left[\begin{array}{c:c:c} \boldsymbol{I_3} & & \boldsymbol{0} \\ (1-\frac{h_d}{L})\boldsymbol{I}_3 & & \frac{h_d}{L}\boldsymbol{I_3} \\ \vdots & \boldsymbol{\Phi^q} & \vdots \\ (1-\frac{(M-1)h_d}{L})\boldsymbol{I}_3 & & \frac{(M-1)h_d}{L}\boldsymbol{I_3} \\ \boldsymbol{0} & & \boldsymbol{I_3} \end{array}\right],$$

and they yield a cost metric in reduced coordinates that reproduces the distributed node-wise errors in the Euclidean space. By this choice, any discontinuities in the cost through discrete-state jumps are minimized, since the cost itself becomes q-independent. The costs also feature a barrier term, $\bar{\ell}^q(\boldsymbol{z})$, that penalises violations of possible state constraints. A logarithmic barrier augmented with a quadratic tail is adopted [34]; unlike a pure logarithmic barrier, the extension preserves the existence of a gradient for infeasible trajectories, thereby steering them back towards the admissible set. The formula is:

$$\bar{\ell}^q(\boldsymbol{z}) \coloneqq \begin{cases} -\kappa \ln\left(-\gamma^q(\boldsymbol{z})\right) & \text{if } -\gamma^q(\boldsymbol{z}) \leq \delta \\ \kappa\left\{\frac{1}{2}\left[\left(\frac{-\gamma^q(z)-2\delta}{\delta}\right)^2 - 1\right] - \ln(\delta)\right\} & \text{otherwise} \end{cases}. \tag{4.5}$$

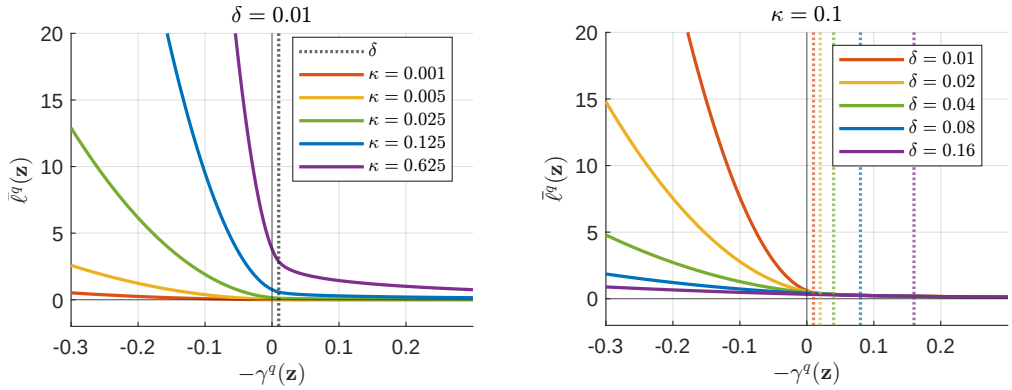
Importantly, $\gamma^q(\boldsymbol{z})$ is the inequality constraint function, i.e. such that:

$$\gamma^q(\boldsymbol{z}) \geq 0, \quad \forall \boldsymbol{z} \in \Gamma^q,$$

with Γ^q being the set of infeasible continuous states at q.
The parameter $\kappa > 0$ controls the severity of the logarithm: larger values amplify the penalty throughout the feasible region, making the constraint influence the optimization even far from the boundary. As κ decreases, the barrier becomes more localized and abrupt, with negligible effect in the interior but steep growth near the boundary – but possibly bringing numerical conditioning problems; $\delta > 0$ can mitigate this as it plays the role of the threshold of the quadratic extension. The effects of the choice of κ and δ are shown in Fig. 4.3.

In accordance with what was said about the internal model, the resulting control input, namely the desired quadrotor acceration, at time t should be[2] $\hat{\boldsymbol{r}}_{\boldsymbol{tt}}^{\boldsymbol{0}}(t) = \boldsymbol{v}^*(0)$, where

[2]The next section brings in a variation to this equation to ensure proper downsampling between the two controllers.



(a) Keeping the threshold δ fixed at 0.01 and varying the barrier severity κ

(b) Fixing the severity $\kappa = 0.1$ and sweeping through threshold values δ.

Figure 4.3. Effect of the parameters κ and δ of the barrier function (4.5). The dotted lines represent the threshold δ – on their left the quadratic extension, on their right the logarithm.

the superscript * denotes a (possibly optimal) solution to the minimization problem. It appears in the expression for the control force of the geometric quadrotor controller via (2.14), after setting the feedback gain matrices $\boldsymbol{K_p}$ and $\boldsymbol{K_v}$ to zero for the proportional and derivative contributions to be cancelled, leaving only the feedforward term.

Owing to its computational efficiency – relevant for prospective real-time implementation – and its capability to manage the hybrid transitions of the system, this work adopts hybrid iLQR (details in Sec. A.1) as the solver for the optimal control problem (4.4). The solver needs to know the reference state trajectories $\hat{\boldsymbol{q}}(k), \forall k = 0, \dots, H$ and $\hat{\boldsymbol{z}}(k), \forall k = 0, \dots, H$, and the reference control sequence $\hat{\boldsymbol{v}}(k), \forall k = 0, \dots, H-1$, all provided by an external offline planner.

4.2 Low-level quadrotor controller

Mass compensation Looking at the dynamics of the quadrotor end of the cable in (3.11), the cable tension acting on the quadrotor has to be balanced out. For this reason, one can set:

$$\begin{cases} m_0 = m_b + \rho_c A L & \text{in } q = 0 \\ m_0 = m_b + \rho_c A L + m_p & \text{in } q = 1 \end{cases}.$$

To justify this, for the sake of simplicity the cable is assumed taut and at rest, hence the equivalent mass associated with the quadrotor is equal to its original mass plus that of the cable and any attached payload.

Interface with the MPC Practical embedded implementations impose different sampling periods on the outer-loop MPC and the inner-loop UAV controller on $SE(3)$. Two complementary reasons justify this disparity:

- **Computational burden**. At every step the MPC solves an optimal control problem whose complexity increases with the prediction horizon, number of states, et cetera. Even with tailored solvers, guaranteeing a deterministic execution time forces the selection of a low-enough update rate.

- **Speed requirement**. The translational closed-loop bandwidth of the quadrotor is bounded by how quickly its attitude – and therefore the thrust vector – can be re-oriented. The inner loop runs at several hundred hertz to reject disturbances (e.g. wind gusts) and to track the references passed down from the MPC. At slower rates, it would compromise both disturbance rejection and the fidelity of the thrust vector with respect to the demand of the outer loop.

In light of the foregoing discussion, the inner $SE(3)$ controller may be regarded as time-continuous from the perspective of the outer MPC provided that its sampling rate is sufficiently high – simply put, it is required that its Nyquist frequency exceeds its bandwidth profusely [11]. From the point of view of the inner-loop controller, though, the input signals are stepwise reference commands whose abrupt discontinuities generate impulsive control transients, potentially exciting high-frequency dynamics that lie outside the nominal model of the system.
The proposed strategy to handle this has two pieces:

- **Interpolation**. At time t, of the control sequence that is returned as the solution to the MPC optimal control problem, not only the first command $\boldsymbol{v}^\star(0)$ is preserved, but also the next $\boldsymbol{v}^\star(1)$. By means of linear interpolation, the reference that reaches the low-level UAV controller is[3]:

$$\hat{\boldsymbol{r}}^{\mathbf{0}}_{\boldsymbol{tt}}(t+t') = \boldsymbol{v}^\star(0) + \frac{t'}{h_k}\left(\boldsymbol{v}^\star(1) - \boldsymbol{v}^\star(0)\right), \quad \forall t' \in [0, h_k].$$

- **Filtering**. Residual discontinuities are still expected at the next call of the MPC, since the predicted control $\boldsymbol{v}^\star(1)$ computed at some time t does not necessarily match $\boldsymbol{v}^\star(0)$ computed at $t + h_k$. This is mitigated through low-pass filtering.

4.3 Segmented optimal trajectory planner

High-level sequential planning This strategy refines a (possibly) infeasible or sub-optimal initial trajectory pair $(\{\hat{\boldsymbol{z}}(k)\}_{k=0}^{\Theta}, \{\hat{\boldsymbol{v}}(k)\}_{k=0}^{\Theta-1})$ by leveraging the same solver used within the MPC module. As illustrated in Fig. 4.4, it uses a divide-and-conquer approach that reduces the complexity of long-horizon trajectory generation by breaking it into smaller, tractable problems. When the original trajectory is significantly infeasible or lies outside the convergence region of the solver, this sequential refinement scheme serves as a robust pre-processing step, reducing the risk of solver failure due to poor initialization. The refined output satisfies the hybrid dynamics constraints and serves as a high-quality reference for the control module.

The planning horizon of length $\Theta \in \mathbb{N}$ is divided into D segments, each of duration Θ_j with $j = 1, \ldots, D$, such that $\sum_{j=1}^{D}(\Theta_j - 1) = \Theta - 1$. Each segment thus overlaps with the next at a single transition step to ensure trajectory continuity.
Given an initial feasible state $\underline{\boldsymbol{z}}(0)$, the following procedure is executed sequentially over the segments:

[3]This assumes that the time the MPC takes to solve the optimal control problem is negligible. If not, the interval of validity of the linear interpolation should be adjusted accordingly, such that the latest available solution is always used.

1. Solve the optimal control problem over the j-th segment:

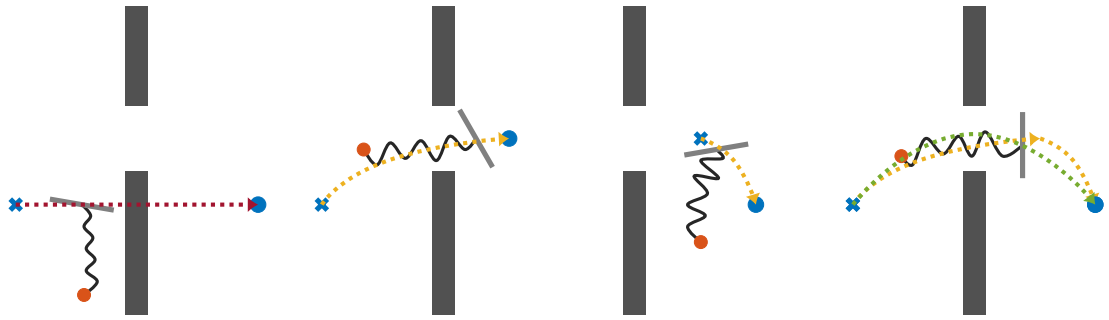
$$\min_{\{\underline{\boldsymbol{v}}(k)\}_{k=0}^{\Theta_j-1}} \left(\sum_{k=0}^{\Theta_j-1} \ell_j^{q(k)} \left(\underline{\boldsymbol{z}}(k), \hat{\underline{\boldsymbol{z}}}(k), \underline{\boldsymbol{v}}(k), \hat{\underline{\boldsymbol{v}}}(k)\right) + \ell_{H,j}^{q(H)} \left(\underline{\boldsymbol{z}}(H), \hat{\underline{\boldsymbol{z}}}(H)\right) \right)$$
$$\text{s.t.} \quad \left(\underline{q}(k+1), \underline{\boldsymbol{z}}(k+1)\right) = \boldsymbol{T}\left(\underline{q}(k), \underline{\boldsymbol{z}}(k), \underline{\boldsymbol{v}}(k)\right), \quad k = 0, \ldots, \Theta_j - 1$$

   where $\boldsymbol{T}$ denotes the discrete-time successor map defined in (4.1), and the cost terms $\ell_j^{q(k)}$ and $\ell_{H,j}^{q(H)}, \forall j = 1, \ldots, D$ mirror those used in the MPC problem (4.4) – but they can differ from segment to segment (e.g. to account for different reference trajectories).

2. Simulate the internal model forward starting from $\underline{\boldsymbol{z}}(0)$ using the optimized input sequence to compute the terminal state $\underline{\boldsymbol{z}}(\Theta_j)$.

3. Set $\underline{\boldsymbol{z}}(0) \leftarrow \underline{\boldsymbol{z}}(\Theta_j)$ and proceed to the next segment.

Once all D segments have been processed, the resulting local solutions are concatenated to form a globally feasible trajectory pair $\left(\{\hat{\boldsymbol{z}}(k)\}_{k=0}^{\Theta}, \{\hat{\boldsymbol{v}}(k)\}_{k=0}^{\Theta-1}\right)$. However, since the optimization was performed only locally, the resulting pair is only optimal over the individual segments, not globally. To improve the quality of the result, the full-horizon problem is solved one last time using the stitched solution as a warm start.



(a) A poor/infeasible initial guess hinders the optimization process.

(b) The optimizer finds a feasible path leading to a given waypoint.

(c) The process is repeated from the final state of the previous iteration.

(d) The composed trajectory serves as a warm start for the global optimizer.

Figure 4.4. Fundamental steps of the proposed segmented planner. Initial conditions are marked with blue crosses, target points with blue dots. Maroon dotted lines represent infeasible trajectories, yellow indicates suboptimal paths, and green identifies the final optimal solution.

Log-barrier homotopy Consider the expression for the log-barrier penalty introduced in the cost function of the MPC formulation, as given in (4.5). This term was introduced to enforce state inequality constraints in a smooth and differentiable manner.

To improve numerical conditioning and promote feasibility in non-convex scenarios (e.g. for obstacle avoidance), the barrier is tightened iteratively. To transition from a relaxed to a strict constraint formulation, a homotopy strategy is employed in which both

the barrier weight κ and the extension-threshold parameter δ are progressively decreased. Tthe update rule follows a geometric decay:

$$\kappa \leftarrow \eta_\kappa \kappa, \qquad \delta \leftarrow \eta_\delta \delta,$$

with decay rates $\eta_\kappa, \eta_\delta \in (0, 1)$. In practice, η_κ is chosen closer to unity to avoid premature hardening of the constraints. The stopping criterion for the homotopy is based on the convergence of the inequality constraint violations, i.e. when $\lambda^q(\boldsymbol{z}) < 0$.
This continuation procedure serves two purposes: both mitigates the consequences of infeasibility in early iterations due to poor initial guesses, and provides a smooth convergence path toward the constrained solution, avoiding abrupt changes in the optimization landscape.

The log-barrier homotopy is applied independently at each subproblem in the segmented trajectory planning pipeline, possibly including the final full-horizon refinement – although in the latter case it is often not necessary, as the initial guess is already feasible by construction.

Part III

Numerical validation

Chapter 5

Simulation setup

5.1 Metrics

For ease of reading, the definitions of the metrics that are used throughout the experimental part are given below:

- **Mode relative energy**. In the context of SVD, it is well known that the square of the m-th singular value represents the amount of energy captured by the corresponding orthonormal vector $\boldsymbol{u_m}$, in the sense that the larger the singular value, the more significant $\boldsymbol{u_m}$ is within the data under investigation. One can then rank the relative importance of mode $\boldsymbol{\phi_m}$ according to the following metric:

$$E_m \coloneqq \frac{\sigma_m^2}{\sum_{m=1}^{M-1} \sigma_m^2}, \quad \forall m = 1, \dots, M-1.$$

- **Wall time**. It is the time taken by the computer to execute a section of code, measured in seconds. From the user's point of view, the wall time is not deterministic, hence it must be characterized statistically. More specifically, T_w is used to denote the total wall time of a simulation, and h_w refers to the measured duration of a single MPC iteration.

- **Maximum time step**. It is the largest time step that can be used in the simulation of a model before causing numerical instability.

