DISTRIBUTED TIME-CRITICAL COORDINATION STRATEGIES FOR UNMANNED AERIAL SYSTEMS IN CLUTTERED ENVIRONMENTS

BY

JAVIER PUIG NAVARRO

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Aerospace Engineering
in the Graduate College of the
University of Illinois Urbana-Champaign, 2021

Urbana, Illinois

Doctoral Committee:

Professor Naira Hovakimyan, Chair
Professor Daniel Liberzon
Professor Srinivasa Salapaka
Professor Cedric Langbort
Dr. B. Danette Allen, NASA Langley Research Center

Abstract

This thesis addresses the problem of cooperative motion planning and control for a group of cooperating unmanned aerial systems through cluttered and uncertain environments, subject to a broad range of coordination and temporal constraints. The proposed solution expands the type of time-critical missions that can be automated using cooperative motion control frameworks.

This work introduces the use of novel geometric queries to aid a sample-based motion-planning algorithm guide the growth of a rapidly-exploring random tree through the narrow passages in cluttered and uncertain scenarios. To this effect, specific silhouette and tolerance verification queries are designed for the geometric objects that represent vehicle motion and environmental obstacles. The combination of the silhouette-informed path planner with a CNC-inspired path-smoothing method, and a centralized cooperative speed-assignment algorithm yields a set of $\mathcal{C}^2$ continuous trajectories that maintain safe separation with all uncertain obstacles and cooperating peers, meet desired mission constraints, and satisfy a set of simplified dynamic constraints.

The vehicles are then tasked to follow their assigned paths and coordinate online to meet mission objectives, desired inter-agent spacing constraints, and temporal constraints—such as a time of arrival or a window of arrival. The thesis introduces two types of inter-agent spacing constraints—tight and loose coordination—and three types of temporal constraints—unenforced, relaxed, and strict—that result in six general time-critical coordination strategies. This thesis presents six distributed coordination protocols to enforce this range of constraints. These coordination protocols rely on a lossy communication network that can be disconnected pointwise in time at all times, but is connected in an integral sense over a sliding temporal window. This work derives transient and steady-state performance bounds for the tight coordination protocols. Simulation results through a cluttered urban-like environment, where vehicles are subject to wind disturbances, corroborate the theoretical results.

Para Carlos, mamá y papá.

Acknowledgments

I would like to recognize my adviser Prof. Naira Hovakimyan for her guidance and support throughout the completion of my Ph.D. degree. The depth and variety of research projects my adviser has exposed me to, and her relentless pursuance of technical rigor have shaped this work considerably.

I feel gratitude for all the friends and colleagues at the University of Illinois at Urbana-Champaign who have provided valuable counsel and support. In particular, Enric Xargay for introducing me to the field of research in adaptive control and networked systems, the many in-depth discussions we have had about some of the theorems within this dissertation, and overall for being a good friend. *Gràcies de debò.* Ronald Choe for his advice on Pythagorean Hodograph Bézier curves, always accompanied with kindness and patience. Kasey Ackerman for endless hours of help and sleep deprivation during the F-16 VISTA and Learjet flight tests, as well as being a supportive roommate. Bilal Mehdi and Aditya Gahlawat for their lively and friendly discussions.

I would also like to express my appreciation to the team at NASA Langley Research Center. Especially, Dr. Danette Allen for always opening her door to provide research, management, and life advice, as well as her willingness to overcome bureaucracy and many other hurdles to bring me on center as a team member. Dr. Natalia Alexandrov for sharing her experience and vision, and providing accurate and honest advice for the ATTRACTOR project. Ben Kelley, Matt Vaughan, and Dr. Loc Tran for teaching me how to transform my ideas into much better code, while promoting a friendly and healthy research environment. I would also like to acknowledge the other members in my Ph.D. committee for their constructive feedback: Prof. Cedric Langbort, Prof. Daniel Liberzon, and Prof. Srinivasa Salapaka.

This work was partially supported by NASA Langley Research Center, and Air Force Office of Scientific Research.

I am grateful to my friends Iván, Juan, Lucía, and Marina for years of encouragement. Tanya Josek, thank you for making sure I stay warm and happy during the winters in Illinois, and helping me navigate the psychology of a Ph.D. degree.

Por último, me gustaría agradecer a mis padres José y Rosa su dedicación, cariño, y apoyo incondicional. Sin vuestra constancia e insistencia no hubiese podido completar la tesis. Estoy enormemente agradecido. Carlos, gracias por ser "la alegría de la huerta", por ser un ejemplo a seguir en cuanto a filosofía y vida, y por compartir esa perspectiva con tu hermano. A mis iaios y abuelos, gracias por construir los cimientos de una familia que valora el conocimiento y las relaciones interpersonales. Os echo de menos.

Table of Contents

List of Tables

List of Figures

Abbreviations and Terminology

ATC	Air Traffic Control.
ATM	Air Traffic Management.
CAS	Control Augmentation.
CNC	Computer Numerical Control.
DCEL	Doubly-Connected Edge List.
EPA	Expanding Polytope Algorithm.
GJK	Gilbert-Johnson-Keerthi.
UB	Uniformly Bounded.
GUES	Globally Uniformly Exponentially Stable.
iISS	integral Input-to-State Stable.
ISS	Input-to-State Stable.
LiDAR	Light Detection And Ranging.
LQR	Linear Quadratic Regulator.
PE	Persistency of Excitation.
PH	Pythagorean-Hodograph.
PRM	Probabilistic RoadMap.
QoS	Quality of Service.
RRT	Rapidly-exploring Random Tree.
SAR	Search And Rescue.
SIT	Silhouette-Informed Tree.
TBO	Trajectory-Based Operations.
UAS	Unmanned Aerial System.

List of Symbols

$\mathbb{R}$	Field of real numbers.
$\mathbb{Z}$	Ring of integer numbers.
$\mathbb{I}_{\mathbf{n}}$	Identity matrix of size n.
$\mathbf{0}$	Zero matrix of appropriate dimensions.
$\mathbf{1}_{\mathbf{n}}$	Vector in $\mathbb{R}^n$ whose components are all 1.
$\dot{\boldsymbol{v}}$	Time-derivative of vector $\boldsymbol{v}$.
$\boldsymbol{v}'$	Parametric derivative of vector $\boldsymbol{v}$.
$\|\boldsymbol{v}\|$	2-norm of vector $\boldsymbol{v}$.
$\|\boldsymbol{v}\|_\infty$	∞-norm of vector $\boldsymbol{v}$.
$\boldsymbol{M}^\top$	Transpose of matrix $\boldsymbol{M}$.
$\|\boldsymbol{M}\|$	Induced 2-norm of matrix $\boldsymbol{M}$.
$\{\mathcal{F}\}$	Reference frame.
$\boldsymbol{R}_{\mathcal{F}_1}^{\mathcal{F}_2}$	Rotation matrix from reference frame $\{\mathcal{F}_1\}$ to reference frame reference frame $\{\mathcal{F}_2\}$.
$\boldsymbol{v}^{\mathcal{F}}$	Vector $\boldsymbol{v}$ resolved in the the reference frame $\{\mathcal{F}\}$.
$\mathrm{card}(\mathcal{S})$	Cardinality of set $\mathcal{S}$.
$\mathrm{spec}(\boldsymbol{M})$	Eigenspectrum of matrix $\boldsymbol{M}$.
$(\cdot)^\wedge$	The hat map.
$\lfloor\cdot\rfloor$	Floor function.
$\lceil\cdot\rceil$	Ceiling function.

Chapter 1

Introduction

This chapter presents a mission scenario that motivates the work in this thesis, provides an overview of the background and related work, outlines the structure of the thesis, and summarizes the contributions.

1.1 Time-Critical Coordination Frameworks

As the capabilities to sense, avoid, and plan in complex scenarios advance, autonomous robotic systems transcend the boundaries of industrial, manufacturing and research environments to become increasingly prevalent in everyday life. Indeed, the decreasing costs and growing interest in autonomous systems has spurred the design of cooperative frameworks for complex multi-agent missions. However, current algorithms lack the ability to enforce the range of coordination, temporal, and spatial constraints that a team of human operators would intuitively use to control multi-agent systems. Despite this, the advent of cooperative collision-free trajectory generation algorithms [1, 2], coupled with synchronized path-following protocols [3, 4], marked a milestone for the safe execution of cooperative missions. To narrow the gap between human operators and current cooperative motion planning and control algorithms, this thesis focuses on two thrusts:

i) the generation of safe trajectories for a cooperating fleet of Unmanned Aerial Systems (UASs) through cluttered environments; and

ii) the consensus protocols that enable online synchronization of the path-following vehicles for a variety of coordination and temporal constraints.

Several cooperative frameworks have been proposed to tackle motion control of a fleet of UASs. The framework in [5] uses temporal logic for mission planning, execution, and monitoring in the context

of a small urban environment. The authors in [6] consider a varying network topology and different clustering techniques but neglect the underlying algorithms responsible for executing autonomous flight. The research in [7] uses a vision-based Bayesian approach to cooperatively search for a target in an environment where the UASs fly above all obstacles. The work in [8] includes a cooperative framework for aerial and surface vehicles embedded in an environment without obstacles and a time-varying network topology. This thesis builds upon the cooperative framework in [4] to broaden the spectrum of temporal and coordination constraints, considers the uncertainty associated with environmental obstacles and path-following algorithms with a focus on cluttered environments, and a lossy communication network that supports the exchange of information among UASs for coordination purposes.

The following section presents a motivational scenario that illustrates the type of time-critical coordination missions addressed here, and highlights the gaps in prior coordination frameworks that this thesis attempts to tackle.

1.2 Motivational Mission Scenario

This section introduces a Search And Rescue (SAR) scenario that highlights the types of spatial, temporal, and coordination constraints that the cooperative framework shall capture to successfully execute a mission. Figure 1.1 shows a mountainous area where a hiker has disappeared. The search team is tasked with exploring the area to gather information and ultimately locate the missing person. The search team is equipped with a set of UASs distributed in two airfields, located beyond the top-left and bottom-right sections of Figure 1.1. Fleets A and B are composed of three UASs scheduled to take off and land at one of the airfields, within a temporal window assigned by the local Air Traffic Control (ATC) authority. Previous work on cooperative motion planning can design trajectories that arrive at the desired location within a temporal window. However, existing coordination algorithms do not observe these arrival windows online, which are common in ATC.

Initially, the members of each fleet will fly in close proximity to each other and will coordinate tightly with its close-proximity neighbors. Meanwhile, fleet A and fleet B will engage in mid-to-far proximity operations. Hence, members of separate fleets need not coordinate accurately with each other until they approach the area of interest for SAR. Again, prior time-critical coordination

algorithms do not conceive a loose coordination behavior, where vehicles are allowed bounded coordination errors.

When the vehicles arrive at the point of interest for SAR, both fleets must start a joint effort to locate the hiker and interact in close proximity. Each vehicle will leverage global information acquired during the planning phase of the mission—terrain maps, hiking routes, and nearby shelter locations—with local information coming from sensors—visible range and infrared cameras, LiDAR, or altimeters—to find signs of a lost hiker. If at least one UAS finds evidence that may lead to the missing person, the fleets may alter their flight plan to further investigate the area and provide detailed information to the rescue team. In this process, the UASs are likely to fly through unstructured cluttered environments such as a cave or underneath the tree canopy. The obstacle information in these scenarios may come from different sources—a database, onboard sensors, or even other UASs—with different levels of uncertainty. Uncertain and cluttered scenarios are still a challenge for motion-planning algorithms.

The search phase will terminate either when the missing hiker has been found or due to mission constraints—limited range, flight time, lossy communication links, or impassable narrow corridors. Then, both fleets will initiate the return to the corresponding airfield and will land within the temporal windows assigned by the local ATC authority.

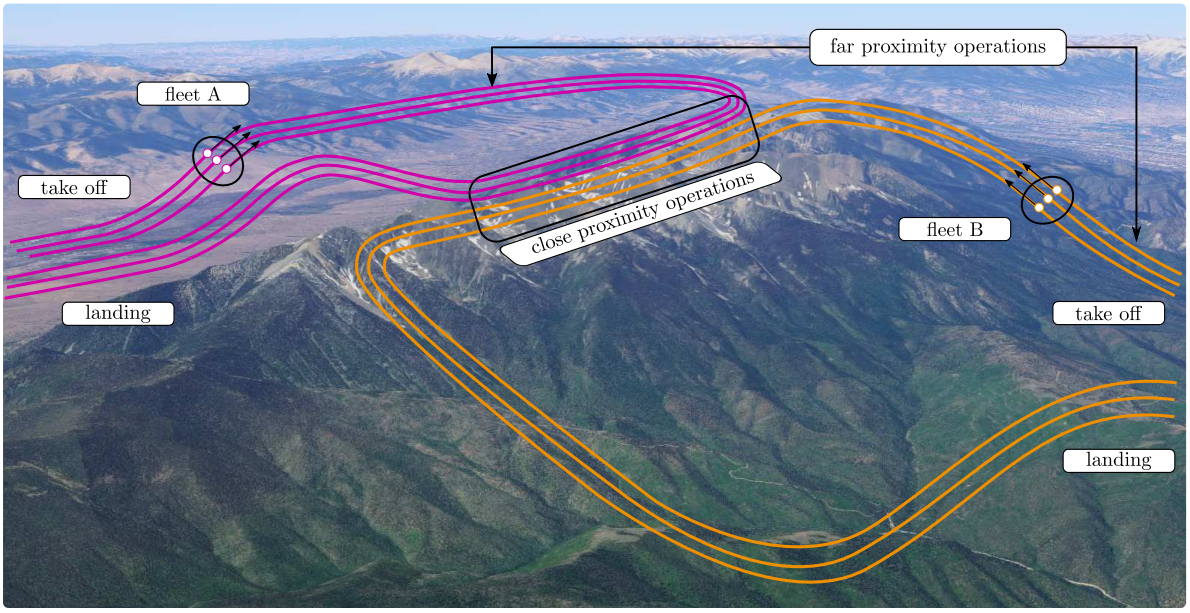


Figure 1.1: Cooperative SAR scenario with two coordinating fleets of UASs.