- **Tracking error**. The measure of the tracking performance of the control system corresponds to the following metrics, respectively in position and velocity:

$$\varepsilon_p(t) \coloneqq \frac{1}{\sqrt{L}} \|\boldsymbol{r}(\cdot, t) - \hat{\boldsymbol{r}}(\cdot, t)\|_{[0,L]}, \quad \varepsilon_v(t) \coloneqq \frac{1}{\sqrt{L}} \|\boldsymbol{r_t}(\cdot, t) - \hat{\boldsymbol{r}}_{\boldsymbol{t}}(\cdot, t)\|_{[0,L]}.$$

  Here, the norm $\|\cdot\|_{[0,L]}$ appoints the L^2 norm over the interval $[0, L]$, while the factor $\frac{1}{\sqrt{L}}$ is used for normalization.
  Extending the errors already defined in the context of the quadrotor controller in Sec. 4.2 to the other nodes of the cable, one may define:

$$\boldsymbol{\varepsilon_p^i}(t) \coloneqq \boldsymbol{r^i}(t) - \hat{\boldsymbol{r}}^{\boldsymbol{i}}(t), \quad \boldsymbol{\varepsilon_v^i}(t) \coloneqq \boldsymbol{r_t^i}(t) - \hat{\boldsymbol{r}}_{\boldsymbol{t}}^{\boldsymbol{i}}(t), \quad \forall i = 1, \dots, N.$$

Hence, the values of the continuous error functions can be approximated using the trapezoidal rule and computed with the available node data:

$$\varepsilon_p(t) \approx \sqrt{\frac{h_d}{2L} \sum_{\substack{i=0 \\ \text{step } d}}^{N} \left(\|\boldsymbol{\varepsilon_p^i}(t)\|^2 + \|\boldsymbol{\varepsilon_p^{i-1}}(t)\|^2 \right)},$$

$$\varepsilon_v(t) \approx \sqrt{\frac{h_d}{2L} \sum_{\substack{i=0 \\ \text{step } d}}^{N} \left(\|\boldsymbol{\varepsilon_v^i}(t)\|^2 + \|\boldsymbol{\varepsilon_v^{i-1}}(t)\|^2 \right)}.$$

- **ROM approximation error**. The performance of the ROM can be assessed relatively to the true behavior of the system by looking at the following error functions, respectively in position and velocity, which are consistent with the tracking errors introduced above:

$$\epsilon_p(t) \coloneqq \frac{1}{\sqrt{L}} \|\boldsymbol{r}(\cdot, t) - \mathfrak{r}(\cdot, t)\|_{[0,L]}, \quad \epsilon_v(t) \coloneqq \frac{1}{\sqrt{L}} \|\boldsymbol{r_t}(\cdot, t) - \mathfrak{r}_{\boldsymbol{t}}(\cdot, t)\|_{[0,L]}.$$

  The Fraktur font is used to denote the outputs of the reduced model.
  Assuming the complete finite-dimensional model as ground truth, after simulating the ROM under its same conditions (including the time step size), the error functions are rewritten in an actually computable form using the trapezoidal rule:

$$\epsilon_p(t) \approx \sqrt{\frac{h_d}{2L} \sum_{\substack{i=d \\ \text{step } d}}^{N} (\|\boldsymbol{r^i}(t) - \mathfrak{r}^{\boldsymbol{i}}(t)\|^2 + \|\boldsymbol{r^{i-1}}(t) - \mathfrak{r}^{\boldsymbol{i-1}}(t)\|^2)},$$

$$\epsilon_v(t) \approx \sqrt{\frac{h_d}{2L} \sum_{\substack{i=d \\ \text{step } d}}^{N} \left(\|\boldsymbol{r_t^i}(t) - \mathfrak{r}_{\boldsymbol{t}}^{\boldsymbol{i}}(t)\|^2 + \|\boldsymbol{r_t^{i-1}}(t) - \mathfrak{r}_{\boldsymbol{t}}^{\boldsymbol{i-1}}(t)\|^2 \right)}.$$

- **Time-lapse visualization**. In the controller evaluation chapter, it is essential to provide the reader with a general qualitative understanding of the behavior of the controlled system. To this end, each experiment is accompanied by a series of figures showing consecutive snapshots of the system state (timestamped above the plot). The snapshots are visually layered with increasing opacity: the oldest is rendered with the highest transparency, while the most recent is fully opaque. In each plot, relatively to the most recent snapshot, the reference trajectory is depicted as a solid green line, and the predicted evolution of the cable tip within the MPC prediction horizon is shown as a dashed line – blue when a free tip is predicted, and orange when a slung payload is expected. Maroon and violet circles denote the guard sets for hybrid transitions, corresponding to attachment and detachment events, respectively. Finally, forbidden regions of space (i.e. obstacles) are enclosed by yellow surfaces.

Occasionally, it may be useful to summarize the time-dependent quantities by computing the root mean square (RMS) over the duration of the simulation, denoted with the

superscript ^{rms}. For instance, ε_p^{rms} and ε_v^{rms} appoint the root mean square of the tracking errors in position and velocity, respectively, while ϵ_p^{rms} and ϵ_v^{rms} denote the same for the ROM approximation errors.

5.2 Configuration[1]

Solver For the MPC, two solver variants are available[2]. The wording *HiLQR* denotes the full version (details in Sec. A.1), incorporating numerical stabilization techniques such as line search and Levenberg-Marquardt regularization. In contrast, *HiLQR-RTI* (*real-time iteration*) is tailored for real-time applications: regularization is disabled, and only a single rollout, backward sweep, and forward sweep are executed per iteration. Where not specified, the standard HiLQR solver is used.

Parameters The following is valid for the entire experimental campaign:

- The system is simulated by integrating the equations of the finite-dimensional hybrid automaton introduced in Sec. 3.4 with $N = 100$ using the Runge-Kutta integration method of order 4 with time step $h_t = 5 \times 10^{-4}$ s (equivalent to a frequency of 2 kHz).
- It is configured according to the physical and geometrical parameters listed in Tab. 5.1.
- The MPC works with time step $h_k = 2.5 \times 10^{-2}$ s and prediction horizon $H = 32$, while the low-level quadrotor controller runs every 5×10^{-3} s (i.e. 40 Hz and 200 Hz frequencies, respectively).

Table 5.1. Physical and geometrical parameters of the aerial system.

Symbol	Value [SI units]	Symbol	Value [SI units]
m_b	0.3	E	1×10^5
$\boldsymbol{J_b}$	$\mathrm{diag}\{1, 1, 2\} \times 10^{-6}$	A	7.85×10^{-5}
c_f	4×10^{-7}	L	1
c_τ	3×10^{-9}	ρ_c	1.27×10^3
$\boldsymbol{p_1}$	$\begin{bmatrix} 0.15 & 0 & 0 \end{bmatrix}^\top$	b_c	1.29×10^{-2}
$\boldsymbol{p_2}$	$\begin{bmatrix} 0 & 0.15 & 0 \end{bmatrix}^\top$	m_p	0.1
$\boldsymbol{p_3}$	$\begin{bmatrix} -0.15 & 0 & 0 \end{bmatrix}^\top$	b_p	1.29×10^{-2}
$\boldsymbol{p_4}$	$\begin{bmatrix} 0 & -0.15 & 0 \end{bmatrix}^\top$	g_0	9.81

[1] All the simulations are carried out in *MATLAB R2024b* on a laptop equipped with an *Intel Core i5-1235U* CPU and 16 GB of RAM.

[2] Both solvers are implemented in CasADi [1] for rapid prototyping and symbolic efficiency, although C code generation has not been performed.

Quadrotor controller gains Below is the expression of the attitude error dynamics of the quadrotor subject to the torque control action (2.18) as adapted from the appendix of [20]:

$$\begin{cases} \dot{\boldsymbol{\varepsilon}}_{\boldsymbol{R}}(t) = \frac{1}{2}\left(\mathrm{tr}(\boldsymbol{R_b}(t)^\top \hat{\boldsymbol{R}}_{\boldsymbol{b}}(t))\boldsymbol{I_3} - \hat{\boldsymbol{R}}_{\boldsymbol{b}}(t)^\top \boldsymbol{R_b}(t)\right)\boldsymbol{\varepsilon}_{\boldsymbol{\omega}}(t) \\ \dot{\boldsymbol{\varepsilon}}_{\boldsymbol{\omega}}(t) = -\boldsymbol{K}_{\boldsymbol{\omega}}\boldsymbol{\varepsilon}_{\boldsymbol{\omega}}(t) - \boldsymbol{K}_{\boldsymbol{R}}\boldsymbol{\varepsilon}_{\boldsymbol{R}}(t) \end{cases},$$

with $\mathrm{tr}(\cdot)$ denoting the trace operator.
Locally, i.e. for small errors around the desired attitude ($\boldsymbol{\varepsilon}_{\boldsymbol{R}}(t) \approx \boldsymbol{0} \iff \boldsymbol{R_b}(t) \approx \hat{\boldsymbol{R}}_{\boldsymbol{b}}(t)$), the latter becomes akin to a second-order linear system in $\boldsymbol{\varepsilon}_{\boldsymbol{R}}(t)$:

$$\ddot{\boldsymbol{\varepsilon}}_{\boldsymbol{R}}(t) \approx -\boldsymbol{K}_{\boldsymbol{\omega}}\dot{\boldsymbol{\varepsilon}}_{\boldsymbol{R}}(t) - \boldsymbol{K}_{\boldsymbol{R}}\boldsymbol{\varepsilon}_{\boldsymbol{R}}(t).$$

According to this rationale, then, it is possible to model the rotational response of the controlled quadrotor by choosing the gains on the basis of the results of linear control theory [6]. Thus, to make the error assume the (desirable) behavior arising from the presence of a pair of complex, conjugate roots with natural frequency $\omega_{n,R} = 10\,\mathrm{rad\,s^{-1}}$ and damping coefficient $\zeta_R = 0.8$, one can set:

$$\boldsymbol{K_R} = \omega_{n,R}^2 \boldsymbol{I_3} = 100\boldsymbol{I_3}, \quad \boldsymbol{K_\omega} = 2\zeta_R \omega_{n,R}\boldsymbol{I_3} = 16\boldsymbol{I_3}.$$

When put in the conditions of rest and tautness assumed in Sec. 4.2, considering instantaneous rotational dynamics and neglecting any constraints, also the translational part can be approximated as a second-order system in $\boldsymbol{\varepsilon}_{\boldsymbol{p}}^{\boldsymbol{0}}(t)$:

$$\ddot{\boldsymbol{\varepsilon}}_{\boldsymbol{p}}^{\boldsymbol{0}}(t) \approx -\boldsymbol{K_v}\dot{\boldsymbol{\varepsilon}}_{\boldsymbol{p}}^{\boldsymbol{0}}(t) - \boldsymbol{K_p}\boldsymbol{\varepsilon}_{\boldsymbol{p}}^{\boldsymbol{0}}(t).$$

Since the effectiveness of the geometric controller is guaranteed if the translatory dynamics is sufficiently slow compared to the rotational dynamics, one can choose $\omega_{n,p} = 3\,\mathrm{rad\,s^{-1}}$ and $\zeta_p = 0.8$, and set:

$$\boldsymbol{K_p} = \omega_{n,p}^2 \boldsymbol{I_3} = 9\boldsymbol{I_3}, \quad \boldsymbol{K_v} = 2\zeta_p \omega_{n,p}\boldsymbol{I_3} = 4.8\boldsymbol{I_3}.$$

5.3 ROM construction

The orthonormal modes used in the ROM are obtained via POD following the steps described in Sec. 3.1, where the tensor $\tilde{\boldsymbol{\mathcal{R}}}$ contains snapshots acquired from the simulation of the system controlled directly via the geometric quadrotor controller – with its gains set to the aforementioned values. In order to derive a computationally lightweight ROM, a decimation factor $d = 10$ is chosen, which corresponds to a spatial sampling step of length $h_d = 0.1\,\mathrm{m}$. This means that $M = 10$, and therefore the selection of $O = 50$ is suitable on the temporal side.

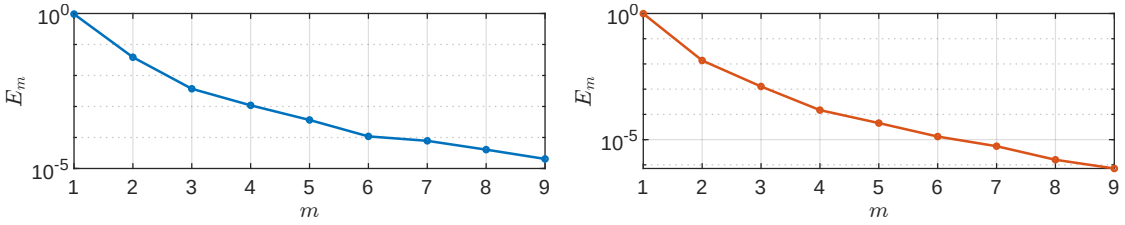
During the simulation interval $[0\,\mathrm{s}, 10\,\mathrm{s}]$, starting from the initial conditions:

$$\boldsymbol{r^i}(0) = -ih_s\boldsymbol{E_z}, \quad \boldsymbol{r_t^i}(0) = \boldsymbol{0}, \quad \forall i = 0, \ldots, N,$$

the quadrotor is required to follow a trajectory capable of exciting the most important modes of the cable. A straightforward option is an isotropic sinusoidal sweep such as the following:

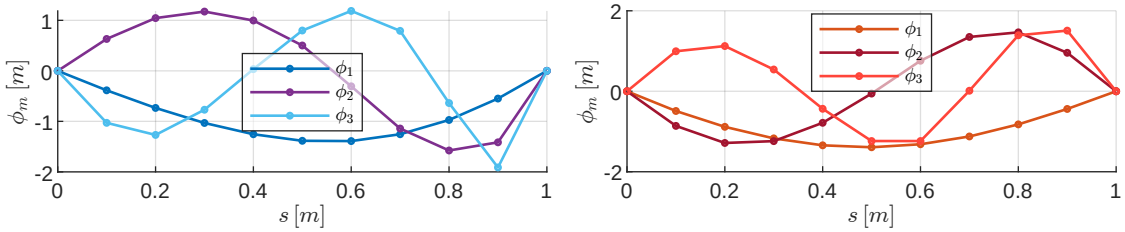
$$\begin{cases} \hat{\boldsymbol{r}}^{\boldsymbol{0}}(t) = 3\,(1 - \cos(2t))\,(\boldsymbol{E_x} + \boldsymbol{E_y} + \boldsymbol{E_z}), \\ \hat{\boldsymbol{r}}_{\boldsymbol{t}}^{\boldsymbol{0}}(t) = 6\sin(2t)(\boldsymbol{E_x} + \boldsymbol{E_y} + \boldsymbol{E_z}), \\ \hat{\boldsymbol{r}}_{\boldsymbol{tt}}^{\boldsymbol{0}}(t) = 12\cos(2t)(\boldsymbol{E_x} + \boldsymbol{E_y} + \boldsymbol{E_z}) \end{cases}.$$

The application of the POD is repeated for both discrete states $q = 0$ and $q = 1$, ultimately yielding two reduced bases. These are evaluated by examining the relative energy contained in each mode and then visually inspecting the spatial structure of the four most energetic modes.
Fig. 5.1 shows that the first two modes $\boldsymbol{\phi_1}$ and $\boldsymbol{\phi_2}$ contain more than 99% of the total energy for both discrete states, with $\boldsymbol{\phi_1}$ constituting more than 95% and 98%, respectively for state $q = 0$ and state $q = 1$. Another notable point is that the modes recall sections of distorted sinusoids (Fig. 5.2), due to the pseudoperiodicity of the evolution of the function $\tilde{\boldsymbol{r}}(\cdot, t)$, and that the distortion is more pronounced in state $q = 0$ and toward the free end of the cable.



(a) Free tip ($q = 0$). **(b)** Slung payload ($q = 1$).

Figure 5.1. Relative energy content of each independent mode found via POD (logarithmic scale).



(a) Free tip ($q = 0$). **(b)** Slung payload ($q = 1$).

Figure 5.2. Spatial representation of the 3 modes with the highest energy content, i.e. the largest singular values.

Chapter 6

ROM evaluation

The results featured within this chapter always refer to the same scenario, characterized by the following choices:

- **Initial conditions**. The initial conditions of the cable have it released horizontally and with null velocity according to:

$$\boldsymbol{r}^{\boldsymbol{i}}(0) = -ih_s\boldsymbol{E_x}, \quad \boldsymbol{r_t^i}(0) = \mathbf{0}, \quad \forall i = 0, \dots, N, \tag{6.1}$$

  and consequently the initial coordinates in the reduced space are found using the projection formulas (3.6).

- **Reference**. The quadrotor is commanded to move toward positive y in a straight line for 1 m under a quintic rest-to-rest timing law over the interval $[0\,\mathrm{s}, 1\,\mathrm{s}]$:

$$\begin{cases} \hat{\boldsymbol{r}}^{\mathbf{0}}(t) = \left(6t^5 - 15t^4 + 10t^3\right)\boldsymbol{E_y}, \\ \hat{\boldsymbol{r}}_{\boldsymbol{t}}^{\mathbf{0}}(t) = \left(30t^4 - 60t^3 + 30t^2\right)\boldsymbol{E_y}, \\ \hat{\boldsymbol{r}}_{\boldsymbol{tt}}^{\mathbf{0}}(t) = \left(120t^3 - 180t^2 + 60t\right)\boldsymbol{E_y} \end{cases} . \tag{6.2}$$

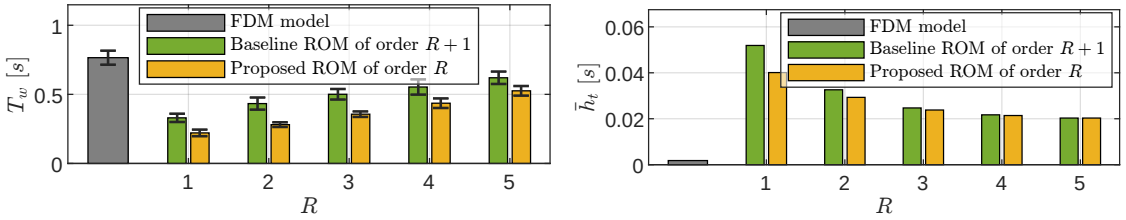
Valuable insight can be gained by carrying out a comparison between the proposed model and the existing approach of [29] – referred to as the *baseline* ROM. For the sake of fairness, since the proposed model does not affect the states associated with the tip of the cable or the payload, whereas in the baseline model these too are projected onto the reduced base, considering one extra mode in the latter provides an even playing field in terms of dimensionality of the reduced state space.

6.1 Performance vs. model order

The goal is to assess computational costs and gains in accuracy as the ROM order varies. The evaluation is performed by comparing five variants of the proposed and of the baseline approach with respect to the complete FDM model. Four distinct aspects are taken into account: the total integration wall time, the maximum time step, and the approximation errors in position and velocity.

From what is shown in Fig. 6.1, it appears that the average wall time required for the integration of the reduced models, which is significantly lower than that required by the

complete model, increases with an almost-linear trend as the order grows. Out of the two reduction approaches, the proposed one is slightly faster, and this is attributable to the need for fewer projection operations for the same state space dimensionality. However, for both cases, the real gain in terms of computation time — especially with an eye toward real-time implementation — is achieved by combining the shorter wall time with the far higher maximum allowable time step supported by the ROMs.

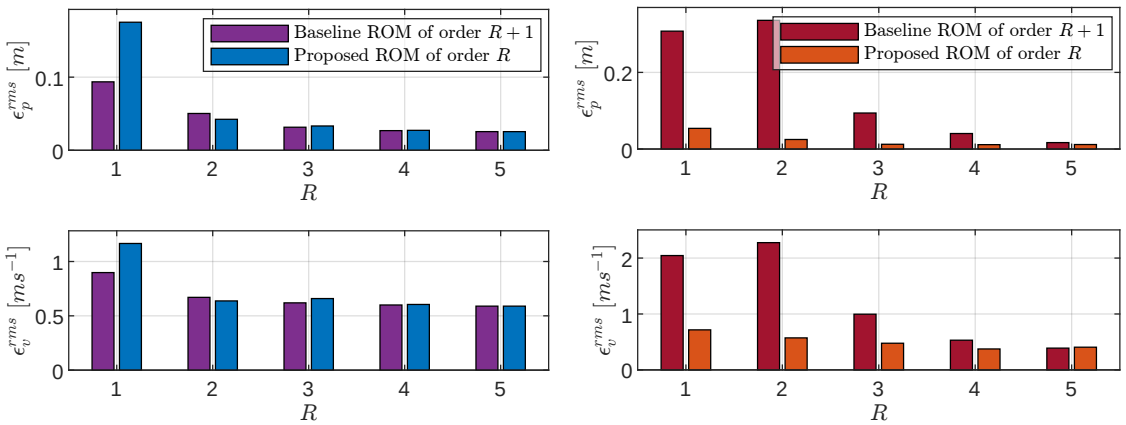


(a) Statistics of the wall time required to integrate the models. Mean and standard deviation are determined from a population of 30 observations per discrete state.

(b) Maximum time step for which the models are stable. The minimum value encountered through trial-and-error among the two discrete states is selected.

Figure 6.1. Time characteristics across model variants – both FDM and ROM.

Meanwhile, in Fig. 6.2 one can easily notice that in the discrete state $q = 0$, the two alternatives produce comparable performance in terms of the RMS of the approximation errors, especially with order at least equal to 2 (3 for the baseline model). This changes dramatically when the payload is attached to the cable tip — corresponding to $q = 1$ —, as the superiority of the proposed method is blatant. Then again, the baseline ROM is not designed for payload transportation.
Focusing on the proposed model, increasing the order beyond the second brings about only limited benefits. In light of these considerations, the second-order proposed ROM is chosen as the reference reduced model for the subsequent experiments in this chapter.



(a) Free tip ($q = 0$).

(b) Slung payload ($q = 1$).

Figure 6.2. Accuracy of the ROMs with respect to the FDM model across different orders.

6.2 Accuracy vs. payload mass

Here, the link between the payload mass and the advantage in accuracy of the proposed reduction method over the baseline is investigated further. To this end, the evolutions of the position and velocity approximation errors of both reduction methods are shown, with the mass of the payload taking the values $m_p^\star \in \{0, 0.01, 0.02, 0.05, 0.1, 0.2\}$kg. Of these, as is obvious, only the null value corresponds to discrete state $q = 0$, while all the other cases are evaluated in $q = 1$. Please bear in mind that the reduced orthonormal bases are recalculated for each $m_p^\star$.

Table 6.1. Effect of the mass of the payload on the ROM accuracy with respect to the FDM model (RMS errors). Errors are computed for the baseline ROM of order 3 and the proposed ROM of order 2.

(a) $m_p = 0\,\mathrm{kg}$

Metric	Baseline	Proposed
$\epsilon_p^{rms}[\mathrm{m}]$	0.046	0.650
$\epsilon_v^{rms}[\mathrm{m\,s^{-1}}]$	0.047	0.660

(b) $m_p = 0.01\,\mathrm{kg}$

Metric	Baseline	Proposed
$\epsilon_p^{rms}[\mathrm{m}]$	0.039	0.601
$\epsilon_v^{rms}[\mathrm{m\,s^{-1}}]$	0.062	0.702

(c) $m_p = 0.02\,\mathrm{kg}$

Metric	Baseline	Proposed
$\epsilon_p^{rms}[\mathrm{m}]$	0.078	0.763
$\epsilon_v^{rms}[\mathrm{m\,s^{-1}}]$	0.039	0.572

(d) $m_p = 0.05\,\mathrm{kg}$

Metric	Baseline	Proposed
$\epsilon_p^{rms}[\mathrm{m}]$	0.179	1.446
$\epsilon_v^{rms}[\mathrm{m\,s^{-1}}]$	0.030	0.553

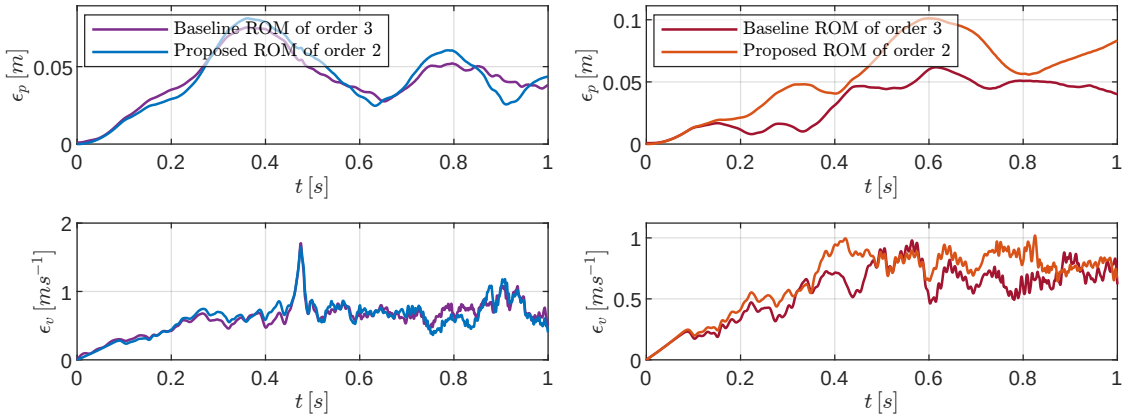
(e) $m_p = 0.1\,\mathrm{kg}$

Metric	Baseline	Proposed
$\epsilon_p^{rms}[\mathrm{m}]$	0.307	2.265
$\epsilon_v^{rms}[\mathrm{m\,s^{-1}}]$	0.023	0.547

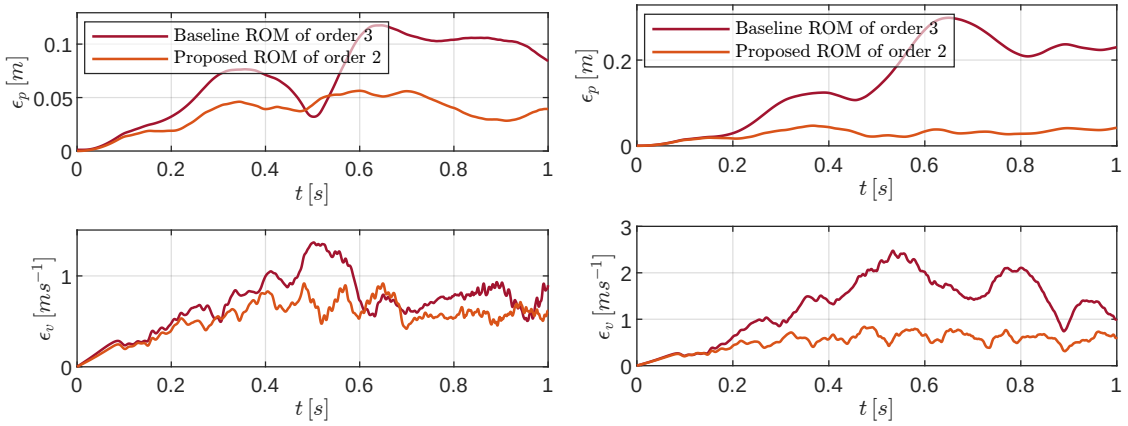
(f) $m_p = 0.2\,\mathrm{kg}$

Metric	Baseline	Proposed
$\epsilon_p^{rms}[\mathrm{m}]$	0.819	8.354
$\epsilon_v^{rms}[\mathrm{m\,s^{-1}}]$	0.038	0.832

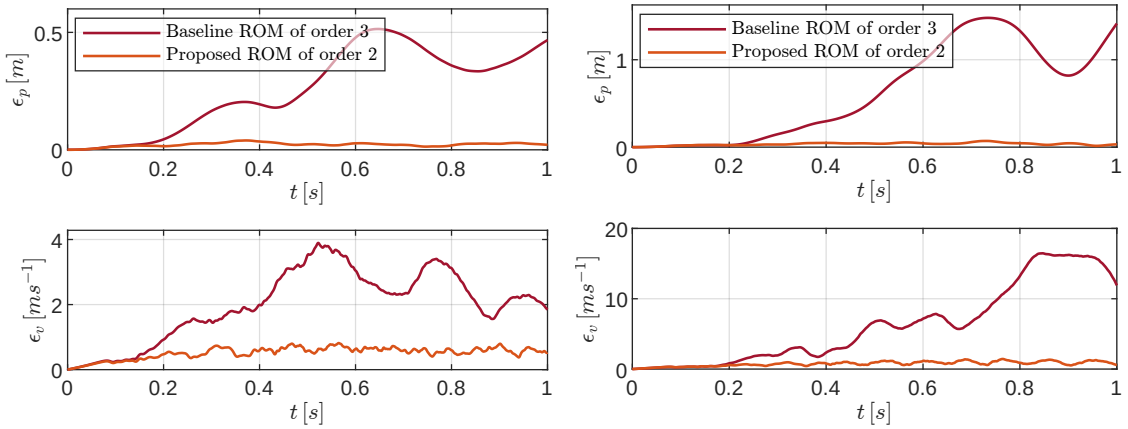
From the results shown in Fig. 6.3 and Tab. 6.1, it is readily seen that, in all the plots, the errors of the two methods follow very similar trends for the earliest instants. Furthermore, although in state $q = 0$ no major differences between the models are appreciated (as anticipated in the previous paragraph), the larger the mass, the better the proposed ROM performs compared to the baseline. However, an exception occurs when the mass is $m_p^\star = 0.1\,\mathrm{kg}$, since it is comparable with the lumped mass of a cable section $\rho_c A h_d$, and the baseline model is able to capture the payload dynamics more effectively. From a practical standpoint, these results suggest that the proposed method could be more suitable for applications such as the hybrid scenario at hand. Conversely, for systems where the payload mass is either negligible or closely matches the cable node mass, the baseline method might still be a viable alternative.



(a) $m_p = 0\,\text{kg}$. **(b)** $m_p = 0.01\,\text{kg}$.



(c) $m_p = 0.02\,\text{kg}$. **(d)** $m_p = 0.05\,\text{kg}$.



(e) $m_p = 0.1\,\text{kg}$. **(f)** $m_p = 0.2\,\text{kg}$.

Figure 6.3. Influence of the payload mass on the accuracy of the ROMs compared to the FDM model.

6.3 Accuracy vs. parametric uncertainty

Ultimately, it is worthwhile to assess the robustness of the ROM with respect to the uncertainty of its parameters. For this, given a selection of key physical parameters, those of the original system are all kept unaltered, while a scaled version of a single one of them at a time is employed in the ROM and the resulting accuracy in terms of approximation errors is measured. As done earlier, each figure details a comparison between baseline and proposed ROM, and plots corresponding to the discrete states $q = 0$ and $q = 1$ are placed side by side.

From the results displayed in Fig. 6.4 and 6.5, let the following be noted.

- **Cable length** $L^\star \in [0.9L, 1.1L]$. The proposed model is significantly more sensitive than the baseline with respect to uncertainty about the cable length. This is not surprising, since the new reduction scheme makes explicit use of L in (3.1). In addition, the plots are asymmetric, and an $L^\star$ shorter than the nominal value harms less than a larger one. Therefore it is crucial that the parameter be as accurate as possible..

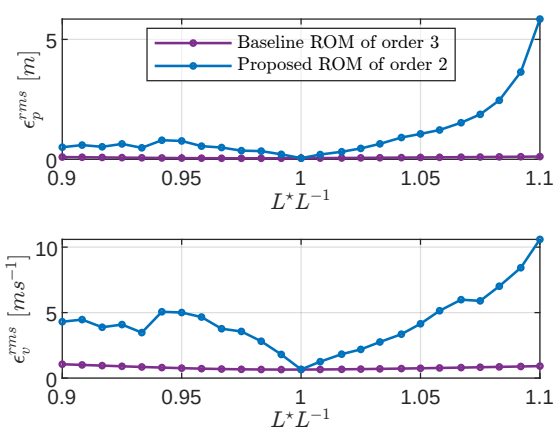
- **Cable linear density** $\rho_c^\star A^\star \in [0.7\rho_c A, 1.3\rho_c A]$. Although in the free tip case the two ROMs are comparable, the proposed one proves its higher accuracy and lower sensitivity to uncertainty when the payload is attached. Given the low reported error values, uncertainty on the linear density is not a major problem for the proposed scheme.

- **Cable axial stiffness** $E^\star A^\star \in [0.7EA, 1.3EA]$. The results outline a similar, but reversed, pattern to the previous case.