1.3 Related work

1.3.1 Path Planning in Cluttered Scenarios

In the context of trajectory generation, rapidly-exploring random trees (RRT) were developed as a single-query method that can quickly expand through the configuration space [9]. More recently, a modified version of the algorithm with asymptotic optimality guarantees, RRT* was proposed [10]. However, these methods tend to require large numbers of samples to identify paths through narrow corridors. Also, convergence to the optimal solution in [10] requires exhaustive sampling of the configuration space. As a result, numerous efforts have pursued the derivation of heuristic techniques that reduce the number of samples without degrading the quality of the path produced. The work in [11] utilized potential functions to guide the sampling towards the regions of interest, reduce memory utilization, and increase the convergence rate towards the optimal solution. Researchers in [12] combined RRT* with a Linear Quadratic Regulator (LQR) for the linearized dynamics of an under-actuated system. In this case, the LQR was used to automatically define a domain-specific distance method and the corresponding node extension mechanism. The authors in [13] guided the growth of the tree towards lower-cost regions in the configuration space through biased sampling. The work in [14] introduced two methods: *i*) local biasing to improve convergence to the optimal solution, only activated after a solution is found, and *ii*) node rejection to reduce the number of nodes in the tree and improve efficiency. In this case, the algorithm rejects nodes that cannot improve the cost of the solution. Finally, they combined these heuristics with RRT* [10] and RRT-connect [15]. Certainly, a significant part of the heuristic efforts in rapidly-exploring random trees focuses on improving the rate of convergence to the optimal solution. On the other hand, the work in [16] increased the probability of sampling in a narrow corridor by borrowing tools initially developed for Probabilistic RoadMaps (PRM) [17]. The authors in [18] suggested the use of a penetration-depth algorithm to sample on the boundary of the obstacle-free region of the configuration space, while the algorithm in [19] analyzes the topology of the configuration space and creates a graph through the obstacle-free region. This graph is then leveraged to sample near the regions of interest from a topological viewpoint.

1.3.2 Time-Critical Coordination

Inspired by clock synchronization and distributed consensus algorithms such as [20, 21, 22], the authors in [23, 24] propose a distributed control framework that addresses online vehicle *coordination*. In this context, coordination implies that agents meet desired relative position constraints such as simultaneous or sequential arrivals, specific formation patterns, or simply safe separation among vehicles, depending on the design of the 4D trajectories. To achieve this, the fleet must reach agreement on some distributed time-variables of interest, and at the same time, ensure that these variables evolve at a desired rate, which sets the pace of the mission. These time-variables are referred to as the coordination states. The work in [23, 24] focuses on coordination and does not address absolute temporal constraints, which require vehicles to be at a particular point at a specific time.

The concept of tight and loose sequencing in [25] shares similarities with the temporal constraints introduced in the SAR mission scenario in Section 1.2. However, [25] applies to the trajectory-planning phase, and does not extend to online coordination. The PI protocol in [23] presents a leader-follower agent hierarchy that is also used within this thesis. However, [23] does not include absolute temporal constraints. Later, [26] extends the guarantees in [23] for the exchange of quantized data, and analyzes its response to a constant disturbance. The work in [27] heavily relies on results from adaptive control and persistence of excitation in [28], which leads to tighter performance bounds. However, it lacks the leader-follower agent hierarchy and the notion of a reference clock time. The authors in [29] propose a proportional protocol with an event-triggered approach for two types of trigger functions. This thesis borrows some this event-triggered approach to derive coordination protocols that observe arrival windows. However, the work here uses a broader class of trigger functions. An overview of the spectrum of coordination and temporal constraints and their applications is presented in [30] without proven guarantees.

1.4 Thesis Overview and Contributions

This thesis is organized in seven main chapters, not including this introductory chapter, and three appendices containing algorithms, proofs, and intermediate results. This section presents a brief

summary of each chapter and the contributions contained within.

- **Chapter 2** describes in detail the cooperative framework used in this thesis, formulates the trajectory generation and time-critical coordination problems, and defines novel time-critical coordination strategies. This chapter also introduces a set of assumptions on the network that supports communications among vehicles. Finally, the applicability of the proposed time-critical coordination strategies is discussed for different real mission scenarios.

- **Chapter 3** proposes a geometric model to represent obstacles and vehicle motions that explicitely considers the uncertainty associated with these. Obstacles are represented by convex polytopes, and the movement of the UASs is represented by piecewise Bézier curves. The chapter describes the proximity queries used to determine the relative position between the different objects in the geometric model, and derives two algorithms to perform tolerance verification queries between a polytope and a Bézier curve, as well as a pair of Bézier curves. These tolerance verification queries avoid the expensive step of explicitly computing buffer regions around the geometric objects to account for the uncertainty in their location and geometric description. Moreover, this chapter introduces the silhouette of an obstacle as a geometric query that can provide useful information to motion-planning algorithms in cluttered environments. Two novel algorithms for the computation of the silhouette of a polytope with different boundary representations are provided.

- **Chapter 4** introduces a new sampled-based motion-planning algorithm that uses the silhouette to guide the growth of a random tree through the narrow passages that occur in cluttered scenarios. The result is a sequence of line segments that connect the initial and final positions of all the UASs in the cooperating fleet. This initial path is processed until $\mathcal{G}^2$ continuity is achieved. To this end, an edge-reduction method is used to reduce the number of line segments that connect the initial and final positions. This is formulated as a linear programming problem. Then, the reduced paths are smoothed using a CNC-inspired corner rounding algorithm that guarantees the resulting Pythagorean-Hodograph (PH) Bézier curves maintain a safe separation with all the obstacles in the environment. Finally, a centralized speed-assignment algorithm is used to define speed profiles for all the cooperating

agents, ensuring safe separation among vehicles. The resulting trajectories are piecewise $\mathcal{C}^2$ continuous curves that satisfy boundary conditions, meet safe separation constraints with obstacles and peers, and meet a set of simplified dynamic constraints. The efficacy of the trajectory-generation solution is illustrated with the computation of trajectories for a group of eight UASs through an urban-like cluttered environment.

- **Chapter 5** proposes a set of distributed protocols for tight coordination under a spectrum of temporal constraints. The chapter uses algebraic graph, Lyapunov, switched systems, and finite state machine theory to prove that the tight coordination protocols solve the coordination control problem when the vehicles implement an ideal velocity-tracking controller. Transient and steady-state performance bounds are derived as a function of the Quality of Service (QoS) of the communication network. Finally, simulation results are used to corroborate the conclusions derived from the different theorems in the chapter.

- **Chapter 6** considers realistic and heterogeneous velocity-tracking controllers for the group of UASs, and extends the theoretical results in Chaper 5 using perturbation theory. Again, transient and steady-state performance bounds are derived. The resulting theorems indicate that the protocols drive the coordination dynamics to a neighborhood of the coordination control objective. The theoretical results are confirmed through a simulated mission in a cluttered urban-like environment, where the vehicles are subject to wind gusts and the entire spectrum of temporal constraints defined within this thesis.

- **Chapter 7** defines novel distributed protocols for loose coordination constraints. The proposed solution requires the UASs to exchange an additional variable with their neighboring peers. This increases the network traffic, but also presents certain advantages in the performance of the loose coordination algorithms. The same simulation as in Chapter 6 is used to test and evaluate the system under loose coordination constraints, and infer several conclusions by comparison with tight coordination.

- **Chapter 8** closes the thesis with concluding remarks, and possible future extensions to the work presented in this thesis.

Chapter 2

Cooperative Framework

This chapter describes the cooperative framework that this thesis builds upon to broaden the spectrum of coordination strategies for a fleet of UASs operating in cluttered environments. The chapter focuses on three areas of interest within the framework: trajectory generation in cluttered environments, the path-following algorithm, and time-critical coordination. It provides a formal definition of the problems associated with these three thrusts, and presents the assumptions considered in the development of solutions for these problems. Finally, the chapter discusses the applicability of the proposed coordination strategies in real mission scenarios.

2.1 Framework Overview

Figure 2.1 shows the algorithms that would participate in the cooperative effort required to execute the mission described in Section 1.2. The framework and algorithms within are founded on the idea of *decoupling space and time*, both in the planning and execution phases of the mission. This concept was first suggested in [31] and has been expanded in subsequent research [4, 23, 32, 33, 2, 34].

Ideally, each UAS implements onboard the algorithms confined within the dashed block in Figure 2.1. The green boxes indicate that vehicles can exchange information with their cooperating peers and a *supervisory control authority* through a wireless communication network. The supervisory control authority can either be a machine or a human, and is typically located at a ground station. It is responsible for generating a mission description that contains: the vehicles involved and their cooperative assignments, a set of mission objectives to be achieved by the fleet, and any mission-specific constraints that the UASs must abide by. In addition, the supervisory control authority has full decision power over the evolution of the mission. It can speed up or slow down the pace of the mission, modify the vehicle assignments, and even terminate all operations.

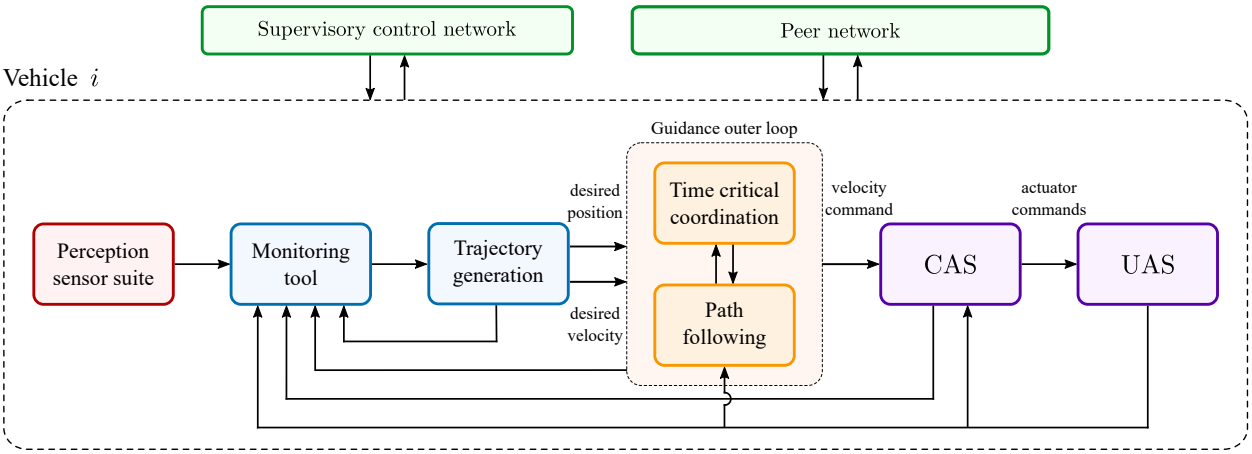


Figure 2.1: Architecture of the cooperative framework.

The perception sensor suite block—colored in red—encompasses all the sensors and algorithms used to merge the different streams of information coming from cameras, LiDAR, altimeters, and inertial measuring units, to name a few examples. The onboard monitoring tool—highlighted in blue—gathers information from the different subsystems within the UAS to check vehicle health and mission safety. The results of this analysis are used to trigger different events such as an emergency landing or a trajectory recomputation, if the mission does not evolve as initially planned.

The trajectory generation block—shaded in blue—ingests the mission description, and a partial or full representation of the environment the vehicles will fly through. Then, leveraging the idea of decoupling space and time, it produces a set of parameterized spatial paths together with a set of speed profiles for each of the UASs involved in the cooperative mission. The underlying algorithm considers the mission objectives and constraints, simplified vehicle dynamics, safe separation rules among obstacles and vehicles, an optimization criterion, and a fixed computational budget to produce feasible trajectories.

The guidance outer loop—colored in orange—is composed of two interdependent algorithms. The path-following controller steers the vehicles towards their assigned paths, and leverages the structural benefits of path-following approaches detailed in [35], as opposed to trajectory-tracking algorithms. On the other hand, the time-critical coordination law is tasked with the synchronization of the vehicles along their paths, and relies on the wireless communication network for the exchange of information among the cooperating agents. The interaction between these algorithms is designed to ensure that vehicles satisfy their relative position constraints as designed by the tra-

jectory generation algorithm, and meet their flyover and arrival times with a prespecified accuracy. The result is a set of higher-level commands that are passed to the Control Augmentation System (CAS)—shaded in purple. While the guidance outer loop governs the vehicle kinematics, the CAS generates actuator-level commands that stabilize the vehicle dynamics and track the outputs of the guidance outer loop.

Two design principles in the architecture detailed in Figure 2.1 were key in the development of new coordination strategies. The cascaded structure and the concept of decoupling space and time helped decompose a complex non-linear networked problem into various more manageable problems that apply to a large class of vehicles [4]. This allowed the introduction of further complexity in the time-critical coordination law with the intention of boosting the capabilities and flexibility of the system. This thesis focuses on the trajectory-generation, and time-critical coordination blocks. The remaining algorithms are beyond the scope of this document, and are included in this framework overview as contextual information.