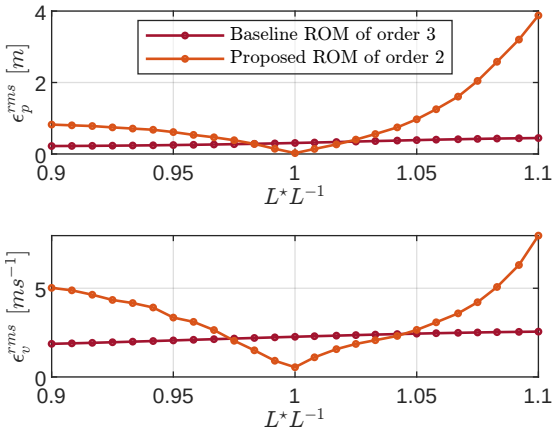
- **Cable drag coefficient** $b_c^\star \in [0, 2b_c]$. Despite the wide range of values tested, both reduced models are shown to be robust to the variation of the cable drag coefficient. This is a positive outcome, as the drag coefficient is a parameter that is difficult to estimate accurately.

- **Payload mass** $m_p^\star \in [0, 2m_p]$: As expected, the baseline model turns out to be noticeably sensitive with respect to uncertainty about the mass of the payload. When the payload is detached, factoring in even a small mass results in a large growth in the approximation error, as the boundary condition at the tip of the cable changes. Conversely, when the payload is attached, the fact that the minimum error is not found in the vicinity of the nominal value m_l outlines the inadequacy of the model to represent this scenario. On the other side, the proposed ROM is significantly more robust in both cases, showing a more continuous error trend even when the tip boundary condition is wrongfully switched.

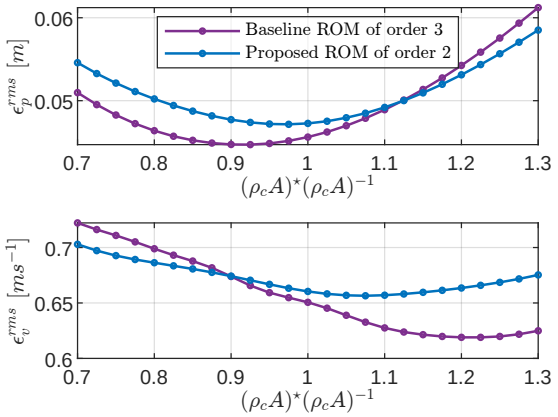
- **Payload drag coefficient** $b_p^\star \in [0, 2b_p]$: This experiment is, of necessity, only performed for the discrete state $q = 1$. The high robustness exhibited for the cable drag coefficient is reconfirmed.



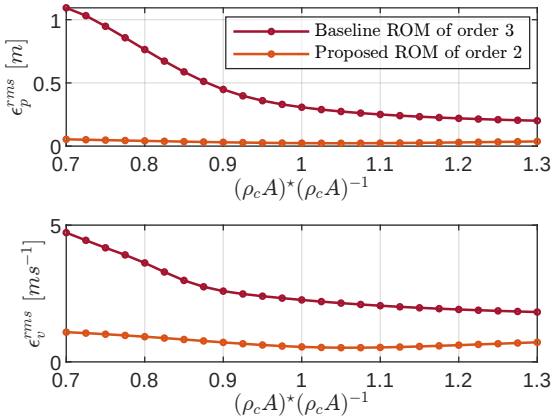
(a) Cable length, free tip ($q = 0$).



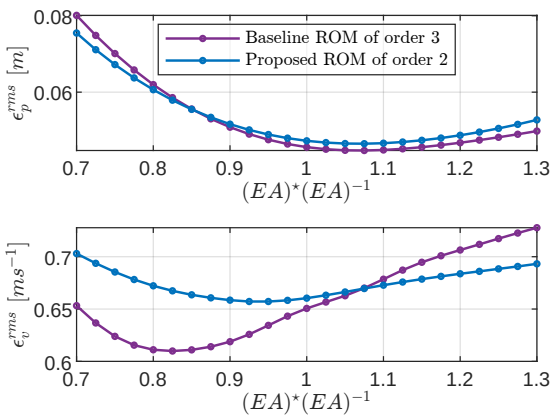
(b) Cable length, slung payload ($q = 1$).



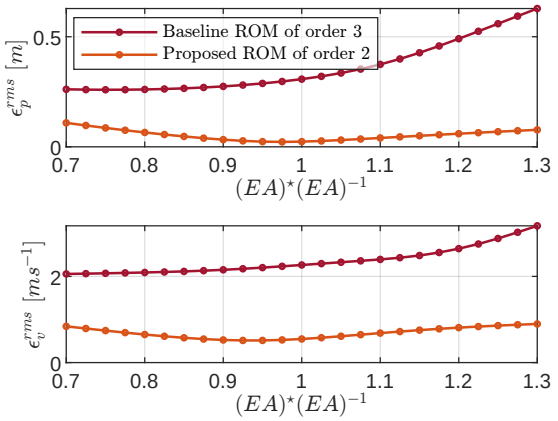
(c) Cable linear density, free tip ($q = 0$).



(d) Cable linear density, slung payload ($q = 1$).

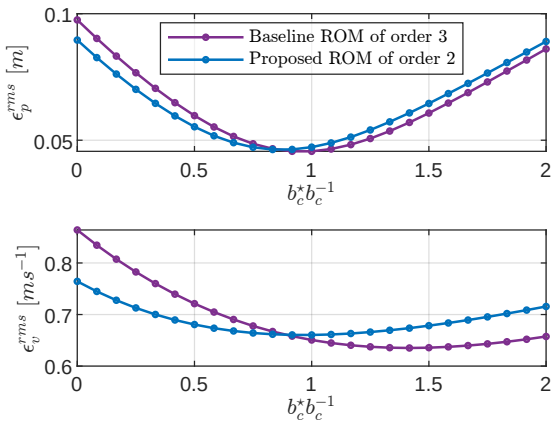


(e) Cable axial stiffness, free tip ($q = 0$).

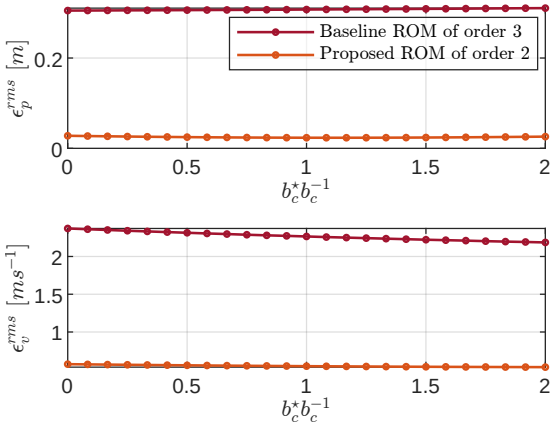


(f) Cable axial stiffness, slung payload ($q = 1$).

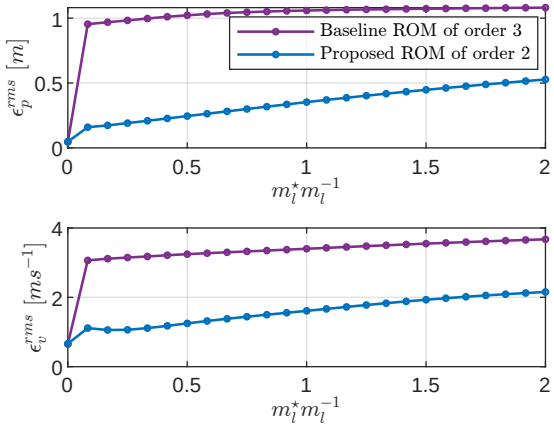
Figure 6.4. ROM robustness to parametric uncertainty, evaluated in terms of accuracy with respect to the FDM model (first part).



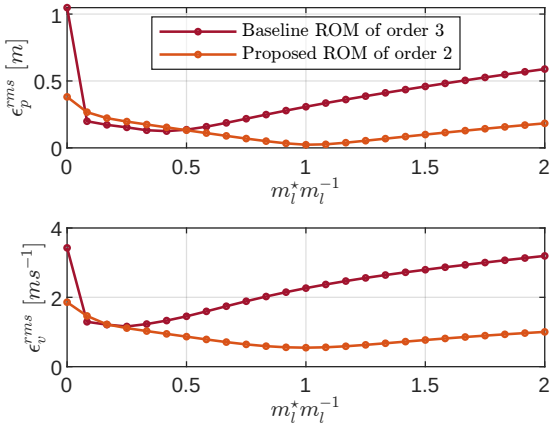
(a) Cable drag coefficient, free tip ($q = 0$).



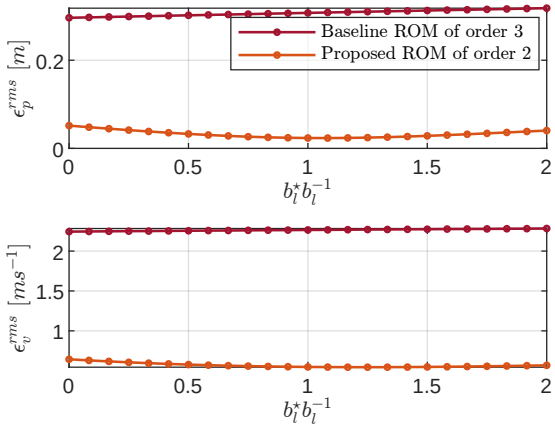
(b) Cable drag coeff., slung payload ($q = 1$).



(c) Payload mass, free tip ($q = 0$).



(d) Payload mass, slung payload ($q = 1$).



(e) Payload drag coefficient.

Figure 6.5. ROM robustness to parametric uncertainty, evaluated in terms of accuracy with respect to the FDM model (second part).

Chapter 7

Controller evaluation

7.1 Setpoint regulation

One of the simplest yet most significant tasks assigned to the control system is setpoint regulation. This comparison highlights the advantage of using the proposed predictive controller, demonstrating how future-state forecasting can improve the transient response of the system – enhancing its ability to damp cable and payload oscillations.
For this experiment, two control strategies are compared: a naive approach using the geometric controller alone, and the full controller with MPC enabled. In the former, the oscillations of the cable and payload are treated as disturbances to be rejected. In the latter, the MPC actively predicts the systems future evolution – based on the proposed ROM – and adjusts the control inputs to mitigate oscillations and accelerate convergence.

Consider the following instructions:

- **Initial conditions**. The aerial system is initially released in a horizontal configuration in accordance with (6.1), intentionally provoking an impulse through the impact of its fall to excite the transient response. The same comparison is repeated both in the absence and in the presence of the payload ($q = 0$ and $q = 1$ respectively), without allowing discrete state transitions.

- **Reference**. The reference setpoint for the control module is the static equilibrium configuration of the cable. This equilibrium – denoted in blackboard font below – is derived by setting the time derivatives in the system equations to zero, adapting the formulation in [29] for both discrete states $q \in \{0, 1\}$:

$$\begin{cases} \mathbb{r}^{\boldsymbol{q}}(s) \coloneqq \mathbb{r}(0) + \left[\left(1 + \frac{g_0(\rho_c AL + qm_p)}{EA}\right) s + g_0 \frac{\rho_c}{2E} s^2\right] \boldsymbol{E_z} \\ \mathbb{r}^{\boldsymbol{q}}_{\boldsymbol{t}}(s) \coloneqq \mathbf{0} \\ \mathbb{r}^{\boldsymbol{q}}_{\boldsymbol{tt}}(s) \coloneqq \mathbf{0} \end{cases}, \quad \forall s \in [0, L]. \tag{7.1}$$

- **MPC configuration**. The weights in the MPC cost function are selected based on empirical tuning to balance tracking performance and control effort. The chosen values values are summarized as follows:

$$\boldsymbol{S} \coloneqq \begin{bmatrix} \boldsymbol{S_p} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{S_v} \end{bmatrix}, \quad \boldsymbol{S_H} \coloneqq 5\boldsymbol{S}, \quad \boldsymbol{W} \coloneqq \boldsymbol{I_3},$$

where:

$$\boldsymbol{S_p} \coloneqq \begin{bmatrix} 5 & \mathbf{0} & 0 \\ \mathbf{0} & 5\boldsymbol{I_{M-1}} & \mathbf{0} \\ 0 & \mathbf{0} & 10 \end{bmatrix} \otimes \boldsymbol{I_3}, \quad \boldsymbol{S_v} \coloneqq \begin{bmatrix} 0.5 & \mathbf{0} & 0 \\ \mathbf{0} & 0.05\boldsymbol{I_{M-1}} & \mathbf{0} \\ 0 & \mathbf{0} & 0.1 \end{bmatrix} \otimes \boldsymbol{I_3}.$$

Importantly, as demonstrated by the results presented in the Sec. 6.1, an internal reduced model with order $R = 1$ and time step size $h_k = 2.5 \times 10^{-2}\,\mathrm{s}$ is sufficient to ensure accurate predictions while maintaining minimal dimensionality of the reduced state. This, in turn, prevents excessive computational load on the optimizer and preserves real-time tractability.

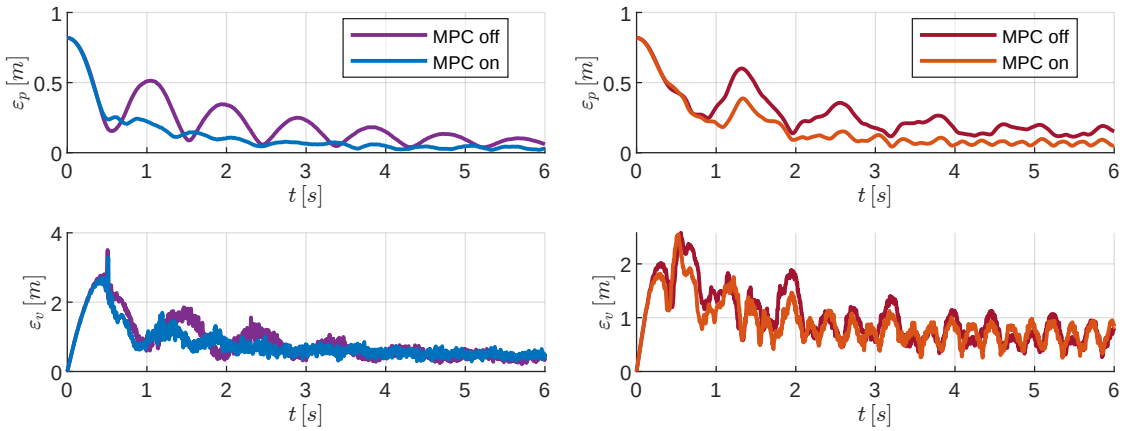
Table 7.1. Full-cable RMS tracking errors and average MPC iteration wall time in the setpoint regulation scenario.

(a) Free tip ($q = 0$).

Metric	No MPC	HiLQR	HiLQR-RTI
$\varepsilon_p^{rms}[\mathrm{m}]$	0.271	0.203	0.213
$\varepsilon_v^{rms}[\mathrm{m\,s^{-1}}]$	1.047	0.936	0.992
$h_w[\mathrm{ms}]$	–	49.2	13.5

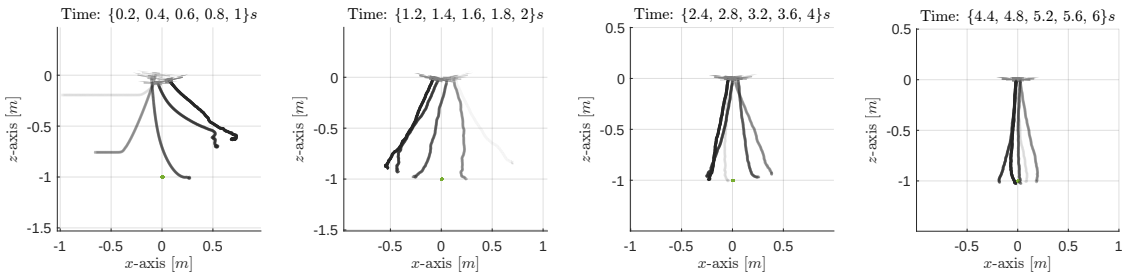
(b) Slung payload ($q = 1$).

Metric	No MPC	HiLQR	HiLQR-RTI
$\varepsilon_p^{rms}[\mathrm{m}]$	0.317	0.242	0.242
$\varepsilon_v^{rms}[\mathrm{m\,s^{-1}}]$	1.101	0.980	0.984
$h_w[\mathrm{ms}]$	–	46.5	14.3

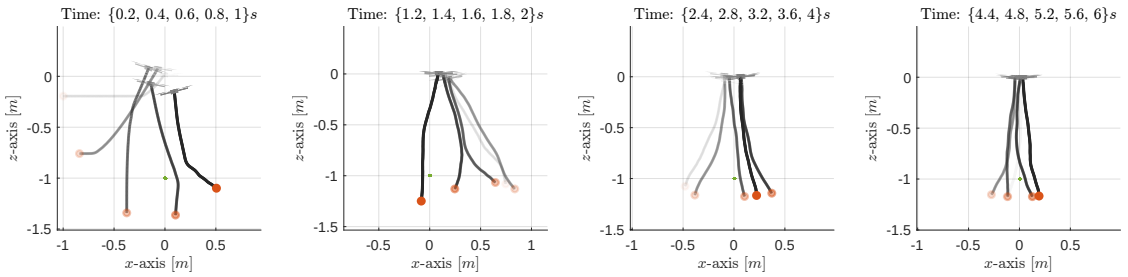


(a) Free tip ($q = 0$). **(b)** Slung payload ($q = 1$).

Figure 7.1. Time-histories position and velocity tracking errors of the whole cable during the setpoint regulation experiment.

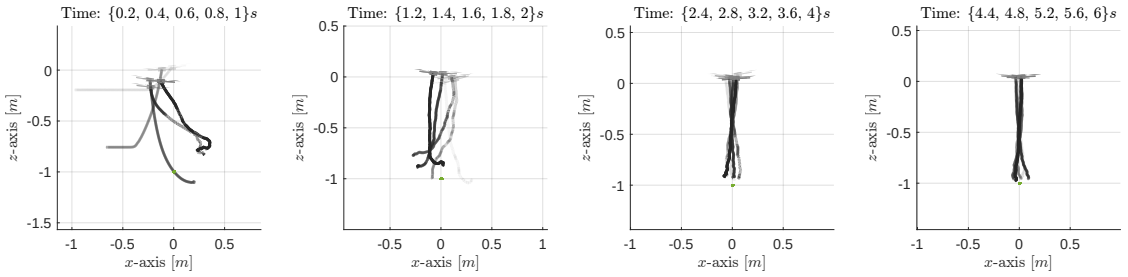


(a) Free tip ($q = 0$).

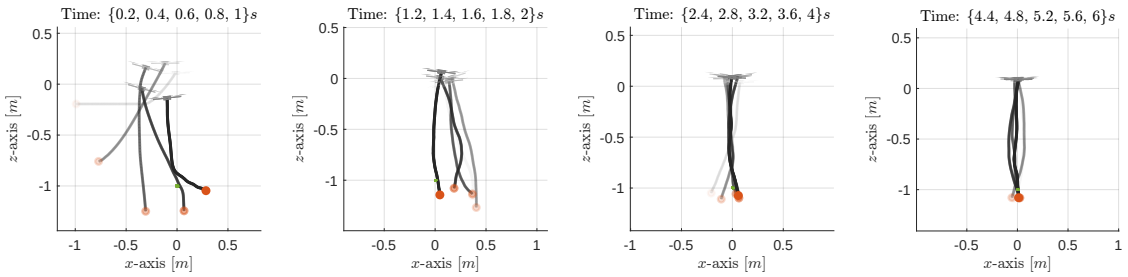


(b) Slung payload ($q = 1$).

Figure 7.2. Time-lapse of the aerial system without MPC (only geometric control) during setpoint regulation (side view).



(a) Free tip ($q = 0$).



(b) Slung payload ($q = 1$).

Figure 7.3. Time-lapse visualization of the aerial system with MPC enabled during setpoint regulation (side view).

After simulating for 10 s, the results clearly validate the initial expectations: both the temporal evolution of the tracking error metric (Fig. 7.1) and a visual inspection of the motion of the system (Fig. 7.2 and 7.3) confirm that the introduction of predictive control significantly enhances regulation performance, accelerating convergence to equilibrium by a considerable margin compared to the naive approach.
The limitations of the controller are also evident: its effectiveness is noticeably greater in the case without the payload, as the reduced load mass – given identical actuation and cable elasticity properties – results in increased control authority and a more responsive system. As a consequence, the damping of the oscillations in the error plots by virtue of the MPC is noticeably more pronounced in the first case compared to the second.
Tab. 7.1 summarizes the results of the experiment, providing a compact quantitative comparison between the RMS errors of the two strategies. It also provides the average wall time required for one iteration of the MPC optimization problem, which is crucial for real-time applications: in this regard, HiLQR-RTI represents a good compromise for ensuring practical applicability, as it achieves performance comparable to HiLQR while significantly reducing computational complexity.

7.2 Trajectory tracking

Online payload trajectory To demonstrate the capability of the control system to execute real-time aerial manipulation tasks without relying on the proposed offline planner – which is inherently computationally intensive – a representative scenario is considered. In this setting, the aerial system must approach a distant payload from above, as travel along the simplest, shortest path is forbidden – for instance because of obstacles. At the end, the payload must be grabbed as well.

The configuration for this experiment is detailed as follows:

- **Initial conditions**. The system is initialized at its static equilibrium (7.1), once without the payload ($q = 0$) and once with it attached ($q = 1$).
- **Guard conditions**. A neighbourhood around the final waypoint satisfies the guard condition associated with payload attachment or detachment. If the experiment starts with a free cable tip, the goal is to grab the payload:

$$\Lambda^{(0,1)} \coloneqq \{\boldsymbol{z} \in \mathcal{Z} \mid \underline{\boldsymbol{r}}^{\boldsymbol{N}} \in B_{0.1}(\begin{bmatrix}0 & 5 & -1\end{bmatrix}^\top)\}, \quad \Lambda^{(1,0)} \coloneqq \varnothing; \tag{7.2}$$

  vice versa if the payload is already attached, the goal is to release it:

$$\Lambda^{(0,1)} \coloneqq \varnothing, \quad \Lambda^{(1,0)} \coloneqq \{\boldsymbol{z} \in \mathcal{Z} \mid \underline{\boldsymbol{r}}^{\boldsymbol{N}} \in B_{0.1}(\begin{bmatrix}0 & 5 & -1\end{bmatrix}^\top)\}. \tag{7.3}$$

  Above, $\underline{\boldsymbol{r}}^{\boldsymbol{N}}$ represents the position of the last cable node in accordance with the underscore-notation introduced in Sec. 4.1, and $B_{0.1}(\begin{bmatrix}0 & 5 & -1\end{bmatrix})$ is a ball of radius 0.1 m around the final waypoint.
- **Reference**. The reference trajectory for the last node – namely $\hat{\boldsymbol{r}}^{\boldsymbol{N}}(t)$ and its derivatives – is constructed using a path of spline segments [30] that interpolate the following sequence of waypoints with continuity up to the second derivative:

$$\{\begin{bmatrix}0 & 0 & -1\end{bmatrix}^\top, \begin{bmatrix}0 & 0 & 2\end{bmatrix}^\top, \begin{bmatrix}0 & 5 & 2\end{bmatrix}^\top, \begin{bmatrix}0 & 5 & -1\end{bmatrix}^\top\}.$$

The temporal scaling is defined by a quintic profile similar to (6.2), computed over the total arc length of the complete path in the interval $[0\,\mathrm{s}, 8\,\mathrm{s}]$. For $t > 8$, the reference is locked to the final waypoint.
Conversely, generating a trajectory that is fully consistent with the full dynamics is not possible without resorting to the offline planner. Nevertheless, to improve convergence, the same reference velocity specified at the cable tip is applied to the entire system (UAV included) – a design choice motivated by empirical observations.

- **MPC configuration**. Compared to the previous section, the only change in the MPC configuration lies in the weighting matrices, i.e.:

$$\boldsymbol{S} \coloneqq \begin{bmatrix} \boldsymbol{S_p} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{S_v} \end{bmatrix}, \quad \boldsymbol{S_H} \coloneqq \boldsymbol{S}, \quad \boldsymbol{W} \coloneqq 0.5\boldsymbol{I_3},$$

with:

$$\boldsymbol{S_p} \coloneqq \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & 20 \end{bmatrix} \otimes \boldsymbol{I_3}, \quad \boldsymbol{S_v} \coloneqq \begin{bmatrix} 1 & \mathbf{0} & 0 \\ \mathbf{0} & 0.01\boldsymbol{I_{M-1}} & \mathbf{0} \\ 0 & \mathbf{0} & 0.1 \end{bmatrix} \otimes \boldsymbol{I_3}.$$

Note that null weights are assigned to the positions of almost all the nodes, with the exception of the cable tip. This choice reflects the absence of references for those nodes.

Table 7.2. Cable-tip RMS tracking errors and average MPC iteration wall time in the online trajectory tracking scenario.

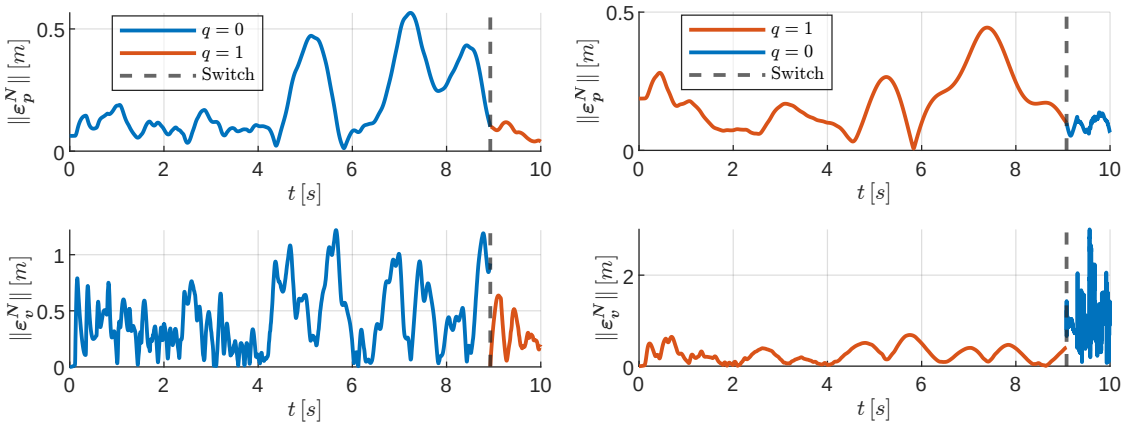
(a) Free-tip start ($q = 0$).

Metric	HiLQR	HiLQR-RTI
$\|\boldsymbol{\varepsilon_p^N}\|^{\mathrm{rms}}$ [m]	0.194	0.174
$\|\boldsymbol{\varepsilon_v^N}\|^{\mathrm{rms}}$ [m s^{-1}]	0.449	0.369
h_w [ms]	32.3	16.4

(b) Slung-payload start ($q = 1$).

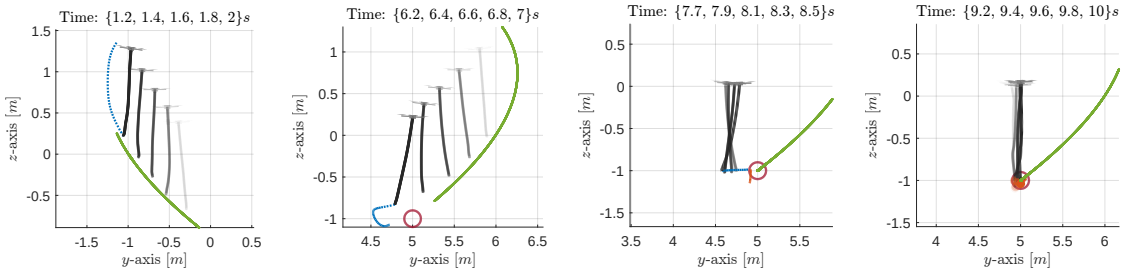
Metric	HiLQR	HiLQR-RTI
$\|\boldsymbol{\varepsilon_p^N}\|^{\mathrm{rms}}$ [m]	0.234	0.201
$\|\boldsymbol{\varepsilon_v^N}\|^{\mathrm{rms}}$ [m s^{-1}]	0.503	0.446
h_w [ms]	36.5	17.8

The results of a 10 s simulation prove that the aerial system successfully completes the assigned task online, without requiring the aid of the planner (see Tab. 7.2 and Fig. 7.4, 7.5). The cable and payload are manipulated with limited error, which as expected features a jump in velocity at $t \approx 9\,\mathrm{s}$ in both cases. The efficiency of the HiLQR-RTI solver is confirmed again, this time with even smaller RMS errors than the HiLQR – which does not necessarily elect a clear winner though, because the RTI variant is inherently less numerical robust. Notably, this scenario marks the first instance in which the MPC is observed to predict and optimize the evolution of the state across discrete

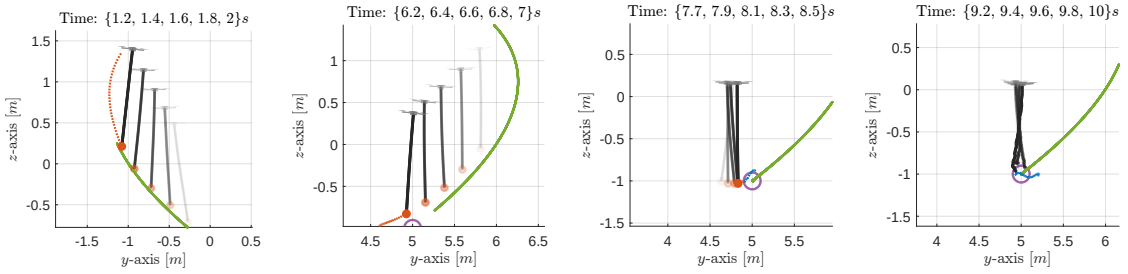


(a) Free-tip start ($q = 0$). **(b)** Slung-payload start ($q = 1$).

Figure 7.4. Time plots of the position and velocity tracking errors of the cable tip during the online trajectory tracking experiment.



(a) Free tip to slung payload ($q = 0 \to 1$).



(b) Slung payload to free tip ($q = 1 \to 0$).

Figure 7.5. Time-lapse of the MPC-controlled cable tip during online trajectory tracking (side view).

mode switches. As a matter of fact, in Fig. 7.5 the predicted transitions are visible at $t = 8.5\,\mathrm{s}$ for both trials.

Offline full-system trajectory When the offline planner is available for use, the control system becomes capable of executing more complex, dynamic, and even impressive maneuvers. For instance, consider a scenario in which a payload must be rapidly retrieved from a secondary location and subsequently released in correspondence of the origin – while (once again) deliberately avoiding the shortest path due to potential external reasons.

The setup is structured as follows:

- **Initial conditions**. The aerial system starts at rest without the payload ($q = 0$).
- **Guard conditions**. The same guard conditions as in the previous section (7.2) and (7.3) hold here too, but they're active simultaneously:

$$\Lambda^{(0,1)} \coloneqq \{\boldsymbol{z} \in \mathcal{Z} \mid \underline{\boldsymbol{r}}^{\boldsymbol{N}} \in B_{0.1}(\begin{bmatrix}0 & 5 & -1\end{bmatrix}^\top)\},$$
$$\Lambda^{(1,0)} \coloneqq \{\boldsymbol{z} \in \mathcal{Z} \mid \underline{\boldsymbol{r}}^{\boldsymbol{N}} \in B_{0.1}(\begin{bmatrix}0 & 0 & -1\end{bmatrix}^\top)\}.$$

- **Reference**. To formulate the reference trajectory, the segmentation strategy detailed in Sec. 4.3 is not necessary, as one segment alone already provides a satisfactory result. For its own reference, the planner is fed with (the usual) spline segments along the following sequence of cable-tip points:

$$\{\begin{bmatrix}0 & 0 & -1\end{bmatrix}^\top, \begin{bmatrix}1 & 2.5 & -1\end{bmatrix}^\top, \begin{bmatrix}0 & 5 & -1\end{bmatrix}^\top, \begin{bmatrix}-1 & 2.5 & -1\end{bmatrix}^\top, \begin{bmatrix}0 & 0 & -1\end{bmatrix}^\top\},$$

  with quintic timing law over $[0\,\mathrm{s}, 5\,\mathrm{s}]$, stopping at the last waypoint for $t > 5$.