2.2 Trajectory Generation

The concept of Trajectory Based Operations (TBO) has its roots in Air Traffic Management (ATM) and models each aircraft as a point that moves over time along a curve in space [36]. The term *trajectory* denotes that these spatial curves specify the position of the vehicles at all points in time. Conversely, the term *path* is used when the curves that represent the movement of the aircraft do not contain temporal assignments. This thesis adopts a TBO framework with the objective of designing n feasible trajectories $\boldsymbol{p_{d,i}}(t_d)$ with $i \in \mathcal{I}$ through a configuration space $\mathcal{X} \subset \mathbb{R}^3$ cluttered with n_o obstacles $\mathcal{O}_k \subset \mathcal{X}$ with $k \in \mathcal{K}$

$$\boldsymbol{p_{d,i}} : [t_{d_{init},i}, t_{d_{end},i}] \to \mathcal{X}, \qquad \forall i \in \mathcal{I},$$

where $t_d \in [t_{d_{init}}, t_{d_{end}}]$ is the *desired mission time*, $t_{d_{init}} := \min_{i\in\mathcal{I}} t_{d_{init},i}$ is the initial mission time, and $t_{d_{end}} := \max_{i\in\mathcal{I}} t_{d_{end},i}$ is the final mission time, with $\mathcal{I} := \{1, \ldots, n\}$ and $\mathcal{K} := \{1, \ldots, n_o\}$ being the set of unique vehicle and obstacle identification numbers, respectively. The desired mission time t_d is the time variable used in the trajectory generation phase. However, the actual mission time can evolve at a different pace, as discussed in Section 2.4.

Leveraging the idea of decoupling space and time, each trajectory $\boldsymbol{p_{d,i}}(t_d)$ is separated into a spatial and a temporal component. The path $\boldsymbol{p_{d,i}}(\zeta_i)$ is a regular[1] curve expressed as a function of a dimensionless parameter ζ_i, and defines the spatial component of the trajectory. The curve parameter ζ_i can be defined over an arbitrary interval. For simplicity, we choose $\zeta_i \in [0, 1]$, which yields the following map:

$$\boldsymbol{p_{d,i}} : [0, 1] \to \mathcal{X}, \qquad \forall i \in \mathcal{I}.$$

On the other hand, the temporal element of the trajectory is prescribed by the desired speed profile

$$v_{d,i}(t_d) := \frac{\mathrm{d}\ell_{d,i}(t_d)}{\mathrm{d}t_d}, \qquad t_d \in \left[t_{d_{init},i}, t_{d_{end},i}\right], \qquad \forall i \in \mathcal{I},$$

where $\ell_{d,i}(t_d)$ represents the *desired arclength* traveled by the ith vehicle from time $t_{d_{init},i}$ to time t_d. Given a path $\boldsymbol{p_{d,i}}(\zeta_i)$ and a speed profile $v_{d,i}(t_d)$, the trajectory $\boldsymbol{p_{d,i}}(t_d)$ can be derived by finding the relation between ζ_i and t_d. To this end, we use the implicit expression

$$\int_0^{\zeta_i} \sigma_i(z)\,\mathrm{d}z = \int_{t_{d_{init},i}}^{t_d} v_{d,i}(\tau)\,\mathrm{d}\tau, \tag{2.1}$$

where $\sigma_i := \mathrm{d}\ell_{d,i}/\mathrm{d}\zeta_i$ is the *parametric speed*, which can be calculated as follows:

$$\sigma_i(\zeta_i) = \sqrt{\frac{\mathrm{d}\boldsymbol{p}_{\boldsymbol{d,i}}^\top(\zeta_i)}{\mathrm{d}\zeta_i}\frac{\mathrm{d}\boldsymbol{p_{d,i}}(\zeta_i)}{\mathrm{d}\zeta_i}}.$$

Note that Equation (2.1) may not yield a closed-form solution for $\zeta_i(t_d)$. However, the utilization of PH curves [37] can greatly simplify the computation of the left-hand side of this equation. Recent developments in cooperative trajectory generation [2] tackle the design of the temporal component of the trajectory in a different manner. First, a *timing law* θ_i that characterizes the evolution of the curve parameter ζ_i over time t_d is designed

$$\theta_i(t_d) := \frac{\mathrm{d}\zeta_i}{\mathrm{d}t_d}, \qquad t_d \in \left[t_{d_{init},i}, t_{d_{end},i}\right], \qquad \forall i \in \mathcal{I},$$

such that it yields a closed-form solution for $\zeta_i(t_d) = \int_{t_{d_{init},i}}^{t_d} \theta_i(\tau)\,\mathrm{d}\tau$. As a result, the speed profile

[1] A parametric curve $\boldsymbol{p_d}(\zeta) : \mathcal{Q} \to \mathbb{R}^m$ defined over an arbitrary interval $\mathcal{Q}$ is regular if $\|\boldsymbol{p_d}'(\zeta)\| \neq 0$ for all $\zeta \in \mathcal{Q}$, where $\boldsymbol{p_d}'(\zeta) := \mathrm{d}\boldsymbol{p_d}(\zeta)/\mathrm{d}\zeta$.

$v_{d,i}(t_d)$ is designed indirectly, and computed through the following expression:

$$v_{d,i}(t_d) = \frac{\mathrm{d}\ell_{d,i}}{\mathrm{d}\zeta_i}\frac{\mathrm{d}\zeta_i}{\mathrm{d}t_d} = \sigma_i(\zeta_i(t_d))\,\theta_i(t_d), \qquad \forall i \in \mathcal{I}. \tag{2.2}$$

This method, further detailed in [2], can efficiently generate n feasible trajectories, and yields a convenient reparameterization of the paths as a function of the desired time t_d. However, the control over the shape of the speed profile, as shown in Equation (2.2), is often limited and heavily influenced by the parametric speed. This couples the spatial and temporal degrees of freedom within the trajectory generation phase. In cluttered scenarios, this approach has difficulty deconflicting a group of UASs that is tasked to pass through the same narrow passage, due to the lack of control over the shape of $v_{d,i}(t_d)$. For this reason, the proposed approach directly designs the speed profile $v_{d,i}(t_d)$ to fully control the temporal degrees of freedom, at the expense of solving a more challenging reparameterization problem, as detailed in Equation (2.1).

Moreover, in the presence of a few environmental hazards, a single parametric curve may be sufficient to steer a vehicle from its initial location to its assigned goal avoiding all obstacles. However, in a cluttered environment such curve will likely not have enough degrees of freedom to achieve this objective. To circumvent this problem, trajectories are defined piecewise as a sequence of trajectory segments subject to certain smoothness conditions

$$\boldsymbol{p_{d,i}}(t_d) = \boldsymbol{p^j_{d,i}}(t_d), \qquad \forall t_d \in \left[t^j_{d_{init},i}, t^j_{d_{end},i}\right], \qquad \forall j \in \mathcal{J}_i, \qquad i \in \mathcal{I},$$

where $\boldsymbol{p^j_{d,i}}(t_d)$ denotes the jth trajectory segment associated with the ith vehicle, $\mathcal{J}_i := \{1, \ldots, n_{p,i}\}$, and $n_{p,i}$ is the total number of trajectory segments assigned to the ith UAS. Similarly, the piecewise definition of this trajectory yields $n_{p,i}$ path segments $\boldsymbol{p^j_{d,i}}(\zeta^j_i)$ with dimensionless parameters $\zeta^j_i \in [0, 1]$, and $n_{p,i}$ speed profile segments $v^j_{d,i}(t_d)$. The following subsection addresses the constraints that these trajectory segments must satisfy to achieve the goal of the trajectory generation problem.

2.2.1 Trajectory Constraints

The types of constraints considered within this section encompass boundary conditions, geometric and parametric continuity conditions, safe separation constraints, and a few simple dynamic constraints.

Boundary Conditions

The trajectory generation phase is initiated by the supervisory control authority, who provides initial and final conditions for each of the cooperating agents. For the purposes of this thesis, assume that the desired position, speed, and acceleration values at the start and end of the missions have been specified

$$\begin{aligned} &\boldsymbol{p_{d,i}}(\zeta_i = 0) = \boldsymbol{p_{d_{init},i}}, \qquad v_{d,i}(t_d = t_{d_{init},i}) = v_{d_{init},i}, \qquad a_{d,i}(t_d = t_{d_{init},i}) = a_{d_{init},i}, \\ &\boldsymbol{p_{d,i}}(\zeta_i = 1) = \boldsymbol{p_{d_{end},i}}, \qquad v_{d,i}(t_d = t_{d_{end},i}) = v_{d_{end},i} \qquad a_{d,i}(t_d = t_{d_{end},i}) = a_{d_{end},i}, \end{aligned} \tag{2.3}$$

for all $i \in \mathcal{I}$, with $a_{d,i} := \mathrm{d}^2\ell_{d,i}/\mathrm{d}t_d^2$. In other instances, the mission manager may also specify different boundary constraints such as intermediate points of interest to be visited by some of the UASs, desired initial and final directions for the paths, simultaneous arrivals $t_{d_{end},i} = t_{d_{end},j}$, or even sequential arrivals $t_{d_{end},i} = t_{d_{end},j} + \Delta_{a_{i,j}}$ for some $i, j \in \mathcal{I}$, where $\Delta_{a_{i,j}}$ denotes the temporal separation at arrival between agents i and j.

$\mathcal{G}^2$ and $\mathcal{C}^2$ Continuity Constraints

The imposition of certain smoothness conditions on the trajectory has two objectives. First, the underlying algorithms that consume the trajectory—and possibly its first and higher-order derivatives—receive a set of signals with known continuity properties. This can be useful in the derivation of guaranteed performance bounds for some of the underlying algorithms such as the CAS. Second, smooth curves appear more natural to human operators, who ultimately have the responsibility to approve or modify the mission plan proposed by the trajectory generation algorithm. In this work trajectories are required to be $\mathcal{C}^2$ *continuous*, that is twice continuously differentiable with respect to the desired mission time t_d. This enforces the following constraints

at the contact points of the different trajectory segments:

$$\boldsymbol{p_{d,i}^{j}}(t_{d_{end},i}^{j}) = \boldsymbol{p_{d,i}^{j+1}}(t_{d_{init},i}^{j+1}), \quad \boldsymbol{v_{d,i}^{j}}(t_{d_{end},i}^{j}) = \boldsymbol{v_{d,i}^{j+1}}(t_{d_{init},i}^{j+1}), \quad \boldsymbol{a_{d,i}^{j}}(t_{d_{end},i}^{j}) = \boldsymbol{a_{d,i}^{j+1}}(t_{d_{init},i}^{j+1}), \tag{2.4}$$

for all $i \in \mathcal{I}$ with $n_{p,i} > 1$ and all $j \in \{1, \ldots, n_{p,i} - 1\}$, where $\boldsymbol{v_{d,i}^{j}}(t_d)$ and $\boldsymbol{a_{d,i}^{j}}(t_d)$ represent the velocity and acceleration vectors

$$\boldsymbol{v_{d,i}^{j}} := \frac{\mathrm{d}\boldsymbol{p_{d,i}^{j}}}{\mathrm{d}\zeta_i^j}\frac{\mathrm{d}\zeta_i^j}{\mathrm{d}t_d}, \qquad \boldsymbol{a_{d,i}^{j}} := \frac{\mathrm{d}^2\boldsymbol{p_{d,i}^{j}}}{\mathrm{d}\zeta_i^{j\,2}}\left(\frac{\mathrm{d}\zeta_i^j}{\mathrm{d}t_d}\right)^2 + \frac{\mathrm{d}\boldsymbol{p_{d,i}^{j}}}{\mathrm{d}\zeta_i^j}\frac{\mathrm{d}^2\zeta_i^j}{\mathrm{d}t_d^2}.$$

Again, leveraging the idea of decoupling space and time, the parametric continuity constraints above can be rewritten as a set of geometric continuity constraints on the paths $\boldsymbol{p_{d,i}^{j}}(\zeta_i^j)$ and $\boldsymbol{p_{d,i}^{j+1}}(\zeta_i^{j+1})$, together with a set of constraints on the evolution of the curve parameters $\zeta_i^j(t_d)$ and $\zeta_i^{j+1}(t_d)$. First, the paths are required to be $\mathcal{G}^2$ *continuous*, that is

$$\frac{\mathrm{d}\boldsymbol{p_{d,i}^{j+1}}}{\mathrm{d}\zeta_i^{j+1}}(\zeta_i^{j+1} = 0) = \beta_1\frac{\mathrm{d}\boldsymbol{p_{d,i}^{j}}}{\mathrm{d}\zeta_i^j}(\zeta_i^j = 1), \quad \frac{\mathrm{d}^2\boldsymbol{p_{d,i}^{j+1}}}{\mathrm{d}\zeta_i^{j+1\,2}}(\zeta_i^{j+1} = 0) = \beta_1^2\frac{\mathrm{d}^2\boldsymbol{p_{d,i}^{j}}}{\mathrm{d}\zeta_i^{j\,2}}(\zeta_i^j = 1) + \beta_2\frac{\mathrm{d}\boldsymbol{p_{d,i}^{j}}}{\mathrm{d}\zeta_i^j}(\zeta_i^j = 1),$$

for some parameters $\beta_1 > 0$ and an arbitrary β_2, see the beta-constraint characterization of geometric continuity in [38]. As a result, the condition on the temporal evolution of the curve parameters induced by the continuity constraint on the velocity vector is

$$\frac{\mathrm{d}\zeta_i^{j+1}}{\mathrm{d}t_d}(\zeta_i^{j+1} = 0) = \frac{1}{\beta_1}\frac{\mathrm{d}\zeta_i^j}{\mathrm{d}t_d}(\zeta_i^j = 1).$$

On the other hand, the continuity condition on the acceleration vector yields

$$\frac{\mathrm{d}\zeta_i^{j+1\,2}}{\mathrm{d}t_d^2}(\zeta_i^{j+1} = 0) = \frac{1}{\beta_1}\left(\frac{\mathrm{d}^2\zeta_i^j}{\mathrm{d}t_d^2}(\zeta_i^j = 1) - \frac{\beta_2}{\beta_1^2}\left(\frac{\mathrm{d}\zeta_i^j}{\mathrm{d}t_d}\right)^2(\zeta_i^j = 1)\right).$$

Higher-order continuity constraints could be imposed following a similar procedure. However, here we considered that the complexity introduced in the structure of the parametric curves $\boldsymbol{p_{d,i}^{j}}(\zeta_i^j)$, and the increased number of conditions at the end points of $\boldsymbol{p_{d,i}^{j}}(\zeta_i^j)$ and $\zeta_i^j(t_d)$ did not justify the benefits of higher order continuity constraints.