- **Planner configuration**. Since the planner is executed only once and offline, it is not subject to real-time constraints, unlike the MPC. To achieve higher accuracy, reduced bases of order $R = 2$ are employed, and for improved numerical stability (from Sec. 6.1), the time step is halved to $h_k = 1.25 \times 10^{-2}\,\mathrm{s}$. The chosen weights are as follows:

$$\boldsymbol{S} \coloneqq \begin{bmatrix}\boldsymbol{S_p} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{S_v}\end{bmatrix}, \quad \boldsymbol{S_H} \coloneqq \boldsymbol{S}, \quad \boldsymbol{W} \coloneqq 0.1\boldsymbol{I_3},$$

  where:

$$\boldsymbol{S_p} \coloneqq \begin{bmatrix}\boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & 100\end{bmatrix} \otimes \boldsymbol{I_3}, \quad \boldsymbol{S_v} \coloneqq \begin{bmatrix}0.5 & \boldsymbol{0} & 0 \\ \boldsymbol{0} & 0.1\boldsymbol{I_{M-1}} & \boldsymbol{0} \\ 0 & \boldsymbol{0} & 1\end{bmatrix} \otimes \boldsymbol{I_3}.$$

- **MPC configuration**. For the controller, no additional precautions are required compared to the previous case, apart from the adjustment of the weight matrices:

$$\boldsymbol{S} \coloneqq \begin{bmatrix}\boldsymbol{S_p} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{S_v}\end{bmatrix}, \quad \boldsymbol{S_H} \coloneqq \boldsymbol{S}, \quad \boldsymbol{W} \coloneqq 5\boldsymbol{I_3},$$

  with:

$$\boldsymbol{S_p} \coloneqq \begin{bmatrix}30 & \boldsymbol{0} & 0 \\ \boldsymbol{0} & 0.1\boldsymbol{I_{M-1}} & \boldsymbol{0} \\ 0 & \boldsymbol{0} & 1\end{bmatrix} \otimes \boldsymbol{I_3}, \quad \boldsymbol{S_v} \coloneqq \begin{bmatrix}10 & \boldsymbol{0} & 0 \\ \boldsymbol{0} & 0.01\boldsymbol{I_{M-1}} & \boldsymbol{0} \\ 0 & \boldsymbol{0} & 0.1\end{bmatrix} \otimes \boldsymbol{I_3}.$$

Table 7.3. Whole-cable RMS tracking errors and average MPC iteration wall time in the offline trajectory tracking scenario.

Metric	HiLQR	HiLQR-RTI
$\varepsilon_p^{\text{rms}}$ [m]	0.287	0.285
$\varepsilon_v^{\text{rms}}$ [m s^{-1}]	1.068	1.063
h_w [ms]	36.8	16.8

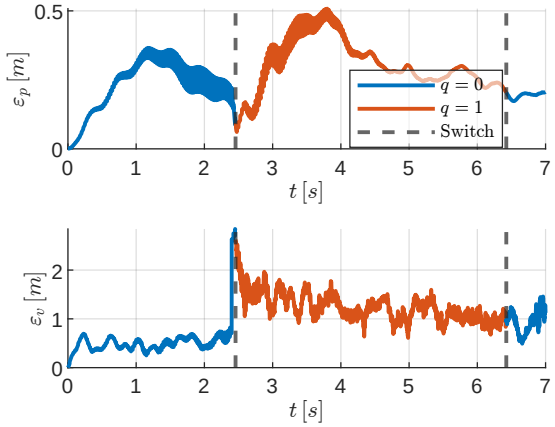


Figure 7.6. Time plots of the position and velocity tracking errors of the whole cable during the offline tracking experiment.

After simulating the scenario for 7 s, the results presented in Tab. 7.3 and Fig. 7.6 confirm the effectiveness of the proposed approach. The collaboration between the planner and the MPC successfully addresses the task, enabling the payload to be lifted and released at the designated locations, as proven by the time-lapse in Fig. 7.7. This demonstrates the feasibility of executing a highly dynamic maneuver with satisfactory performance in terms of both accuracy and speed. Notably, the real-time solver HiLQR-RTI proves again to be a viable option for this application, achieving average errors comparable to those of the full HiLQR implementation.

7.3 Obstacle avoidance

A pick-and-place task with obstacle avoidance represents the most complex scenario tackled in this work. Successfully executing it requires all components of the proposed control framework to operate in concert—from solving hybrid optimal control problems to applying sequential refinement strategies for generating optimal and feasible reference trajectories. To the best of the author's knowledge, such a task has not been addressed in the literature with comparable consideration for both endpoint objectives and internal cable constraints, while maintaining a similar level of accuracy and performance.
To this end, the following scene is set: a room contains a wall with a narrow gap or window, beyond which lies the payload to be transported. The objective is to navigate through the gap, pick up the payload, and return to the starting point by passing through the gap once more – without any part of the robot making contact with the edges of the window throughout the entire maneuver.

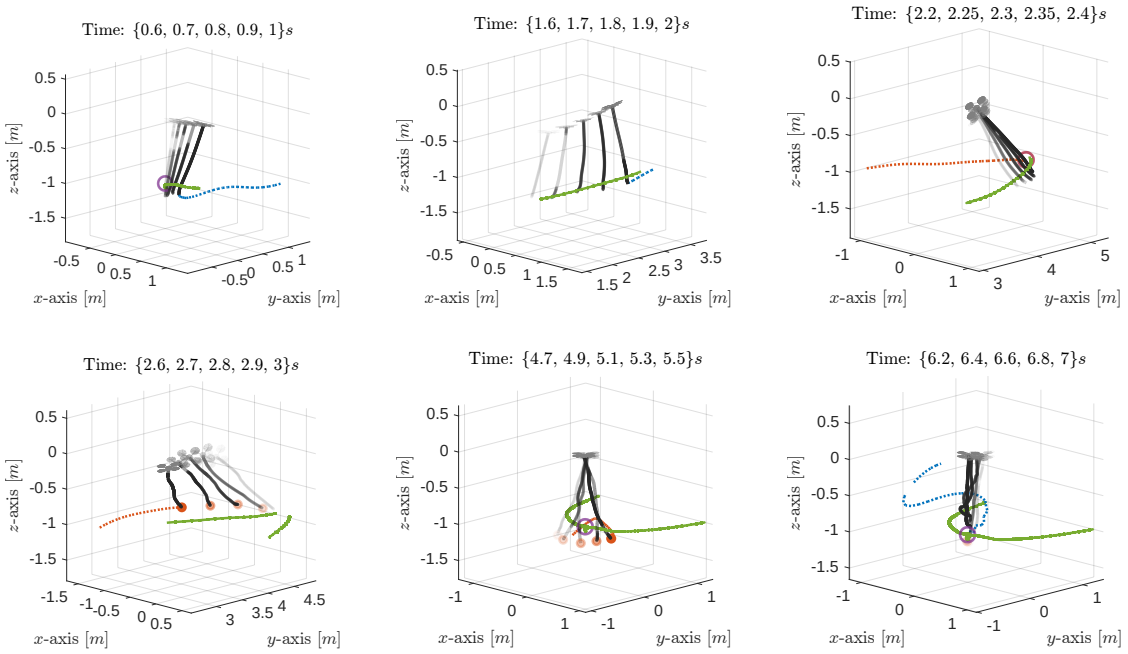


Figure 7.7. Time-lapse of the MPC-controlled system tracking the offline-computed trajectory.

The experiment is defined by the following specifications:

- **Initial conditions**. Also here, the system starts at equilibrium without the payload ($q = 0$).
- **Guard conditions**. The payload is assumed to be located 10 meters along the positive y-axis from the initial position, which also corresponds to the designated release point. Similarly to the previous cases, the geometrical guard conditions for attachment and detachment are:

$$\Lambda^{(0,1)} := \{\boldsymbol{z} \in \mathcal{Z} \mid \underline{\boldsymbol{r}}^{\boldsymbol{N}} \in B_{0.1}(\begin{bmatrix}0 & 10 & -1\end{bmatrix}^\top)\},$$
$$\Lambda^{(1,0)} := \{\boldsymbol{z} \in \mathcal{Z} \mid \underline{\boldsymbol{r}}^{\boldsymbol{N}} \in B_{0.1}(\begin{bmatrix}0 & 0 & -1\end{bmatrix}^\top)\}.$$

- **Constraints**. If the window is positioned halfway between the origin and the initial location of the payload, two ellipsoidal regions Υ_b and Υ_u can be defined to enclose respectively the lower and upper edges of the window:

$$\Gamma^q := \Upsilon_b \cup \Upsilon_u, \quad \forall q \in \{0, 1\}.$$

These ellipsoids are considered to have infinitely long axes along the x-direction, as motion along this axis is not expected this time, thus eliminating the need for lateral bounds. A safety margin of 0.2 m between the upper and lower borders of the window and the surfaces of the ellipsoids (upper and lower, respectively) ensures sufficient clearance and prevent any potential collision with the propellers of the quadrotor.

- **Reference**. When constraints are present, the highly-nonconvex optimization landscape makes it difficult to formulate an optimal feasible trajectory. For this reason, the problem under consideration requires leveraging the segmented planner to its full extent, and the trajectory is divided into four segments, each associated with one of the following fixed waypoints for the quadrotor to reach:

$$\{\begin{bmatrix}0 & 6 & 0\end{bmatrix}^\top, \begin{bmatrix}0 & 10 & 0\end{bmatrix}^\top, \begin{bmatrix}0 & 4 & 0\end{bmatrix}^\top, \begin{bmatrix}0 & 0 & 0\end{bmatrix}^\top\}.$$

  Each segment is assigned a nominal duration of 4 s; however, these boundaries may be softened during the final global replanning phase.
  In addition, a 2 s cooldown period with a static reference (the rest configuration of the system, with the cable tip placed at the current objective) is inserted right after each anticipated hybrid transition to allow the MPC to attenuate residual oscillations or vibrations in the cable before proceeding with the task. This last adjustment is applied directly to the stitched trajectory used as a warm-start for the global planner.

- **Planner configuration**. The ROM order and the time step length of the planner are unchanged. The gains are selected to encourage the system to reach the specified waypoints using any trajectory deemed suitable, while simultaneously limiting the velocity of the nodes throughout the maneuver. Accordingly, the following matrices are adopted:

$$\boldsymbol{S} \coloneqq \begin{bmatrix}\boldsymbol{0} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{S_v}\end{bmatrix}, \quad \boldsymbol{S_H} \coloneqq \begin{bmatrix}\boldsymbol{S_{H.p}} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}, \quad \boldsymbol{W} \coloneqq 0.1\boldsymbol{I_3},$$

  where:

$$\boldsymbol{S_v} \coloneqq 0.1\boldsymbol{I_{M+1}} \otimes \boldsymbol{I_3}, \quad \boldsymbol{S_{H.p}} \coloneqq \begin{bmatrix}10000 & \boldsymbol{0} & 0\\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0}\\ 0 & \boldsymbol{0} & 3000\end{bmatrix} \otimes \boldsymbol{I_3}.$$

  With regard to the log-barrier function homotopy scheme – introduced here as a fundamental component for the success of the planning procedure – the initial values and update coefficients are set as follows:

$$\kappa \coloneqq 1 \times 10^{-1}, \quad \delta \coloneqq 1 \times 10^{-3},$$
$$\eta_\kappa \coloneqq 0.95, \quad \eta_\delta \coloneqq 0.5.$$

- **MPC configuration**. The MPC is configuration is updated with the weight matrices specified below:

$$\boldsymbol{S} \coloneqq \begin{bmatrix}\boldsymbol{S_p} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{S_v}\end{bmatrix}, \quad \boldsymbol{S_H} \coloneqq \boldsymbol{S}, \quad \boldsymbol{W} \coloneqq \boldsymbol{I_3},$$

  using:

$$\boldsymbol{S_p} \coloneqq \begin{bmatrix}30 & \boldsymbol{0} & 0\\ \boldsymbol{0} & 1\boldsymbol{I_{N-1}} & \boldsymbol{0}\\ 0 & \boldsymbol{0} & 10\end{bmatrix} \otimes \boldsymbol{I_3}, \quad \boldsymbol{S_v} \coloneqq \begin{bmatrix}0.5 & \boldsymbol{0} & 0\\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0}\\ 0 & \boldsymbol{0} & 0.1\end{bmatrix} \otimes \boldsymbol{I_3}.$$

  It is likewise augmented with the log-barrier function, employing the same values for the parameters κ and δ, but with homotopy disabled. This corresponds to setting:

$$\eta_\kappa \coloneqq 1, \quad \eta_\delta \coloneqq 1.$$

Table 7.4. RMS tracking errors related to the whole cable and average MPC iteration wall time in the obstacle avoidance experiment. The HiLQR-RTI solver failed to converge in this scenario.

Metric	HiLQR	HiLQR-RTI
$\varepsilon_p^{\text{rms}}$ [m]	0.153	–
$\varepsilon_v^{\text{rms}}$ [m s^{-1}]	0.941	–
h_w [ms]	42.8	–

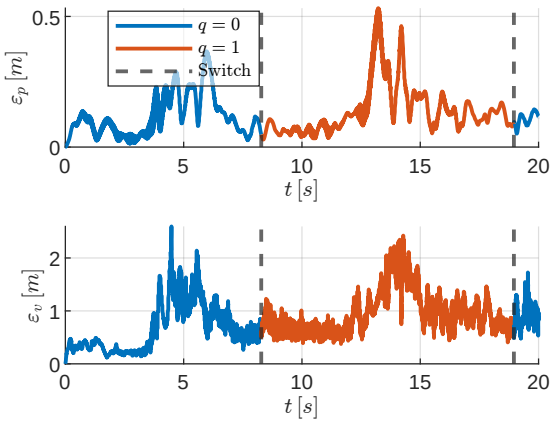


Figure 7.8. Time histories of the position and velocity tracking errors of the whole cable in the obstacle avoidance scenario.

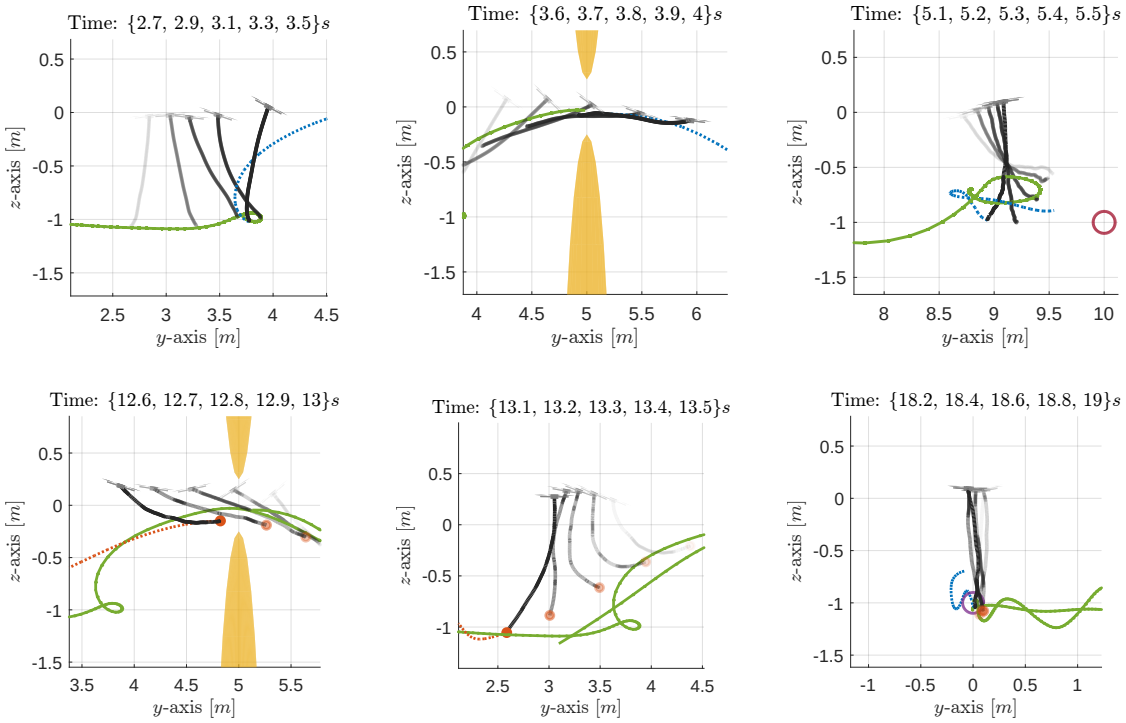


Figure 7.9. Time-lapse visualization of the obstacle avoidance experiment.

The results are obtained after 20 s of simulation. After looking at Tab. 7.4, it is first worth noting that, in this significantly more challenging scenario, the simplifications introduced by the HiLQR-RTI solver compromise the robustness of the controller, as this variant is unable to navigate the complex, non-convex optimization landscape and consequently fails to produce a feasible solution. Even the full HiLQR solver is affected, though not primarily in terms of accuracy, but rather in computation time – an increase attributable to the numerical stabilization techniques described in Appx. A.1. Nonetheless, the adopted strategy proves effective, enabling the aerial system to successfully complete the assigned pick-and-place task with limited error despite its high level of difficulty (Fig. 7.8 and 7.9).

Part IV

Conclusions

Chapter 8

Conclusions and outlook

8.1 Summary of contributions

This thesis introduced the first hybrid, continuum-aware framework that enables a quadrotor to manipulate a slung payload through an extensible cable while remaining computationally tractable. The key achievements are:

1. **Model-order reduction scheme.** A novel POD scheme, tailored to extensible string PDEs with dynamic boundary conditions, yields a reduced-order model in which only few deformation modes capture most of the cable strain energy. For this, constraining the POD basis to vanish at both ends proved decisive. Compared the original system, it offers improved numerical stability and computational efficiency. These advantages – combined with its demonstrated accuracy, adaptability, and robustness – make it particularly well suited to power real-time predictive control.

2. **Aerial system hybrid model.** A high-fidelity model of the aircraft system under consideration—comprising the quadrotor, tether, and payload—was developed by discretizing the governing PDEs via a finite difference scheme. The resulting model accurately captures the hybrid dynamics induced by payload pick-up and release, which are handled through boundary condition switches. This detailed simulation framework serves as a ground-truth reference and provides synthetic data for training the reduced-order model.

3. **Predictive control architecture.** The ROM is integrated into a hierarchical predictive optimal control framework specifically designed to govern the motion of the quadrotor while manipulating the cable-payload ensemble. The controller explicitly accounts for the hybrid nature of the dynamics, enabling it to predict not only the future evolution of the flexible cable but also the timing and nature of interactions with the payload. A comprehensive simulation campaign substantiates these results, proving – and quantitatively evaluating – the ability of the controller to stabilize the system and track complex trajectories involving both continuous motion and discrete transitions, even in the presence of obstacles. Moreover, the use of the ROM ensures that the overall control scheme remains computationally tractable, making it suitable for real-time implementation.

4. **Segmented trajectory generation.** An optimization-based planner, leveraging

the same ROM offline, generates dynamically feasible reference trajectories that smoothly traverse discrete modes and respect workspace constraints. These references enable the control module to execute aggressive pick-and-place maneuvers with precision. The algorithmic structure of the planner makes it a robust tool for translating high-level objectives – such as moving a payload beyond an obstacle – into valid, constraint-satisfying trajectories that complement and enhance the capabilities of the MPC.

Together, these contributions bridge the gap between oversimplified abstractions and intractable continuum models, enabling precise, agile, and robust hybrid aerial manipulation. In conclusion, this thesis presents a coherent end-to-end pipeline – spanning from high-fidelity continuum modeling to hybrid control and planning – that demonstrates strong performance across a range of simulated scenarios.

8.2 Directions for future work

Building on the results presented in this thesis, several promising directions emerge for further development, both in terms of theoretical refinement and practical implementation:

- **Improved handling of slack cable conditions.** Enhance the robustness of the controller when the payload rests on a surface and the cable becomes slack, a regime not contemplated by the current formulation that may cause degraded performance.
- **State estimation and filtering.** Introduce filtering strategies to improve robustness against sensor noise, latency, and model mismatch. In particular, measurements on the cable and payload that correspond to dynamics not captured by the reduced-order model should be attenuated. A multi-layered observer architecture, such as a cascaded or hybrid extended Kalman filter (EKF), could mitigate the impact of these artifacts.
- **Robustness validation.** Perform extensive simulations to evaluate closed-loop robustness with respect to model parameter uncertainties, external disturbances (e.g., wind gusts), and intermittent contact events.
- **Horizon tuning and stiffness-aware control.** Derive analytical relationships between the ROM dominant time scales and the optimal MPC prediction horizon. This analysis could guide prediction horizon selection and help mitigate stiffness-related numerical challenges, especially under constrained computational budgets. Where analytical expressions are intractable, the full FDM model can be leveraged instead.
- **Adaptive and learning-enhanced ROMs.** Extend the current POD-based model with online adaptation mechanisms to track slow parameter variations – such as changes in cable length or drag – without full retraining. Lightweight subspace updates could maintain accuracy over long deployments by adapting to evolving system dynamics.
- **Planner refinement with established methods.** Explore and adapt more general planning techniques – such as kinodynamic RRT* or augmented Lagrangian

methods – to improve efficiency, robustness, and scalability. These methods could extend the current approach while offering stronger guarantees for complex tasks.

- **End-effector and multi-contact modeling.** Augment the hybrid automaton with discrete grasp states and compliant contact dynamics, enabling richer interaction modes such as docking or suspended perching – where the end-effector stabilizes by anchoring itself to the environment, potentially even from below. These behaviors allow energy-efficient, stable interaction without requiring continuous UAV hover. Transition guards can be extended to capture contact-induced resets, detachment events, and frictional constraints.

- **Mass-switching hybrid modes.** Extend the hybrid model to account for multiple discrete mass configurations, enabling the system to represent operations such as staged payload release or progressive unloading. Each mode could correspond to a different total mass or mass distribution, with transitions triggered by delivery events.

- **Variable-length and multi-tether extensions.** Extend the continuum model to incorporate variable-length cables via reel mechanisms and multiple attachment points. These enhancements would enable capabilities such as obstacle-aware tether management, coordinated manipulation, and cooperative payload transport using multiple UAVs.

- **Hardware implementation and benchmarking.** Port the MATLAB-based controller to optimized C/C++ code and deploy it onboard a mid-scale UAV platform. Experimental validation should include tests under wind disturbances, sensor delays, and estimation noise. Reliable real-time performance will require integration with onboard state estimators, such as EKFs tailored to the system dynamics.

Addressing these challenges will further consolidate the viability of model-based control for tethered aerial manipulation and broaden its applicability to real-world scenarios involving contact, uncertainty, and interaction with dynamic environments.

Appendices

Appendix A

Detailed derivations

A.1 HiLQR algorithm

Hybrid iterative linear quadratic regulator (HiLQR) [17] is an extension to piecewise-smooth systems of iLQR [35], which is an efficient trajectory optimization algorithm for nonlinear, discrete-time control systems. It combines principles from optimal control and dynamic programming, and is widely used in real-time applications due to its speed and scalability.
This section presents an overview of the algorithm, specialised to the optimal control problem introduced in Sec. 4.1, namely (2.4).

Classic iLQR In accordance with a non-hybrid discrete-time dynamics such as:

$$\boldsymbol{z}(k+1) = \boldsymbol{G}(\boldsymbol{z}(k), \boldsymbol{v}(k)), \quad \forall k = 0, \dots, H-1, \tag{A.1}$$

over a prediction horizon of length H, the state sequence $\{\boldsymbol{z}(k)\}_{k=0}^{H}$ is uniquely identified by the initial state $\boldsymbol{z}(0)$ and the control sequence $\{\boldsymbol{v}(k)\}_{k=0}^{H-1}$.

The iLQR algorithm plays around the so-called *value function*, defined as the optimal cost-to-go from step $\underline{k}$ to the end of the horizon:

$$V(\underline{k}, \boldsymbol{z}(\underline{k})) = \min_{\{\boldsymbol{v}(k)\}_{k=\underline{k}}^{H-1}} \left(\sum_{k=\underline{k}}^{H-1} \ell(\boldsymbol{z}(k), \boldsymbol{v}(k)) + \bar{\ell}(\boldsymbol{z}(H)) \right),$$

where $\ell(\boldsymbol{z}(k), \boldsymbol{v}(k))$ is the running cost at step k, and $\bar{\ell}(\boldsymbol{z}(H))$ is the terminal cost at step H.
In particular, it holds that:

$$V(H, \boldsymbol{z}(H)) = \bar{\ell}(\boldsymbol{z}(H)). \tag{A.2}$$

Following Bellman's principle of optimality [5] and using (A.1), the value function can be rewritten as:

$$V(k, \boldsymbol{z}(k)) = \min_{\boldsymbol{v}(k)} \left(\ell(\boldsymbol{z}(k), \boldsymbol{v}(k)) + V(k+1, \boldsymbol{G}(\boldsymbol{z}(k), \boldsymbol{v}(k))) \right).$$

The argument of the minimization in the latter is the well-known *Q-function*:

$$Q(k, \boldsymbol{z}(k), \boldsymbol{v}(k)) = \ell(\boldsymbol{z}(k), \boldsymbol{v}(k)) + V(k+1, \boldsymbol{G}(\boldsymbol{z}(k), \boldsymbol{v}(k))), \tag{A.3}$$

which results in:

$$V(k, \boldsymbol{z}(k)) = \min_{\boldsymbol{v}(k)} Q(k, \boldsymbol{z}(k), \boldsymbol{v}(k)). \tag{A.4}$$

From this point onward through this section, to improve readability the dependence on the step index k is suppressed, and the quantities referred to the next step $k+1$ are denoted with the superscript $'$, except where otherwise specified.
Consider a nominal state-control pair $(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})$, with $\underline{\boldsymbol{z}}' = \boldsymbol{G}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})$. The Q-function can be approximated by a second-order Taylor expansion around these:

$$\begin{aligned} Q(\underline{\boldsymbol{z}} + \boldsymbol{\delta z}, \underline{\boldsymbol{v}} + \boldsymbol{\delta v}) \approx{}& Q(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) + Q_{\boldsymbol{z}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})\boldsymbol{\delta z} + \\ &+ Q_{\boldsymbol{v}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})\boldsymbol{\delta v} + \frac{1}{2}\begin{bmatrix}\boldsymbol{\delta z}\\ \boldsymbol{\delta v}\end{bmatrix}^\top \begin{bmatrix} Q_{\boldsymbol{zz}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) & Q_{\boldsymbol{zv}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) \\ Q_{\boldsymbol{vz}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) & Q_{\boldsymbol{vv}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) \end{bmatrix} \begin{bmatrix}\boldsymbol{\delta z}\\ \boldsymbol{\delta v}\end{bmatrix}. \end{aligned} \tag{A.5}$$

In the latter, the subscripts refer to the variables with respect to which the partial derivatives are taken, hence $Q_{\boldsymbol{z}}$ is the Jacobian of the Q-function with respect to the state, $Q_{\boldsymbol{v}}$ is the Jacobian with respect to the control, $Q_{\boldsymbol{zz}}$ is the Hessian with respect to the state, $Q_{\boldsymbol{vv}}$ is the Hessian with respect to the control, and $Q_{\boldsymbol{zv}}$ and $Q_{\boldsymbol{vz}}$ are the cross-derivatives. This assumes sufficient smoothness of the dynamics and cost functions, typically requiring that $\boldsymbol{G}$, ℓ, and $\bar{\ell}$ be at least twice differentiable with respect to their arguments.
The actual values of the derivatives of the Q-function can be found by differentiating also (A.3) with the chain rule and equating the coefficients of the same order[1]:

$$\begin{aligned} Q_{\boldsymbol{z}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) &= \ell_{\boldsymbol{z}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) + V'_{\boldsymbol{z}}(\underline{\boldsymbol{z}}')\boldsymbol{G}_{\boldsymbol{z}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}), \\ Q_{\boldsymbol{v}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) &= \ell_{\boldsymbol{v}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) + V'_{\boldsymbol{z}}(\underline{\boldsymbol{z}}')\boldsymbol{G}_{\boldsymbol{v}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}), \\ Q_{\boldsymbol{zz}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) &\approx \ell_{\boldsymbol{zz}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) + \boldsymbol{G}_{\boldsymbol{z}}^\top(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})V'_{\boldsymbol{zz}}(\underline{\boldsymbol{z}}')\boldsymbol{G}_{\boldsymbol{z}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}), \\ Q_{\boldsymbol{vv}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) &\approx \ell_{\boldsymbol{vv}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) + \boldsymbol{G}_{\boldsymbol{v}}^\top(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})V'_{\boldsymbol{zz}}(\underline{\boldsymbol{z}}')\boldsymbol{G}_{\boldsymbol{v}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}), \\ Q_{\boldsymbol{zv}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) &\approx \ell_{\boldsymbol{zv}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) + \boldsymbol{G}_{\boldsymbol{z}}^\top(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})V'_{\boldsymbol{zz}}(\underline{\boldsymbol{z}}')\boldsymbol{G}_{\boldsymbol{v}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}), \\ Q_{\boldsymbol{vz}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) &= Q_{\boldsymbol{zv}}^\top(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}). \end{aligned} \tag{A.6}$$

The optimal control adjustment in response to a state deviation $\boldsymbol{\delta z}$ is obtained by minimizing the Q-function with respect to $\boldsymbol{\delta v}$, yielding the local state-affine policy:

$$\boldsymbol{\delta v}^\star = \boldsymbol{\pi}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) + \boldsymbol{\Pi}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})\boldsymbol{\delta z}, \tag{A.7}$$

with coefficients given by:

$$\begin{aligned} \boldsymbol{\pi}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) &= -Q_{\boldsymbol{vv}}^{-1}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})Q_{\boldsymbol{v}}^\top(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}), \\ \boldsymbol{\Pi}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) &= -Q_{\boldsymbol{vv}}^{-1}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})Q_{\boldsymbol{zv}}^\top(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}). \end{aligned} \tag{A.8}$$

Similarly, the value function is expanded as well:

$$V(\underline{\boldsymbol{z}} + \boldsymbol{\delta z}) \approx V(\underline{\boldsymbol{z}}) + V_{\boldsymbol{z}}(\underline{\boldsymbol{z}})\boldsymbol{\delta z} + \frac{1}{2}\boldsymbol{\delta z}^\top V_{\boldsymbol{zz}}(\underline{\boldsymbol{z}})\boldsymbol{\delta z}.$$

[1] If the expansion of (A.1) stops at the first order, the algorithm is called iterative linear quadratic regulator (iLQR); however, if the dynamics is expanded up to the second order, one has differential dynamic programming (DDP). The former is is preferred in real-time control due to its robustnes and lower computational cost.

Using (A.4) with $\boldsymbol{z} = \underline{\boldsymbol{z}} + \boldsymbol{\delta z}$ and $\boldsymbol{v} = \underline{\boldsymbol{v}} + \boldsymbol{\delta v}$, and plugging (A.7) into (A.5), one can extract the gradient and Hessian of the value function, whose expressions, once simplified according to [35], are:

$$\begin{aligned} V_z(\underline{\boldsymbol{z}}) &= Q_z(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) - \boldsymbol{\pi}^\top(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) Q_{vv}^{-1}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) \boldsymbol{\Pi}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}), \\ V_{zz}(\underline{\boldsymbol{z}}) &= Q_{zz}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) - \boldsymbol{\Pi}^\top(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) Q_{vv}^{-1}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) \boldsymbol{\Pi}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}). \end{aligned} \tag{A.9}$$

The algorithm proceeds iteratively, starting from an initial nominal control sequence $\{\underline{\boldsymbol{v}}(k)\}_{k=0}^{H-1}$, following the steps below:

1. **Rollout.** From the initial state $\underline{\boldsymbol{z}}(0) = \boldsymbol{z}(0)$, compute the nominal trajectory $\{\underline{\boldsymbol{z}}(k)\}_{k=0}^{H}$ by recursively applying the discrete-time dynamics (A.1) with the current nominal control sequence.

2. **Backward pass.** At the end of the horizon, using the identity (A.2), evaluate the value function $V(H, \underline{\boldsymbol{z}}(H))$ and its derivatives in $\boldsymbol{z}$. This unlocks the ability to compute the coefficients of the Q-function expansion around the nominal pair using (A.6), then the gains (A.8), and finally the coefficients of the value function expansion using (A.9). These last three steps have to be repeated backward $\forall k = H-1, \dots, 0$ until reaching the initial state.