Safe Separation Constraints

To ensure that vehicle frames—not just the points that represent the vehicles in the TBO framework—maintain a safe separation with obstacles, each pair $(\boldsymbol{p_{d,i}}, \mathcal{O}_k)$ is assigned a safety distance $d_s(\boldsymbol{p_{d,i}}, \mathcal{O}_k)$. Similarly, to guarantee that cooperating peers avoid collisions among themselves each pair $(\boldsymbol{p_{d,i}}, \boldsymbol{p_{d,j}})$ is assigned a safety distance $d_s(\boldsymbol{p_{d,i}}, \boldsymbol{p_{d,j}})$ with $i \neq j \in \mathcal{I}$. Then, we define the *unsafe configuration space* for the ith vehicle as the union of two sets

$$\mathcal{X}_{u,i}(t_d) := \mathcal{X}_{u_o,i} \cup \mathcal{X}_{u_p,i}(t_d),$$

where $\mathcal{X}_{u_o,i}$ and $\mathcal{X}_{u_p,i}$ are the *obstacle-induced* and *peer-induced unsafe configurations spaces*, defined as follows:

$$\begin{aligned} \mathcal{X}_{u_o,i} &:= \bigcup_{k \in \mathcal{K}} \left\{ \boldsymbol{p} \in \mathcal{X} \,\middle|\, \|\boldsymbol{p} - \boldsymbol{p_o}\| \leq d_s(\mathcal{O}_k, \boldsymbol{p_{d,i}}), \quad \forall \boldsymbol{p_o} \in \mathcal{O}_k \right\}, \\ \mathcal{X}_{u_p,i}(t_d) &:= \bigcup_{\substack{j \in \mathcal{I} \\ j \neq i}} \left\{ \boldsymbol{p} \in \mathcal{X} \,\middle|\, \|\boldsymbol{p} - \boldsymbol{p_{d,j}}(t_d)\| \leq d_s(\boldsymbol{p_{d,i}}, \boldsymbol{p_{d,j}}) \right\}. \end{aligned} \tag{2.5}$$

Then, the *safe configuration space* for the ith vehicle is defined as

$$\mathcal{X}_{s,i}(t_d) := \mathcal{X} \setminus \mathcal{X}_{u,i}(t_d).$$

The following definition specifies the conditions under which the ith trajectory maintains a safe separation with all the obstacles in the cluttered environment $\mathcal{X}$, as well as all the trajectories of the cooperating peers.

Definition 1 (Deconflicted Trajectories) *A trajectory $\boldsymbol{p_{d,i}}(t_d)$ is said to be deconflicted if the image of the trajectory map lies in the safe configuration space $\mathcal{X}_{s,i}(t_d)$, that is*

$$\boldsymbol{p_{d,i}}(t_d) \in \mathcal{X}_{s,i}(t_d), \qquad \forall t_d \in [t_{d_{init},i}, t_{d_{end},i}], \qquad i \in \mathcal{I}.$$

Notice that in Equation (2.5) the obstacle-induced unsafe configuration space is static, whereas the peer-induced unsafe configuration space varies with the mission time t_d. Consequently, to ensure

that a trajectory $\boldsymbol{p_{d,i}}(t_d)$ maintains a safe separation with all the obstacles in a cluttered environment, the corresponding path $\boldsymbol{p_{d,i}}(\zeta_i)$ is designed to lie in the *obstacle-induced safe configuration space* $\mathcal{X}_{s_o,i} := \mathcal{X} \setminus \mathcal{X}_{u_o,i}$, that is

$$\boldsymbol{p_{d,i}}(\zeta_i) \in \mathcal{X}_{s_o,i}, \qquad \forall \zeta_i \in [0,1], \qquad i \in \mathcal{I}.$$

However, when it comes to ensuring safe separation among different trajectories, one can exploit the spatial and temporal degrees of freedom to deconflict them. This distinction leads to the following definitions.

Definition 2 (Spatial Separation) *Two trajectories $\boldsymbol{p_{d,i}}(t_d)$ and $\boldsymbol{p_{d,j}}(t_d)$ are said to be spatially separated if the corresponding paths $\boldsymbol{p_{d,i}}(\zeta_i)$ and $\boldsymbol{p_{d,j}}(\zeta_j)$ satisfy*

$$\|\boldsymbol{p_{d,i}}(\zeta_i) - \boldsymbol{p_{d,j}}(\zeta_j)\| > d_s(\boldsymbol{p_{d,i}}, \boldsymbol{p_{d,j}}), \qquad \forall \zeta_i, \zeta_j \in [0,1], \qquad i \neq j, \qquad i,j \in \mathcal{I}.$$

If a pair of trajectories is spatially separated and the corresponding UASs follow their assigned paths as expected, these vehicles cannot collide during the mission even if they do not observe the mission time t_d with accuracy. In cluttered scenarios, vehicles often have to fly through the same narrow passages, which makes spatial separation impractical. In these cases, the temporal degrees of freedom are leveraged to deconflict the trajectories, which leads to the following definition.

Definition 3 (Temporal Separation) *Two trajectories $\boldsymbol{p_{d,i}}(t_d)$ and $\boldsymbol{p_{d,j}}(t_d)$ are said to be temporally separated if they are not spatially separated and satisfy*

$$\|\boldsymbol{p_{d,i}}(t_d) - \boldsymbol{p_{d,j}}(t_d)\| > d_s(\boldsymbol{p_{d,i}}, \boldsymbol{p_{d,j}}), \quad \forall t_d \in [\max\{t_{d_{init},i}, t_{d_{init},j}\}, \min\{t_{d_{end},i}, t_{d_{end},j}\}], \tag{2.6}$$

with $i \neq j$, and $i,j \in \mathcal{I}$. By definition, if the time intervals of $\boldsymbol{p_{d,i}}(t_d)$ and $\boldsymbol{p_{d,j}}(t_d)$ do not overlap, the interval in Equation (2.6) *is empty, and we say that the trajectories are temporally separated.*

If two trajectories are temporally separated, it is important that the vehicles have a shared notion of the mission time. Otherwise, the trajectory maps will not enforce the safe separation constraints detailed in Equation (2.6), which could lead to mid-air collisions. Hence, time coordination is

safety critical when the fleet is assigned temporally separated trajectories. The focus on cluttered environments of this work makes temporal separation the natural choice to deconflict trajectories.

Note that the notion of temporal and spatial separation can be easily extended to dynamic obstacles, if the sets $\mathcal{O}_k$ that represent the obstacles are parameterized as a function of the desired mission time $\mathcal{O}_k(t_d)$. Nonetheless, this problem is beyond the scope of this thesis.

Dynamics Constraints

Adherence to the dynamic constraints of each vehicle ensure that the underlying CAS can track the desired commands with precision; and consequently, the UASs can follow their assigned trajectories with known accuracy. In this thesis, we adopt an extremely simplified set of dynamic constraints for multirotor vehicles

$$v_{d_{\min},i} \leq v_{d,i}(t_d) \leq v_{d_{\max},i}, \qquad a_{d_{\min},i} \leq a_{d,i}(t_d) \leq a_{d_{\max},i}, \qquad \forall t_d \in \left[t_{d_{init},i}, t_{d_{end},i}\right], \tag{2.7}$$

where $v_{d_{\min},i}$ and $v_{d_{\max},i}$ denote the minimum and maximum desired speed for the ith agent; whereas $a_{d_{\min},i}$ and $a_{d_{\max},i}$ are the minimum and maximum desired acceleration. The value of these parameters should be chosen carefully so vehicles do not constantly operate on the edge of their flight envelope, and thus

$$v_{d_{\min},i} > v_{\min,i}, \qquad v_{d_{\max},i} < v_{\max,i}, \qquad a_{d_{\min},i} > a_{\min,i}, \qquad a_{d_{\max},i} < a_{\max,i},$$

where $v_{\min,i}$, $v_{\max,i}$, $a_{\min,i}$, and $a_{\max,i}$ define the true speed and acceleration limits of the ith UAS. A better dynamic constraint model for multirotors would consider the individual minimum and maximum thrust for each rotor. Other platforms lead to a different set of dynamics constraints. For instance, minimum path curvature, maximum flight path angle, or maximum climb rate are relevant variables for a fixed-wing aircraft.

2.2.2 Problem Formulation

Inspired by the work in [31, 39, 4, 2], we aim to decouple the spatial component of the problem from the temporal assignments. As a result, we divide the trajectory generation problem into two

sub-problems: path planning and speed assignment. This approach was first utilized in [40] and lets us adjust the spatial paths and speed profiles independently. The path-planning problem is exclusively defined by the spatial specifications of the trajectory such as initial and final positions, directions, curvature, or altitude constraints; whereas the speed-assignment problem consumes the spatial paths and determines a speed profile considering the remaining constraints, as detailed next.

Definition 4 (Path-Planning Problem) *Design n piecewise regular spatial paths $\boldsymbol{p_{d,i}}(\zeta_i)$ parameterized by a dimensionless variable $\zeta_i \in [0, 1]$ that*

i) satisfy the desired spatial boundary conditions,

ii) meet desired geometric continuity constraints, and

iii) maintain safe separation with all the obstacles in the workspace, $\boldsymbol{p_{d,i}} \in \mathcal{X}_{s_o,i}$ for all $i \in \mathcal{I}$.

Note that this problem definition does not mention a cost function. Here, the focus is not necessarily finding the optimal solution, but rapidly finding a solution through a very complex scenario. This approach trades path optimality for computational efficiency. In this regard, there exist algorithms with asymptotic optimality guarantees [10, 41, 42], where the probability of finding the optimal solution approaches 1 as the algorithm increases the number of samples in the configuration space. The path-planning approach developed here is significantly influenced by some of these algorithms. Once the spatial paths have been generated, the second portion of the trajectory generation problem is initiated, and determines the temporal assignments for each UAS.

Definition 5 (Speed-Assignment Problem) *Design n piecewise speed profiles $v_{d,i}(t_d)$ parameterized by the desired mission time $t_d \in [t_{d_{init},i}, t_{d_{end},i}]$ that*

i) satisfy the desired temporal boundary constraints,

ii) ensure the associated trajectories $\boldsymbol{p_{d,i}}(t_d)$ satisfy the desired parametric continuity constraints,

iii) guarantee the vehicle dynamic constraints are not violated, and

iv) enforce temporal separation among the trajectories $\boldsymbol{p_{d,i}}(t_d)$ of the cooperating agents.

As in the path-planning problem, the speed-assignment problem does not aim to find the optimal solution to make the problem more tractable. The following section introduces a simple mechanism to follow the trajectories that result from solving the path-planning and speed-assignment problems.

2.3 Path Following

As described in [4, 43], the key idea of the path-following algorithm is to use the control effectors of the UAS to follow a *virtual target* that slides along the path. To this effect, a moving frame is attached to the virtual target, and a generalized error vector that characterizes the distance between this moving coordinate system and a frame attached to the UAS is defined, as shown in Figure 2.2. To control the movement of the virtual target along the path, we introduce a *virtual time* $\xi_i(t)$ that defines the position of the virtual target as follows:

$$\boldsymbol{p_{\mathcal{T},i}}(t) := \boldsymbol{p_{d,i}}(\xi_i(t)), \tag{2.8}$$

and define the position error

$$\boldsymbol{e_{p,i}}(t) := \boldsymbol{p_i}(t) - \boldsymbol{p_{\mathcal{T},i}}(t),$$

where $\boldsymbol{p_i}(t)$ denotes the actual position of the ith vehicle at time t. Then, the dynamics of the position error are

$$\dot{\boldsymbol{e}}_{\boldsymbol{p,i}}(t) = \boldsymbol{v_i}(t) - \boldsymbol{v_{\mathcal{T},i}}(t), \tag{2.9}$$

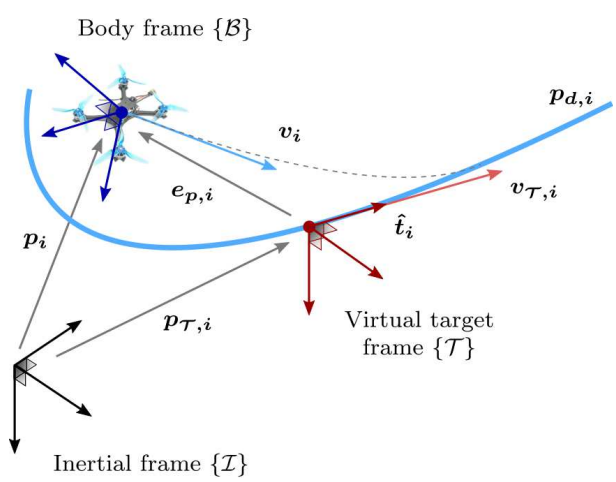


Figure 2.2: Multirotor tracking a virtual target.

where $\boldsymbol{v_i}(t)$ is the velocity vector of the UAS, and $\boldsymbol{v_{\mathcal{T},i}}(t)$ is the velocity vector of the virtual target, as depicted in Figure 2.2. The velocity of the virtual target can be expressed in terms of the desired velocity as

$$\boldsymbol{v_{\mathcal{T},i}}(t) = \dot{\xi}_i(t)\, \boldsymbol{v_{d,i}}(\xi_i(t)). \tag{2.10}$$

Now, define the velocity-tracking error

$$\boldsymbol{e_{v,i}}(t) = \boldsymbol{v_i}(t) - \boldsymbol{v_{cmd,i}}(t),$$

where $\boldsymbol{v_{cmd,i}}(t)$ is the velocity command generated by the guidance outer loop, and choose the following control law for the velocity command:

$$\boldsymbol{v_{cmd,i}}(t) := \boldsymbol{v_{\mathcal{T},i}}(t) - k_{PF,i}\, \boldsymbol{e_{p,i}}(t), \tag{2.11}$$

where $k_{PF,i} > 0$ is a control gain. Note that vehicles within the cooperative fleet can implement different gains $k_{PF,i}$. This offers the possibility of fine tuning this algorithm for each UAS if the fleet is composed of heterogeneous multirotors. The following assumption considers that the CAS can only track the velocity command $\boldsymbol{v_{cmd,i}}(t)$ with known precision $\bar{e}_{v,i}$, as long as the norm of the velocity command is bounded by the true speed limit $v_{\max,i}$.

Assumption 1 *If $\|\boldsymbol{v_{cmd,i}}(t)\| \leq v_{\max,i}$, the inner-loop velocity-tracking controller can track the velocity command $\boldsymbol{v_{cmd,i}}(t)$ with known precision $\bar{e}_{v,i}$, and thus*

$$\sup_{t \geq t_0} \|\boldsymbol{e_{v,i}}(t)\| \leq \bar{e}_{v,i}, \qquad \forall i \in \mathcal{I}.$$

The following lemma uses Assumption 1, the position error dynamics, and Lyapunov theory to prove that the origin of the system in (2.9) is Input-to-State Stable (ISS) with respect to the velocity-tracking precision.

Lemma 1 *If Assumption 1 is met, then the velocity command in Equation* (2.11) *with control gain $k_{PF,i} > 0$ ensures that the origin of the position error dynamics in Equation* (2.9) *is ISS, and the position error is bounded by*

$$\|\boldsymbol{e_{p,i}}(t)\| \leq \|\boldsymbol{e_{p_0,i}}\| e^{-k_{PF,i}(t-t_0)} + \frac{\bar{e}_{v,i}}{k_{PF,i}} \left(1 - e^{-k_{PF,i}(t-t_0)}\right), \qquad \boldsymbol{e_{p_0,i}} = \boldsymbol{e_{p,i}}(t_0).$$

Proof. See Appendix C.1. □

Remark 1 *Given $\bar{e}_{v,i}$, the ultimate bound on the position error can be made arbitrarily small by*

increasing the value of the control gain $k_{PF,i}$. However, besides the well-known drawbacks of high-gain controllers, if the velocity command is limited by $\|\boldsymbol{v_{cmd,i}}(t)\| < v_{max_i}$, then a large $k_{PF,i}$ reduces the allowable position error and mission rate that yield a feasible command

$$\|\boldsymbol{v_{cmd,i}}(t)\| \leq \dot{\xi}_i(t)\,\|\boldsymbol{v_{d,i}}(\xi_i(t))\| + k_{PF,i}\,\|\boldsymbol{e_{p,i}}(t)\| < v_{max_i}.$$

The following section addresses the synchronization of the virtual targets to ensure the vehicles have a shared notion of the mission time, and maintain safe separation constraints during mission execution when their trajectories are temporally separated.

2.4 Time-Critical Coordination

The time-critical coordination algorithm governs the movement of the virtual targets along their paths, which is leveraged—together with the path-following error—to compute a speed command, as described in Section 2.3. The objective of the time-critical coordination algorithm is to enforce coordination and temporal constraints. In this respect, previous synchronized path-following algorithms do not clearly distinguish between coordination and temporal constraints. Here, coordination refers to the agreement in the virtual times $\xi_i(t)$, and implies that agents meet desired relative position constraints such as simultaneous or sequential arrivals, specific formation patterns, or inter-agent spacing constraints as outlined in the trajectory generation phase. However, coordination constraints do not impose the time a vehicle must fly past a particular point along its path. These are temporal constraints[2]. To enforce them, a virtual entity that runs independently of all other agents is introduced, the *reference agent*. It imparts the cooperating team with the notion of a global time, and serves as a mechanism to speed up or slow down the entire mission schedule, if necessary. In pursuit of a more general architecture, only a subset of vehicles has access to the reference information—the link peers; whereas the remaining agents—the end peers—must learn that information, as depicted in Figure 2.3. Link and end peers are often referred to as leaders and followers in the literature. However, this thesis avoids that notation to highlight the information

[2]Another way to understand these constraints is to think of coordination constraints as relative temporal constraints; and what is referred here as temporal constraints can be understood as absolute temporal constraints, like a desired arrival time or an arrival window.

flow in Figure 2.3. Note that link peers serve as the nexus between the reference and the end peers, located at both extremes of the graph in Figure 2.3.

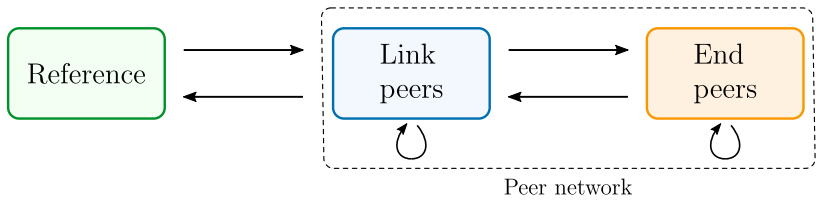


Figure 2.3: Structure of the communication network and agent hierarchy.

To achieve the goal of the time-critical coordination algorithm, the UASs exchange their virtual times $\xi_i(t)$ over the network. This provides a measure of how far along the mission a vehicle is to its neighboring agents. Then, the cooperating fleet engages in a negotiation process to reach consensus on the evolution of these virtual times. This is referred to as virtual target synchronization. As a result, if a few UASs move ahead in the mission due to favorable winds the group in advance will slow down, whereas the group that has fallen behind will increase its pace until they catch up with their neighbors. Since the virtual times $\xi_i(t)$ define the coordination mechanism, they are also referred to as the *coordination states*. The following section formally defines the time-critical coordination problem.

2.4.1 Problem Formulation

Consider a network of n *integrator agents*

$$\dot{\xi}_i(t) = u_{c,i}(t) + u_{\mathcal{T}_\vDash,i}(t), \qquad i \in \mathcal{I}\,, \tag{2.12}$$

with *dynamic information flow* $\mathcal{G}(t) := (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ represents the vertices and $\mathcal{E}(t)$ the edges of the communication graph, $\xi_i(t) \in \mathbb{R}$ is the coordination state of the ith agent, $u_{c,i}(t)$ is a coordination control input, and $u_{\mathcal{T}_\vDash,i}(t)$ is the long-track target-tracking error feedback[3]

$$u_{\mathcal{T}_\vDash,i}(t) = k_{e_p} \boldsymbol{e_{p,i}}(t) \cdot \hat{\boldsymbol{t}}_{\boldsymbol{i}}(\xi_i(t)), \tag{2.13}$$

[3]To aid in the interpretation of this nomenclature, all long-track errors include the subscript $\vDash$, as opposed to the cross-track errors that would include the subscript $\Vdash$.

where $k_{e_p} > 0$ is a control gain, $\boldsymbol{e_{p,i}}(t) \in \mathbb{R}^3$ is the position error, and $\hat{\boldsymbol{t}}_{\boldsymbol{i}}(t) \in \mathbb{R}^3$ is the unit tangent vector depicted in Figure 2.4. The target-tracking feedback is particularly relevant in cluttered scenarios. It establishes a negotiation process between each vehicle and its virtual target that mitigates *corner cutting* when the vehicle has difficulty tracking its virtual target. As shown in Figure 2.4, cutting corners in narrow passages can be safety critical. Next, to enforce temporal constraints the *reference agent* is introduced with the following dynamics:

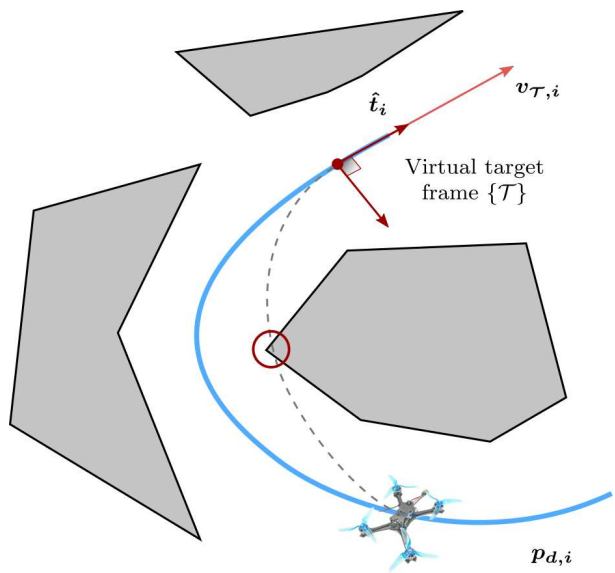


Figure 2.4: Collision caused by corner cutting.

$$\dot{\xi}_R = \rho, \qquad \xi_R(0) = \xi_{R_0}, \tag{2.14}$$

where $\xi_R(t) \in \mathbb{R}$ is the reference state, and ρ is a constant reference rate. This agent does not attempt coordination, but provides a global reference value so that, if desired, each $\xi_i(t)$ can be forced to converge to a neighborhood of $\xi_R(t)$. As shown in Figure 2.3, agents are classified by their informational needs as:

i) the *reference* who shares its state and rate with the link peers;

ii) a group of n_ℓ *link peers* that have access to the reference information, but also exchange their coordination states and rates with a set of neighboring agents; and

iii) the *end peers* that can only exchange their coordination states and rates with a set of neighboring agents.

Without loss of generatlity, the vehicle identification numbers are organized so that $\mathcal{I}_\ell := \{1, \ldots, n_\ell\}$ and $\mathcal{I}_e := \{n_\ell + 1, \ldots, n\}$ represent the set of link and end peers, respectively. These restrictions on the flow of information aim to capture a rather general scenario, where data from the reference may not be available to some agents. In this context, the control objective is to design a coordination control law that solves the following problem.

Definition 6 (Time-Critical Coordination Problem) *Design a distributed protocol that guides the coordination, temporal, and rate errors to a neighborhood of the origin*

$$\xi_i(t) - \xi_j(t) \overset{t\to\infty}{\longrightarrow} [-\Delta_c\,,\,\Delta_c]\,, \qquad \forall i,j \in \mathcal{I}, \tag{2.15a}$$

$$\xi_i(t) - \xi_R(t) \overset{t\to\infty}{\longrightarrow} [-\Delta_t\,,\,\Delta_t]\,, \qquad \forall i \in \mathcal{I}, \tag{2.15b}$$

$$\dot{\xi}_i(t) - \dot{\xi}_R \quad \in \quad [-\Delta_r\,,\,\Delta_r]\,, \qquad \forall i \in \mathcal{I}, \tag{2.15c}$$

where $\Delta_c(t)$*,* $\Delta_t(t)$*, and* $\Delta_r(t) \geq 0$ *define the width of the coordination, temporal, and rate windows, respectively.*

In the problem definition above, Equation (2.15a) defines the coordination constraints; Equation (2.15b) imposes temporal specifications, and Equation (2.15c) enforces bounds on the desired mission rate, which in turn defines bounds on the velocity command according to Equations (2.10) and (2.11). Two types of coordination constraints are defined, depending on the value of Δ_c:

i) *Tight coordination* denotes the system specifications when $\Delta_c \equiv 0$. This is typical of scenarios where accurately observing the desired inter-agent spacing constraints is safety critical, such as in close proximity operations.

ii) *Loose coordination* is used when $\Delta_c(t)$ is bounded and away from 0. In this case, $\Delta_c(t)$ defines an allowable coordination error, which is useful in mid and far-proximity operations.

Similarly, three types of temporal specifications are defined as a function of Δ_t:

i) *Unenforced temporal constraints* do not impose additional requirements, $\Delta_t \to \infty$. Thus, they are used in missions that do not require the specification of an arrival time or arrival window at any point along the path of the vehicles.

ii) *Relaxed temporal specifications* encompass all $\underline{\Delta}_t \leq \Delta_t(t) \leq \overline{\Delta}_t$ bounded and away from 0, and are used to impose a desired arrival window for one or more points along the vehicle paths.

iii) *Strict temporal constraints* force each $\xi_i(t)$ to track $\xi_R(t)$, $\Delta_t \equiv 0$, specifying a flyover time for each point along the vehicle path.

Table 2.1: Time-critical coordination strategies.

		Coordination Constraints	
		Tight	Loose
Temporal Constraints	Unenforced	$\Delta_c \equiv 0\,,\ \Delta_t \to \infty$	$\Delta_c \in (0,\infty)\ ,\ \Delta_t \to \infty$
	Relaxed	$\Delta_c \equiv 0\,,\ \Delta_t \in (0,\infty)$	$\Delta_c \in (0,\infty)\ ,\ \Delta_t \in (0,\infty)$
	Strict	$\Delta_c \equiv 0\,,\ \Delta_t \equiv 0$	$\Delta_c \in (0,\infty)\ ,\ \Delta_t \equiv 0$

The spectrum of coordination and temporal specifications introduced defines six generic time-critical coordination strategies, organized in Table 2.1 according to the values of Δ_c and Δ_t. Note also that Equation (2.15c) indicates that the coordination rate of the ith vehicle should remain within a neighborhood of ρ. Given a mission design, the following assumption constrains the acceptable values of ρ to ensure that if everything goes as planned the norm of the speed command is smaller than the true speed limit of the vehicle.

Assumption 2 *The choice of mission rate satisfies*

$$\max_{t_d \in [t_{d_{init},i}, t_{d_{end},i}]} \rho \|\boldsymbol{v_{d,i}}(t_d)\| < v_{\max,i}, \qquad \forall i \in \mathcal{I}.$$

The following section addresses the assumptions on the communication network used to develop a solution for the time-critical coordination problem.

2.4.2 Communication Network

For the sake of generality, we do not wish to impose any specific structure on the topology of the peer network, or assume any apriori knowledge about the amount of data exchanged among group members. Accordingly, the *peer communication network* satisfies the following general assumptions:

Assumption 3 *The ith peer exchanges information with a time-varying set of peers, denoted by* $\mathcal{N}_i(t) \subseteq \mathcal{I}$.