3. **Forward pass.** Using the computed gains, first update the nominal control with the local state-affine policy, coming from (A.7):

   $$\boldsymbol{v} = \underline{\boldsymbol{v}} + \boldsymbol{\pi}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) + \boldsymbol{\Pi}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) (\boldsymbol{z} - \underline{\boldsymbol{z}})$$

   and then compute the corresponding next state $\boldsymbol{z}'$. These two steps are repeated forward $\forall k = 0, \dots, H-1$, until reaching the end of the horizon. At this point, the nominal trajectory is updated to $\{\boldsymbol{z}(k)\}_{k=0}^{H}$ and the nominal control sequence to $\{\boldsymbol{v}(k)\}_{k=0}^{H-1}$.

The alternation of backward and forward passes goes on until a satisfactory solution is found, often determined by a cost-based convergence criterion.

To ensure numerical stability and promote convergence, the algorithm is typically complemented by two stabilization strategies: line search and Levenberg-Marquardt regularization. For the implementation details and further theoretical insights, the reader is referred to [35].

Hybrid extension For this, consider a discrete-time hybrid automaton as the one introduced in Sec. 2.1, namely (2.2).
Besides straightforwardly using the discrete-time hybrid successor (2.3) in the rollout and forward pass, the crux of the HiLQR algorithm lies in the backward pass, where the so-called *saltation matrix* [18] is introduced to account for the discontinuities. The saltation matrix maps first-order perturbations of the state $\boldsymbol{\delta z}$ before and after the jump. Prior to defining it, with the usual convention and notation, an alternative description of the guard conditions is introduced:

$$\Lambda^e = \{\boldsymbol{z} \in \mathcal{Z} \mid \lambda^e(\boldsymbol{z}) = 0\},$$

for some edge $(q^-, q^+) = e \in \mathcal{E}$. This equation says that Λ^e is the zero-level set of a scalar guard function $\lambda^e : \mathcal{Z} \to \mathbb{R}$, meaning that, when the continuous state $\boldsymbol{z}$ is such that $\lambda^e(\boldsymbol{z}) = 0$, the system will transition along the edge e.
The saltation matrix for the hybrid system at hand is defined as:

$$\boldsymbol{\Xi}^e(\boldsymbol{z}, \boldsymbol{v}) \coloneqq \boldsymbol{\rho}_{\boldsymbol{z}}^e(\boldsymbol{z}) + \frac{\boldsymbol{G}^{q^+}(\boldsymbol{\rho}^e(\boldsymbol{z}), \boldsymbol{v}) - \boldsymbol{\rho}_{\boldsymbol{z}}^e(\boldsymbol{z})\boldsymbol{G}^{q^-}(\boldsymbol{z}, \boldsymbol{v})}{\lambda_{\boldsymbol{z}}^e(\boldsymbol{z})\boldsymbol{G}^{q^-}(\boldsymbol{z}, \boldsymbol{v})}\lambda_{\boldsymbol{z}}^e(\boldsymbol{z})$$

With the simplification that the switch occurs always at the end of the time step, the coefficients of the Q-function expansion at the switching steps (identified during the rollout/forward pass) are modified to account for the discontinuities. One has:

$$\begin{aligned}
Q_{\boldsymbol{z}}(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) &= \ell_{\boldsymbol{z}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) + V'_{\boldsymbol{z}}(\underline{\boldsymbol{z}}')\boldsymbol{\Xi}(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})\boldsymbol{G}_{\boldsymbol{z}}(q^-, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}),\\
Q_{\boldsymbol{v}}(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) &= \ell_{\boldsymbol{v}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) + V'_{\boldsymbol{z}}(\underline{\boldsymbol{z}}')\boldsymbol{\Xi}(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})\boldsymbol{G}_{\boldsymbol{v}}(q^-, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}),\\
Q_{\boldsymbol{zz}}(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) &\approx \ell_{\boldsymbol{zz}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) + \boldsymbol{G}_{\boldsymbol{z}}^\top(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})\boldsymbol{\Xi}^\top(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})V'_{\boldsymbol{zz}}(\underline{\boldsymbol{z}}')\boldsymbol{\Xi}(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})\boldsymbol{G}_{\boldsymbol{z}}(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}),\\
Q_{\boldsymbol{vv}}(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) &\approx \ell_{\boldsymbol{vv}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) + \boldsymbol{G}_{\boldsymbol{v}}^\top(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})\boldsymbol{\Xi}^\top(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})V'_{\boldsymbol{zz}}(\underline{\boldsymbol{z}}')\boldsymbol{\Xi}(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})\boldsymbol{G}_{\boldsymbol{v}}(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}),\\
Q_{\boldsymbol{zv}}(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) &\approx \ell_{\boldsymbol{zv}}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) + \boldsymbol{G}_{\boldsymbol{z}}^\top(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})\boldsymbol{\Xi}^\top(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})V'_{\boldsymbol{zz}}(\underline{\boldsymbol{z}}')\boldsymbol{\Xi}(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}})\boldsymbol{G}_{\boldsymbol{v}}(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}),\\
Q_{\boldsymbol{vz}}(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}) &= Q_{\boldsymbol{zv}}^\top(e, \underline{\boldsymbol{z}}, \underline{\boldsymbol{v}}).
\end{aligned}$$

The extension of the gains (A.8) and the value function derivatives (A.9) is trivial.

Bibliography

[1] Joel A E Andersson et al. "CasADi – A software framework for nonlinear optimization and optimal control". In: *Mathematical Programming Computation* 11.1 (2019), pp. 1–36. DOI: 10.1007/s12532-018-0139-4.

[2] S. Antman. *Nonlinear Problems of Elasticity*. Applied Mathematical Sciences. Springer New York, 2005. ISBN: 9780387276496. URL: https://books.google.it/books?id=_MagNyCXNqMC.

[3] Kendall E. Atkinson. *An Introduction to Numerical Analysis*. Second. New York: John Wiley & Sons, 1989. ISBN: 0471500232. URL: http://www.worldcat.org/isbn/0471500232.

[4] Tianshu Bao et al. "A New Hybrid Automaton Framework with Partial Differential Equation Dynamics". In: *ArXiv* abs/2404.11900 (2024). URL: https://api.semanticscholar.org/CorpusID:269214406.

[5] Richard Bellman. *Dynamic Programming*. Dover Publications, 1957. ISBN: 9780486428093.

[6] P. Bolzern, R. Scattolini, and N. Schiavoni. *Fondamenti di controlli automatici*. McGraw-Hill Education, 2025. ISBN: 9788838613166. URL: https://books.google.it/books?id=XrsQ0QEACAAJ.

[7] Peng Chang et al. "A new adaptive control design for a quadrotor system with suspended load by an elastic rope". In: *Nonlinear Dynamics* 111.20 (2023), pp. 19073–19092. ISSN: 1573-269X. DOI: 10.1007/s11071-023-08847-3. URL: https://doi.org/10.1007/s11071-023-08847-3.

[8] Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. "A Multilinear Singular Value Decomposition". In: *SIAM Journal on Matrix Analysis and Applications* 21.4 (2000), pp. 1253–1278. DOI: 10.1137/S0895479896305696. eprint: https://doi.org/10.1137/S0895479896305696. URL: https://doi.org/10.1137/S0895479896305696.

[9] Kenth Engø-Monsen. "On the BCH-formula in so(3)". In: *Bit Numerical Mathematics* 41 (Dec. 2000), pp. 629–632. DOI: 10.1023/A:1021979515229.

[10] Philipp Foehn et al. "Fast trajectory optimization for agile quadrotor maneuvers with a cable-suspended payload". In: (July 2017).

[11] Gene Franklin, J.D. Powell, and M.L. Workman. *Digital Control of Dynamic Systems-Third Edition*. Nov. 2022. ISBN: ISBN: 0-9791226-3-5 or ISBN13: 978-0-9791226-3-7.

[12] Chiara Gabellieri and Antonio Franchi. "Differential Flatness and Manipulation of Elasto-flexible Cables Carried by Aerial Robots in a Possibly Viscous Environment". In: *2023 International Conference on Unmanned Aircraft Systems (ICUAS)*. 2023, pp. 963–968. DOI: 10.1109/ICUAS57906.2023.10156297.

[13] Rafal Goebel, Ricardo G. Sanfelice, and Andrew R. Teel. "Hybrid dynamical systems". In: *IEEE Control Systems Magazine* 29.2 (2009), pp. 28–93. DOI: 10.1109/MCS.2008.931718.

[14] Farhad A. Goodarzi, Daewon Lee, and Taeyoung Lee. "Geometric control of a quadrotor UAV transporting a payload connected via flexible cable". In: *International Journal of Control, Automation and Systems* 13.6 (2015), pp. 1486–1498. DOI: 10.1007/s12555-014-0304-0. URL: https://doi.org/10.1007/s12555-014-0304-0.

[15] Jacob R. Goodman, Thomas Beckers, and Leonardo J. Colombo. "Geometric Control for Load Transportation With Quadrotor UAVs by Elastic Cables". In: *IEEE Transactions on Control Systems Technology* 31.6 (2023), pp. 2848–2862. DOI: 10.1109/TCST.2023.3296730.

[16] Olivier Goury and Christian Duriez. "Fast, Generic, and Reliable Control and Simulation of Soft Robots Using Model Order Reduction". In: *IEEE Transactions on Robotics* 34.6 (2018), pp. 1565–1576. DOI: 10.1109/TRO.2018.2861900.

[17] Nathan J. Kong, George Council, and Aaron M. Johnson. *iLQR for Piecewise-Smooth Hybrid Dynamical Systems*. 2021. arXiv: 2103.14584 [cs.RO]. URL: https://arxiv.org/abs/2103.14584.

[18] Nathan J. Kong et al. "Saltation Matrices: The Essential Tool for Linearizing Hybrid Dynamical Systems". In: *Proceedings of the IEEE* 112.6 (June 2024), pp. 585–608. ISSN: 1558-2256. DOI: 10.1109/jproc.2024.3440211. URL: http://dx.doi.org/10.1109/JPROC.2024.3440211.

[19] Prasanth Kotaru, Guofan Wu, and Koushil Sreenath. "Dynamics and control of a quadrotor with a payload suspended through an elastic cable". In: *2017 American Control Conference (ACC)*. 2017, pp. 3906–3913. DOI: 10.23919/ACC.2017.7963553.

[20] Taeyoung Lee, Melvin Leok, and N. Harris McClamroch. "Control of Complex Maneuvers for a Quadrotor UAV using Geometric Methods on SE(3)". In: *arXiv e-prints*, arXiv:1003.2005 (Mar. 2010), arXiv:1003.2005. DOI: 10.48550/arXiv.1003.2005. arXiv: 1003.2005 [math.OC].

[21] Guanrui Li and Giuseppe Loianno. "Nonlinear Model Predictive Control for Cooperative Transportation and Manipulation of Cable Suspended Payloads with Multiple Quadrotors". In: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Oct. 2023, pp. 5034–5041. DOI: 10.1109/iros55552.2023.10341785. URL: http://dx.doi.org/10.1109/IROS55552.2023.10341785.

[22] Zhijie Liu, Jinkun Liu, and Wei He. "Deadzone compensation based boundary control of a flexible aerial refueling hose with output constraint". In: *IFAC-PapersOnLine* 50.1 (2017). 20th IFAC World Congress, pp. 645–650. ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2017.08.113. URL: https://www.sciencedirect.com/science/article/pii/S2405896317301477.

[23] Kyle Niemeyer. *Finite-Difference Solution for Boundary Value Problems*. https://kyleniemeyer.github.io/ME373-book/content/bvps/finite-difference.html. 2023. URL: https://kyleniemeyer.github.io/ME373-book/content/bvps/finite-difference.html.

[24] Anibal Ollero et al. "Past, Present, and Future of Aerial Robotic Manipulators". In: *IEEE Transactions on Robotics* 38.1 (2022), pp. 626–645. DOI: 10.1109/TRO.2021.3084395.

[25] Ivana Palunko, Rafael Fierro, and Patricio Cruz. "Trajectory generation for swing-free maneuvers of a quadrotor with suspended payload: A dynamic programming approach". In: *2012 IEEE International Conference on Robotics and Automation*. 2012, pp. 2691–2697. DOI: 10.1109/ICRA.2012.6225213.

[26] Igor Henrique Beloti Pizetta, Alexandre Santos Brandão, and Mário Sarcinelli-Filho. "Modelling and control of a PVTOL quadrotor carrying a suspended load". In: *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*. 2015, pp. 444–450. DOI: 10.1109/ICUAS.2015.7152321.

[27] Johnny Quisenberry and Andrew Arena. "Discrete Cable Modeling and Dynamic Analysis". In: *44th AIAA Aerospace Sciences Meeting and Exhibit*. 2012. DOI: 10.2514/6.2006-424. eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.2006-424. URL: https://arc.aiaa.org/doi/abs/10.2514/6.2006-424.

[28] Mrunal Sarvaiya, Guanrui Li, and Giuseppe Loianno. *HPA-MPC: Hybrid Perception-Aware Nonlinear Model Predictive Control for Quadrotors with Suspended Loads*. 2024. arXiv: 2411.11982 [cs.RO]. URL: https://arxiv.org/abs/2411.11982.

[29] Yaolei Shen, Chiara Gabellieri, and Antonio Franchi. *Aerial Robots Carrying Flexible Cables: Dynamic Shape Optimal Control via Spectral Method Model*. English. WorkingPaper. ArXiv.org, Mar. 2024. DOI: 10.48550/arXiv.2403.17565.

[30] Bruno Siciliano et al. *Robotics: Modelling, Planning and Control*. 1st. Springer Publishing Company, Incorporated, 2008. ISBN: 1846286417.

[31] Koushil Sreenath, Taeyoung Lee, and Vijay Kumar. "Geometric control and differential flatness of a quadrotor UAV with a cable-suspended load". In: *52nd IEEE Conference on Decision and Control*. 2013, pp. 2269–2274. DOI: 10.1109/CDC.2013.6760219.

[32] Kunihiko Taira et al. "Modal Analysis of Fluid Flows: An Overview". In: *AIAA Journal* 55.12 (Dec. 2017), pp. 4013–4041. ISSN: 1533-385X. DOI: 10.2514/1.j056060. URL: http://dx.doi.org/10.2514/1.J056060.

[33] Sarah Tang and Vijay Kumar. "Mixed Integer Quadratic Program trajectory generation for a quadrotor with a cable-suspended payload". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. 2015, pp. 2216–2222. DOI: 10.1109/ICRA.2015.7139492.

[34] Yunxi Tang et al. *Inequality Constrained Trajectory Optimization with A Hybrid Multiple-shooting iLQR*. 2023. arXiv: 2109.07131 [cs.RO]. URL: https://arxiv.org/abs/2109.07131.

[35] Yuval Tassa, Tom Erez, and Emanuel Todorov. "Synthesis and stabilization of complex behaviors through online trajectory optimization". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2012, pp. 4906–4913. DOI: 10.1109/IROS.2012.6386025.

[36] The Engineering ToolBox. *Young's Modulus, Tensile Strength and Yield Strength Values for some Materials*. 2003. URL: https://www.engineeringtoolbox.com/young-modulus-d_417.html.

[37] Marilena Vendittelli. *Modeling and Control of Multi-Rotor UAVs*. Course material, Sapienza University of Rome. 2024.

[38] Haokun Wang et al. "Impact-Aware Planning and Control for Aerial Robots With Suspended Payloads". In: *IEEE Transactions on Robotics* 40 (2024), pp. 2478–2497. DOI: 10.1109/TRO.2024.3381555.

[39] Chenyu Yang et al. *Collaborative Navigation and Manipulation of a Cable-towed Load by Multiple Quadrupedal Robots*. 2022. arXiv: 2206.14424 [cs.RO]. URL: https://arxiv.org/abs/2206.14424.

[40] Jun Zeng, Prasanth Kotaru, and Koushil Sreenath. "Geometric Control and Differential Flatness of a Quadrotor UAV with Load Suspended from a Pulley". In: *2019 American Control Conference (ACC)*. 2019, pp. 2420–2427. DOI: 10.23919/ACC.2019.8815173.