Assumption 4 *Communications are bidirectional and* $\xi_i(t)$ *is transmitted continuously and with no delays.*

Assumption 5 *The graph $\mathcal{G}(t)$ that models the communication network satisfies the Persistency of Excitation (PE)-like condition [44]*

$$\frac{1}{n}\frac{1}{T}\int_{t}^{t+T} \bar{\boldsymbol{L}}\left(\tau\right) d\tau \geq \mu\, \mathbb{I}_{\mathbf{n-1}}\,, \quad \forall\ t \geq T\,, \tag{2.16}$$

where $\bar{\boldsymbol{L}}\left(t\right) := \mathbf{QL}\left(t\right)\mathbf{Q}^{\top}$, $\boldsymbol{L} \in \mathbb{R}^{n\times n}$ is the piecewise constant Laplacian of $\mathcal{G}(t)$, and $\mathbf{Q} \in \mathbb{R}^{(n-1)\times n}$ satisfies $\mathbf{Q1_n} = \mathbf{0}$, and $\mathbf{QQ}^{\top} = \mathbb{I}_{\mathbf{n-1}}$, where $\mathbf{1_n} \in \mathbb{R}^{n}$ is a vector of all ones.

Parameters $T > 0$ and $\mu \in (0, 1]$ characterize the Quality of Service (QoS) of the network. To aid in the interpretation of Equation (2.16), define and order the eigenvalues of $\mathbf{L}\left(t\right)$ such that

$$0 \equiv \lambda_1(t) \leq \lambda_2(t) \leq \ldots \leq \lambda_n(t) \leq n.$$

Then, the eigenspectrum of $\bar{\boldsymbol{L}}\left(t\right)$ is $\text{spec}\left(\bar{\boldsymbol{L}}\left(t\right)\right) = \{\lambda_2(t), \ldots, \lambda_n(t)\}$. The Fiedler eigenvalue $\lambda_2(t)$ is a measure of the algebraic connectivity, and $\lambda_2(t) > 0$ if and only if $\mathcal{G}(t)$ is connected at time t. Consequently, μ is an integral measure of the connectivity of graph $\mathcal{G}(t)$ over a sliding temporal window of width T. Equation (2.16) captures dynamic communication topologies arising from temporary loss of communication and switching communication links. In fact, under this condition $\mathcal{G}(t)$ may even fail to be connected pointwise in time at all times during the mission.

2.4.3 Time-Critical Coordination Strategies

The set of strategies in Table 2.1 opens a wide range of possibilities to automate time criticality, and narrow the gap between human operators and motion control algorithms. Indeed, Figures 2.5 and 2.6 present real scenarios that can be automated using these strategies. Figure 2.5 focuses on applications for tight coordination, whereas Figure 2.6 presents examples for loose coordination.

The scenarios in Figures 2.5a and 2.5b require the aircraft to meet desired inter-agent spacing specifications with precision due to close proximity operations. However, while the aircraft at the airshow in Figure 2.5a may not have a temporal requirement, the tanker in Figure 2.5b has to serve multiple groups of fighters, which are often assigned a temporal slot to perform refueling. On the other hand, Figure 2.5c shows a representative example that combines tight coordination and strict temporal constraints. The figure shows the coordinated trajectories of a DC-8 aircraft

and the CryoSat-2 spacecraft [45], along with the times of the satellite passover. Calibration and validation of satellite data often requires that the satellite and aircraft measurements take place simultaneously over the same region, since the variables measured—temperature, humidity, ice thickness—may vary in space and time.

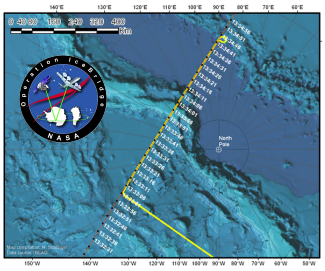
Unenforced

(a) Coordinated maneuvers at an airshow, courtesy of U.S. Navy Blue Angels

Relaxed

(b) Coordinated aerial refueling, courtesy of U.S. Air Force by Suzanne M. Jenkins

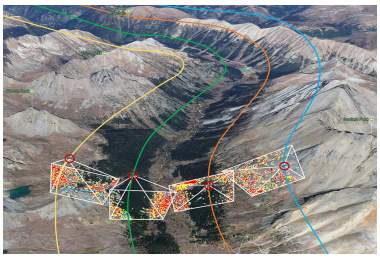
Strict



(c) Calibration of satellite data, courtesy of Michael Studinger [45]

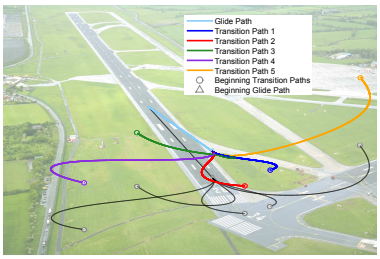
Figure 2.5: Missions with tight coordination.

Unenforced



(a) Search and rescue operation

Relaxed



(b) Auto-landing scenario

Strict

(c) Sea-air maneuvers, courtesy of U.S. Navy, adapted by Javier Puig-Navarro

Figure 2.6: Missions with loose coordination.

Figures 2.6a and 2.6b showcase a SAR mission and an auto-landing scenario, both operating in mid-proximity. In these cases, coordination is used to ensure safe operations since trajectories are not spatially deconflicted, see Figure 2.6b; and also achieve mission-specific objectives such as in Figure 2.6a, where a certain overlap in the field of view of the onboard cameras is desirable. While the aircraft participating in the SAR mission may not have any temporal requirements, ATC authorities will often assign landing slots for the aircraft in Figure 2.6b. Figure 2.7 shows the nominal arrival time $t_{d_{end},i}$, and desired arrival window for each of the vehicles in Figure 2.6b. Note that the UASs can arrive at any time within the assigned landing slot. However, the spacing of

these slots guarantees that the vehicles land at least two minutes apart. Finally, Figure 2.6c shows a fleet of vessels and aircraft engaged in coordinated maneuvers that can be implemented using loose coordination and strict temporal constraints. The vessels do not require tight coordination among themselves, since small deviations in their formation will not lead to collisions. However, they may be tasked to arrive to their destination at a particular time, which requires strict temporal constraints.

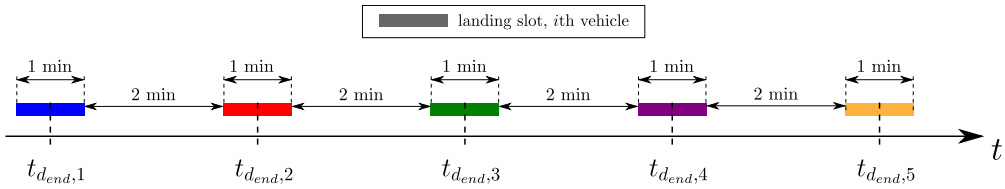


Figure 2.7: Landing slots for the auto-landing scenario.

Figures 2.5 and 2.6 highlight the flexibility that the proposed framework introduces. However, there exist missions that cannot be captured by the coordination strategies presented so far. In fact, the motivational scenario in Figure 1.1 does not fall under any of the strategies detailed in Table 2.1. Note that in Figure 1.1 fleets A and B introduce two different but coexisting coordination behaviors that are incompatible with the current problem formulation. At the start of the mission, the agents within fleet A must coordinate tightly with their fleet peers, but loosely with the agents in fleet B, and vice versa. To further expand the types of missions that can be automated with this framework, we propose the following *agent-specific coordination problem*:

$$
\begin{aligned}
\xi_i(t) - \xi_j(t) &\overset{t\to\infty}{\longrightarrow} \left[-\Delta_{c_{i,j}}\,,\, \Delta_{c_{i,j}}\right], && \forall i,j \in \mathcal{I},\\
\xi_i(t) - \xi_R(t) &\overset{t\to\infty}{\longrightarrow} \left[-\Delta_{t,i}\,,\, \Delta_{t,i}\right], && \forall i \in \mathcal{I},\\
\dot{\xi}_i(t) - \dot{\xi}_R &\in \left[-\Delta_{r,i}\,,\, \Delta_{r,i}\right], && \forall i \in \mathcal{I},
\end{aligned}
$$

where the coordination window $\Delta_{c_{i,j}}(t) \geq 0$ is defined for every pair of agents, and the temporal and rate windows $\Delta_{t,i}(t)$ and $\Delta_{r,i}(t) \geq 0$ are defined for every peer.

Chapter 3

Geometric Queries

In the context of computational geometry, proximity queries are a set of methods that return information regarding the relative distance between two or more geometric objects, such a curve and a polyhedron. Proximity queries are the backbone of path-planning algorithms, and are used to check whether candidate paths violate safe separation constraints. These queries fall within a larger category of methods called geometric queries, which report information regarding one or more objects in a topological space. This chapter introduces the geometric objects of interest for this thesis; studies and expands proximity queries for these objects; and proposes the silhouette as a novel geometric query to aid path-planning algorithms in cluttered environments, possibly with incomplete obstacle information.

3.1 Geometric Model

Consider a fleet of n UASs that must maneuver through a three-dimensional euclidean space scattered with n_o obstacles to accomplish a cooperative mission. Using lingo from the field of robotics [46], this space will be referred to as the *workspace*. To plan this mission, the different elements in the workspace are modeled as follows:

i) Environmental obstacles are represented by convex polytopes $\mathcal{O}_k \subset \mathbb{R}^3$ with $k \in \mathcal{K}$ and $\mathcal{K} := \{1, \ldots, n_o\}$. As a result, any concave obstacle must be partitioned into multiple convex polytopes[1]. Each obstacle is assigned a buffer distance $d_{o,k}$ that captures the uncertainty in its geometric description and location. This uncertainty depends on the source of information that provided the geometry of the obstacle such as a map, a LiDAR, or a stereo vision system, as well as the algorithms that processed and fused this information.

[1]The problem of partitioning a concave polyhedron into a minimum number of convex polyhedra is NP-hard [47] and, if possible, should be solved just once before addressing the trajectory planning problem.

ii) The desired position for the center of mass of the ith UAS is given by a sequence of $n_{p,i}$ Bézier curves $\boldsymbol{p_{d,i}^j}(\zeta_i^j) : [0,1] \to \mathbb{R}^3$, where ζ_i^j is the dimensionless parameter for the jth curve of the ith vehicle, with $i \in \mathcal{I}$ and $j \in \mathcal{J}_i := \{1, \ldots, n_{p,i}\}$. To avoid collisions, each path is assigned a buffer distance $d_{p,i}$ that accounts for the dimensions of the vehicle frame, and the path-following error under nominal conditions. All pairs $(\boldsymbol{p_{d,i}^j}\,,\, d_{p,i})$ define a tube through which the ith vehicle will fly if the mission evolves as planned. The *trajectory segments* $\boldsymbol{p_{d,i}^j}(t_d)$ are simply the *spatial paths* $\boldsymbol{p_{d,i}^j}(\zeta_i^j)$ expressed as a function of the *desired time* t_d. Given the *desired initial time* $t^j_{d_{init},i}$ and *desired end time* $t^j_{d_{end},i}$ for each trajectory segment, define the *normalized time*

$$\hat{t}^j_{d,i} := \frac{1}{\delta t^j_{d,i}} \left(t_d - t^j_{d_{init},i} \right), \qquad t_d \in \left[t^j_{d_{init},i}\,,\, t^j_{d_{end},i} \right],$$

where $\delta t^j_{d,i} := t^j_{d_{end},i} - t^j_{d_{init},i}$ is the *desired duration* of the jth trajectory segment assigned to the ith UAS. Then, the *desired speed profile* along each path segment

$$v^j_{d,i}(\hat{t}^j_{d,i}) := \frac{1}{\delta t^j_{d,i}} \left\| \frac{\mathrm{d}\boldsymbol{p_{d,i}^j}(\hat{t}^j_{d,i})}{\mathrm{d}\hat{t}^j_{d,i}} \right\| : [0,1] \to \mathbb{R}^+$$

is also given in the form of a Bézier curve. The desired speed profile and the *parametric speed*

$$\sigma^j_i(\zeta^j_i) := \left\| \frac{\mathrm{d}\boldsymbol{p_{d,i}^j}(\zeta^j_i)}{\mathrm{d}\zeta^j_i} \right\| : [0,1] \to \mathbb{R}^+$$

define the relationship between the curve parameter ζ^j_i and the normalized time $\hat{t}^j_{d,i}$. The objective of the trajectory generation algorithms is to design $\boldsymbol{p_{d,i}^j}(\zeta^j_i)$ and $v^j_{d,i}(\hat{t}^j_{d,i})$.

Figure 3.1 illustrates the types of geometric objects in this model. It shows a concave obstacle in dark gray, and its partition into seven convex polytopes. A safety buffer is drawn around each of these polytopes, highlighted in light gray. Each safety buffer is the convex hull of the Minkowski set sum[2] between the convex polytope $\mathcal{O}_k$ and a sphere of radius $d_{o,k}$ centered at the origin, with $k \in \mathcal{K}$. For a pair of polytopes with n_v vertices each, the Minkowski set sum returns n_v^2 points. The convex hull of n_v^2 points takes $O(n_v^2 \log n_v)$ operations [48, 49, 50]. Hence, evaluating these buffers

[2]The Minkowski set sum between two set $\mathcal{A} \subset \mathbb{R}^m$ and $\mathcal{B} \subset \mathbb{R}^m$ is defined as $\mathcal{A} \oplus \mathcal{B} := \{\boldsymbol{x} + \boldsymbol{y} \,|\, \boldsymbol{x} \in \mathcal{A},\ \boldsymbol{y} \in \mathcal{B}\}$.

at runtime is computationally expensive and must be avoided, if possible. This chapter introduces proximity queries that circumvent explicitly computing these safety buffers. Figure 3.1 also shows the paths assigned to two vehicles, and associated tubes in red and blue. The following subsections provide some insight into the decision process that lead to the proposed geometric model.

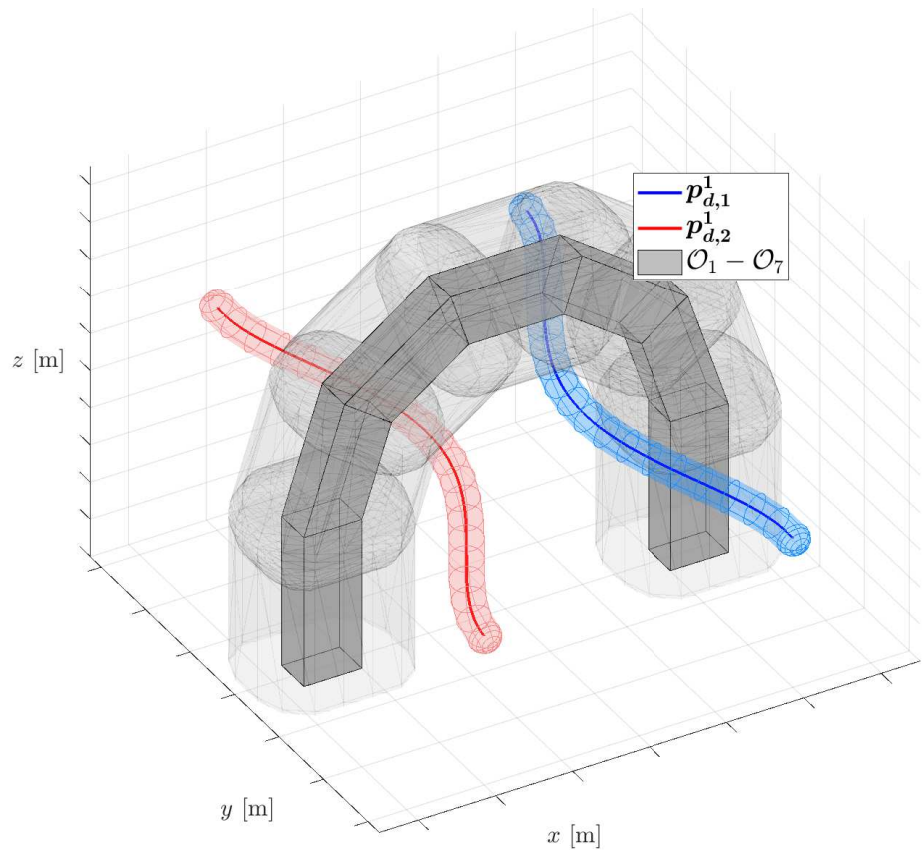


Figure 3.1: Geometric objects and uncertainty buffers.

3.1.1 Polytopes

This section provides a brief overview of polytopes, their properties, representations, and algorithms available. For simplicity, this section drops the subscript in $\mathcal{O}_k$, and uses $\mathcal{O}$ to denote a polytope.

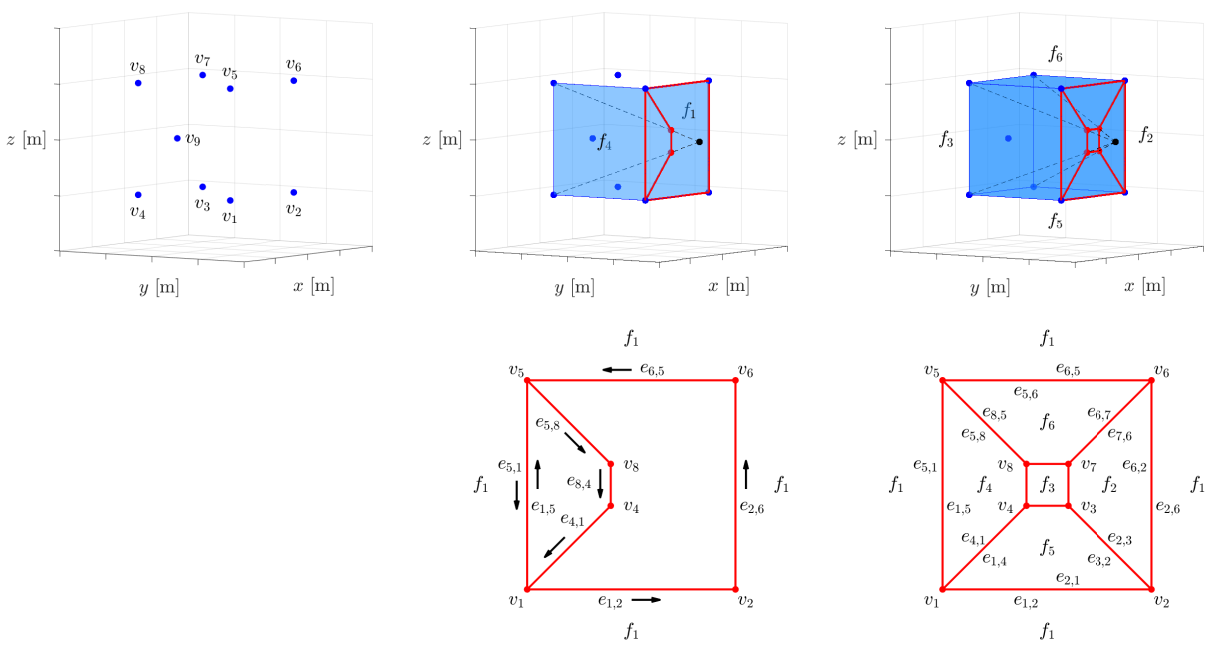
Definition 7 *Let $\mathcal{O} \subset \mathbb{R}^m$ be a polytope. Then the following statements are equivalent:*

i) $\mathcal{O}$ is the convex hull of a finite set of points,

ii) $\mathcal{O}$ is a bounded intersection of finitely many closed half-spaces.

In the field of robotics, polytopes are often given by a finite set of points, the $\mathcal{V}$*-representation*. There are three main reasons for this: first, CAD models store in memory the vertices of the objects in a workspace; second, obstacle information is sometimes collected by sensors that produce point clouds; and third, this representation is conducive to an easy interpretation and visualization of the data. The minimal $\mathcal{V}$-representation is the minimum number of vertices whose convex hull defines the polytope, and thus contains no internal vertices. In the field of combinatorial optimization, polytopes are frequently given by a set of linear inequalities that represent a family of half-spaces, the $\mathcal{H}$*-representation*. Similarly, the minimal $\mathcal{H}$-representation is the set of half-spaces that define the facets of the polytope. Given the $\mathcal{V}$-representation of a polytope, the convex hull algorithm in [51] returns the minimal $\mathcal{H}$-representation.

Steinitz's theorem proves that the boundary of a polytope $\mathcal{O} \subset \mathbb{R}^3$ can be represented by a planar graph $\mathcal{G}_o(\mathcal{V}_o, \mathcal{E}_o)$ [52]. The set of nodes $\mathcal{V}_o$ corresponds to the vertices in $\mathcal{O}$, and the set of graph edges $\mathcal{E}_o$ represents the edges in $\mathcal{O}$. An intuitive tool to construct and visualize $\mathcal{G}_o$ is a *Schlegel diagram*. Figure 3.2 illustrates the same polytope with three different boundary representations.



(a) No boundary (b) Incomplete boundary (c) Complete boundary

Figure 3.2: $\mathcal{V}$-representation and boundary information (top), Schlegel diagram with the DCEL member variables (bottom).

The bottom images show the Schlegel diagram associated with the boundary representation in the top figures. To draw a Schlegel diagram of a polytope $\mathcal{O} \subset \mathbb{R}^3$, choose a facet f_k and a point $\boldsymbol{p_v} \in \mathbb{R}^3$, such that $\boldsymbol{p_v}$ is contained in the half spaces that define all facets except f_k. Then, the projection of $\mathcal{O}$ onto f_k through rays that emanate from $\boldsymbol{p_v}$ defines the Schlegel diagram. In Figure 3.2, point $\boldsymbol{p_v}$ is shown in black, whereas the projection of $\mathcal{O}$ onto f_1 is colored in red. Note that each undirected edge in the Schlegel diagrams in Figure 3.2 is assigned two directed edges $e_{i,j}$ and $e_{j,i}$, referred to as *twin* edges. The reason for this is that the data structure chosen to represent $\mathcal{G}_o$ in this thesis is the *Doubly Connected Edge List* (DCEL), proposed in [53] with the slight modifications introduced in [54]. This data structure is designed to efficiently perform some of the underlying operations[3] in the geometric queries used here. The DCEL has three lists as member variables:

i) Vertex list: contains the coordinates of each vertex v_i, and a pointer to an arbitrary incident edge $e_{i,j} := \{v_i, v_j\}$, that has v_i as the origin.

ii) Edge list: each entry is composed of a pointer to the origin v_i of the edge $e_{i,j}$, a pointer to the twin edge $e_{j,i}$ and incident facet f_k, as well as pointers to the next and previous edges that bound f_k when traversing it counterclockwise, see Figure 3.2.

iii) Facet list: each item consists of a pointer to an arbitrary incident edge.

Additional fields may be added to each of these lists when necessary. For instance, when evaluating whether a facet is visible from a point of view $\boldsymbol{p_v} \in \mathbb{R}^3$, it is useful to store in memory the components of a unit vector normal to each facet. Tables 3.1, 3.2 and 3.3 contain the vertex, edge, and facet lists for the example in Figure 3.2b with incomplete boundary information. Note that the *nil* pointers correspond to features that are not defined, because either the boundary information is incomplete, or the vertex in question is not on the boundary of the polytope.

While using the $\mathcal{V}$ or $\mathcal{H}$-representation can make a great difference in the computational cost of certain algorithms, this thesis uses the $\mathcal{V}$-representation combined with three different levels of boundary information, depicted in Figure 3.2:

[3]Common operations that benefit from the DCEL data structure are the traversal of all the edges that are incident to a facet, and the retrieval of all the edges that are incident to a vertex.

i) In some cases, polytopes may not include a boundary representation, as shown in Figure 3.2a.

ii) In other instances, polytopes may contain an incomplete boundary representation, as in Figure 3.2b. This happens if an obstacle has been partially observed, or the algorithms responsible for generating the boundary representation have not finished execution.

iii) In the best case scenario, polytopes with a complete boundary representation are provided, see Figure 3.2c, which can reduce the computational cost of some geometric queries [55].

Table 3.1: Vertex list.

Vertex	**Coordinates**	**Incident Edge**
v_1	$[0.00, 0.00, 0.00]$	$e_{1,2}$
v_2	$[1.00, 0.00, 0.00]$	$e_{2,6}$
v_3	$[1.00, 1.00, 0.00]$	*null*
v_4	$[0.00, 1.00, 0.00]$	$e_{4,1}$
v_5	$[0.00, 0.00, 1.00]$	$e_{5,1}$
v_6	$[1.00, 0.00, 1.00]$	$e_{6,5}$
v_7	$[1.00, 1.00, 1.00]$	*null*
v_8	$[0.00, 1.00, 1.00]$	$e_{8,4}$
v_9	$[0.25, 0.75, 0.50]$	*null*

Table 3.2: Edge list.

Edge	**Origin**	**Twin**	**Next**	**Previous**	**Incident Facet**
$e_{1,2}$	v_1	*nil*	$e_{2,6}$	$e_{5,1}$	f_1
$e_{2,6}$	v_2	*nil*	$e_{6,5}$	$e_{1,2}$	f_1
$e_{6,5}$	v_6	*nil*	$e_{5,1}$	$e_{2,6}$	f_1
$e_{5,1}$	v_5	$e_{1,5}$	$e_{1,2}$	$e_{6,5}$	f_1
$e_{1,5}$	v_1	$e_{5,1}$	$e_{5,8}$	$e_{4,1}$	f_4
$e_{5,8}$	v_5	*nil*	$e_{8,4}$	$e_{1,5}$	f_4
$e_{8,4}$	v_8	*nil*	$e_{4,1}$	$e_{5,8}$	f_4
$e_{4,1}$	v_4	*nil*	$e_{1,5}$	$e_{8,4}$	f_4

Finally, some of the polytopes that arise in the deconfliction of trajectories have internal vertices. Thus, it is important that the geometric queries used do not require the $\mathcal{V}$-representation to be minimal. Note that vertex v_9 in Figure 3.2 is internal, and does not appear in the Schlegel diagrams.

Table 3.3: Facet list.

Facet	**Incident Edge**
f_1	$e_{1,2}$
f_4	$e_{1,5}$

3.1.2 Bézier curves

A brief overview of Bézier curves, their properties, and algorithms is provided in this section. For simplicity, all subscripts and superscripts are dropped and $\boldsymbol{p_d}(\zeta)$ is used to denote a spatial path, whereas $v_d(\hat{t}_d)$ refers to the desired speed profile.

Definition 8 *A Bézier curve in $\mathbb{R}^m$ is a polynomial function $\boldsymbol{r}(\zeta) : [0, 1] \to \mathbb{R}^m$*

$$\boldsymbol{r}(\zeta) = \sum_{k=0}^{n} \bar{\boldsymbol{r}}_{\boldsymbol{k}}\, b_k^n(\zeta), \qquad b_k^n(\zeta) := \binom{n}{k}(1-\zeta)^{n-k}\zeta^k,$$

where ζ is the dimensionless curve parameter, $\bar{\boldsymbol{r}}_{\boldsymbol{k}} \in \mathbb{R}^m$ is the kth control point, and $b_k^n(\zeta)$ are the Bernstein polynomials, which comprise the Bernstein basis of degree n.

The Bernstein basis was first introduced by Russian mathematician Sergei N. Bernstein in 1912 as a mathematical artifact in the proof of the Weierstrass theorem [56]. In the 1960s, French engineer Pierre Bézier leveraged this basis for the generation of curves and surfaces in automotive design [57]. Since then, Bézier curves have become ubiquitous in CAD applications, gaming and animation, and even digital calligraphy for their advantageous properties [37]. In this thesis, Bézier curves were chosen as an element in the geometric model for a variety of reasons. Numerical stability, favorable geometric properties, availability of efficient proximity queries and root-finding algorithms are the most notable. The following is a partial list of the attributes that are leveraged across this thesis, and were decisive in the selection of Bézier curves:

i) **Non-negative basis**: the Bernstein polynomials $b_k^n(\zeta) \geq 0$ for all $\zeta \in [0, 1]$ and $k = 0, \ldots, n$.

ii) **Partition of unity**: the Bernstein polynomials that comprise a basis of degree n satisfy

$$\sum_{k=0}^{n} b_k^n(\zeta) \equiv 1, \qquad \forall \zeta \in \mathbb{R}.$$

iii) **Bounding polytope**: properties i) and ii) prove that Bézier curves lie within the convex hull of their control points, henceforth referred to as the *control polytope*. Figure 3.3 shows in red a spatial path $\boldsymbol{p}_{\boldsymbol{d}}(\zeta)$ designed using a Bézier curve of degree 6. The control points $\bar{\boldsymbol{p}}_{\boldsymbol{d}_{\boldsymbol{k}}}$ are depicted by circular markers, and the facets of the control polytope are colored in magenta. As illustrated in Figure 3.3, the control polytope naturally defines a bounding region where the curve is contained in its entirety. This property and the Gilbert-Johnson-Keerthi (GJK) algorithm [58, 59, 60] can be leveraged to perform proximity queries that involve Bézier curves and polytopes in $\mathbb{R}^2$ and $\mathbb{R}^3$ [61]. Note that some of the control points that define the curve could be internal vertices in the control polytope. This highlights the importance of dealing with non-minimal $\mathcal{V}$-representations when performing proximity queries with Bézier curves. Another direct consequence of the Bernstein basis, often leveraged in proximity queries, is that Bézier curves start and finish at their first and last control points, as shown in Figure 3.3.

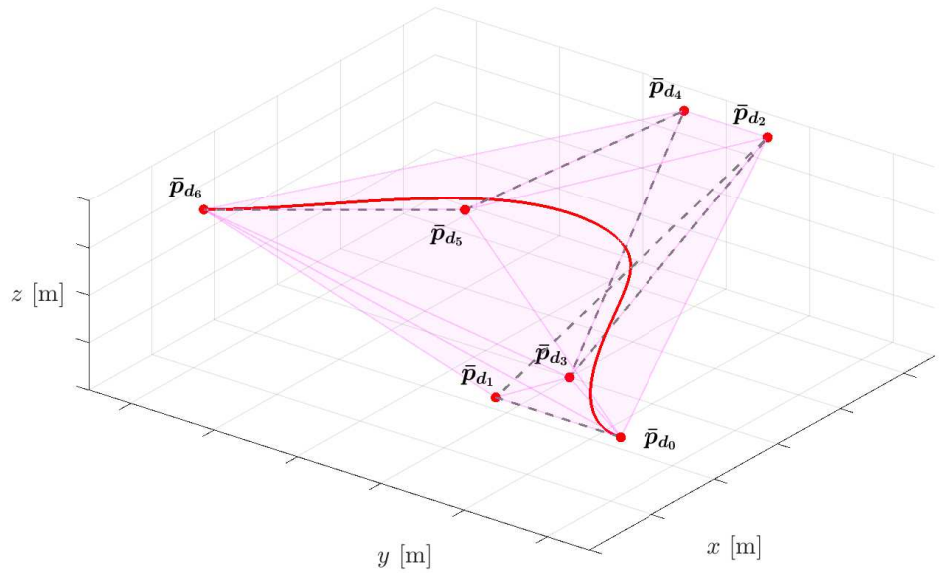


Figure 3.3: Spatial path designed using a Bézier curve.

iv) **Degree elevation**: a polynomial curve $\boldsymbol{r}(\zeta)$ expressed in a Bernstein basis of degree n with control points $\bar{\boldsymbol{r}}_{\boldsymbol{k}} \in \mathbb{R}^m$ can be rewritten in a Bernstein basis of degree $n+1$ as

$$\boldsymbol{r}(\zeta) = \sum_{k=0}^{n} \bar{\boldsymbol{r}}_{\boldsymbol{k}}^{*}\, b_k^{n+1}(\zeta), \qquad \text{with} \qquad \bar{\boldsymbol{r}}_{\boldsymbol{k}}^{*} = \begin{cases} \bar{\boldsymbol{r}}_{\boldsymbol{0}}, & \text{if } k = 0, \\ \frac{k}{n+1}\bar{\boldsymbol{r}}_{k-1} + \left(1 - \frac{k}{n+1}\right)\bar{\boldsymbol{r}}_k, & \text{if } k = 1, \ldots, n, \\ \bar{\boldsymbol{r}}_{\boldsymbol{n}}, & \text{if } k = n+1. \end{cases}$$

v) The set of polynomial curves is closed under the operations of addition/subtraction, multiplication, composition, differentiation, and integration. This facilitates the development of numerical software that exclusively relies on the manipulation of the control points.

- **Addition/subtraction**: let $\boldsymbol{r}(\zeta)$ and $\boldsymbol{s}(\zeta)$ be polynomial curves expressed in a Bernstein basis of degree n with control points $\bar{\boldsymbol{r}}_{\boldsymbol{k}}, \bar{\boldsymbol{s}}_{\boldsymbol{k}} \in \mathbb{R}^m$. Then, the control points of $\boldsymbol{r}(\zeta) \pm \boldsymbol{s}(\zeta)$ are obtained by adding/subtracting the corresponding control points

$$\boldsymbol{r}(\zeta) \pm \boldsymbol{s}(\zeta) = \sum_{k=0}^{n} \left(\bar{\boldsymbol{r}}_{\boldsymbol{k}} \pm \bar{\boldsymbol{s}}_{\boldsymbol{k}}\right) b_k^n(\zeta).$$

  If the degree of the bases for $\boldsymbol{r}(\zeta)$ and $\boldsymbol{s}(\zeta)$ does not match, then the lowest-degree basis must be degree elevated until their degrees coincide.

- **Multiplication**: let $u(\zeta)$ be a polynomial curve expressed in a Bernstein basis of degree p with control points $\bar{u}_k \in \mathbb{R}$. Then, the product of $u(\zeta)$ and $\boldsymbol{r}(\zeta)$ yields a Bézier curve $\boldsymbol{v}(\zeta) = u(\zeta)\boldsymbol{r}(\zeta)$ with a basis of degree $n+p$ and control points

$$\bar{\boldsymbol{v}}_{\boldsymbol{k}} = \sum_{j=\max(0,k-n)}^{\min(p,k)} \frac{\binom{p}{j}\binom{n}{k-j}}{\binom{n+p}{k}} \bar{u}_j \bar{\boldsymbol{r}}_{\boldsymbol{k-j}}, \qquad k = 0, \ldots, n+p.$$

- **Composition**: let $\zeta(\hat{t}_d)$ be a polynomial curve expressed in a Bernstein basis of degree p with control points $\bar{\zeta}_k \in \mathbb{R}$. The composition of $\boldsymbol{r}(\zeta)$ with $\zeta(\hat{t}_d)$ yields a Bézier curve $\boldsymbol{r}(\hat{t}_d) = \boldsymbol{r} \circ \zeta = \boldsymbol{r}(\zeta(\hat{t}_d))$ with a basis of degree np

$$\boldsymbol{r}(\hat{t}_d) = \sum_{k=0}^{np} \bar{\boldsymbol{r}}^{*}_{\boldsymbol{k}}\, b_k^{np}(\hat{t}_d),$$

with control points $\bar{\boldsymbol{r}}^{*}_{\boldsymbol{k}} = \boldsymbol{a}^{\boldsymbol{n}}_{\boldsymbol{0,k}}$ for $k = 0, \ldots, np$. The coefficients $\boldsymbol{a}^{\boldsymbol{\ell}}_{\boldsymbol{i,j}}$ are defined through an iterative procedure [62] as follows:

$$\boldsymbol{a}^{\boldsymbol{\ell}}_{\boldsymbol{i,j}} = \begin{cases} \bar{\boldsymbol{r}}_{\boldsymbol{i}}, & \text{if } \ell = 0, \\ \dfrac{1}{\binom{\ell p}{j}} \displaystyle\sum_{q=\max(0,j-p)}^{\min(j,p(\ell-1))} \binom{p(\ell-1)}{q}\binom{p}{j-q}\left((1-\bar{\zeta}_{j-q})\boldsymbol{a}^{\boldsymbol{\ell-1}}_{\boldsymbol{i,q}} + \bar{\zeta}_{j-q}\boldsymbol{a}^{\boldsymbol{\ell-1}}_{\boldsymbol{i+1,q}}\right), & \text{otherwise,} \end{cases}$$

with $\ell = 0, \ldots, n$, $i = 0, \ldots, n-\ell$, and $j = 0, \ldots, \ell p$. In this thesis, this method is often used in the reparameterization of paths as a function of the normalized time $\hat{t}_d$.

- **Derivation**: the parametric derivative of $\boldsymbol{r}(\zeta)$ can be expressed in a Bernstein basis of degree $n-1$ as

$$\boldsymbol{r}'(\zeta) := \frac{\mathrm{d}\boldsymbol{r}(\zeta)}{\mathrm{d}\zeta} = \sum_{k=0}^{n-1} n\left(\bar{\boldsymbol{r}}_{k+1} - \bar{\boldsymbol{r}}_k\right) b_k^{n-1}(\zeta).$$

As a result, first and higher-order boundary conditions on a path $\boldsymbol{p_d}(\zeta)$ can be imposed by solving a system of linear equations on the control points. Figure 3.3 illustrates the geometrical interpretation of the equation above. The path $\boldsymbol{p_d}(\zeta)$ is tangent to the line segments defined by the first and second control points $(\bar{\boldsymbol{p}}_{\boldsymbol{d_0}}, \bar{\boldsymbol{p}}_{\boldsymbol{d_1}})$, as well as as the second-to-last and last control points $(\bar{\boldsymbol{p}}_{\boldsymbol{d_5}}, \bar{\boldsymbol{p}}_{\boldsymbol{d_6}})$.

- **Integration**: the integral of $\boldsymbol{r}(\zeta)$ can be expressed in a Bernstein basis of degree $n+1$ as

$$\int \boldsymbol{r}(\zeta)\,\mathrm{d}\zeta = \boldsymbol{C} + \sum_{k=1}^{n+1}\left(\frac{1}{n+1}\sum_{j=0}^{k-1}\bar{\boldsymbol{r}}_j\right) b_k^{n+1}(\zeta),$$

  where $\boldsymbol{C} \in \mathbb{R}^m$ is a constant vector. Hence, integral conditions also yield linear systems of equations on the control points. The expression above is often used in the design of the speed profile $v_d(\hat{t}_d)$, to ensure that vehicles travel through the entire arclength ℓ associated to each trajectory segment in the allocated time δt_d

$$\delta t_d \int_0^1 v_d(\hat{t}_d)\,\mathrm{d}\hat{t}_d = \ell.$$

vi) **Shape handles**: the control points of a Bézier curve serve as "shape handles" [37, 2] to intuitively modify its geometry. As a result, portions of the flight plan produced by the trajectory generation algorithm that do not fully capture the considerations of a human operator can be adjusted on the fly by dragging the corresponding control points [63].

vii) **Numerical stability**: the Bernstein basis is "*optimally stable*" [64]. In other words, it is not possible to design a non-negative basis for generic polynomials of degree n that can consistently reduce the condition number[4] of this basis. Consequently, Bézier curves present excellent numeric stability, and floating-point error propagation behavior.

viii) **Curve evaluation**: given a Bézier curve $\boldsymbol{r}(\zeta)$ and a parameter value $\alpha \in [0,1]$ where the curve is to be evaluated, a modification of the *Horner's method* for the Bernstein basis, similar to the one detailed in [65], is used in this thesis to return $\boldsymbol{r}(\alpha)$.

ix) **Curve subdivision**: given a Bézier curve $\boldsymbol{p_d}(\zeta)$ expressed in a Bernstein basis of degree n with control points $\bar{p}_{d_i}$ with $i = 0, \ldots, n$, and a parameter value $\alpha \in [0,1]$ where the curve is to be split, the *de Casteljau algorithm* is used in this thesis to return two Bézier curves

[4] Assume the coefficients $\bar{u}_k \in \mathbb{R}$ of a polynomial $u(\zeta)$ expressed in a basis $\beta = \{\beta_0^n, \ldots, \beta_n^n\}$ of degree n are subject to quantization errors with maximum relative magnitude ϵ. Then, the bound on the perturbation δu caused by the quantization errors can be expressed in terms of the condition number $C_\beta(u(\zeta))$ for the polynomial $u(\zeta)$ in the basis β as $|\delta u| \leq C_\beta(u(\zeta))\epsilon$ with $C_\beta(u(\zeta)) = \sum_{k=0}^{n} |\bar{u}_k \beta_k^n(\zeta)|$.

$$\boldsymbol{q_d}(\zeta) = \sum_{k=0}^{n} \bar{\boldsymbol{q}}_{\boldsymbol{d_k}}\, b_k^n(\zeta), \qquad \boldsymbol{r_d}(\zeta) = \sum_{k=0}^{n} \bar{\boldsymbol{r}}_{\boldsymbol{d_k}}\, b_k^n(\zeta),$$

with control points $\bar{\boldsymbol{q}}_{\boldsymbol{d_k}} = \boldsymbol{a_k^k}$ and $\bar{\boldsymbol{r}}_{\boldsymbol{d_k}} = \boldsymbol{a_n^{n-k}}$ for $k = 0, \ldots, n$. The coefficients $\boldsymbol{a_i^\ell}$ are defined through the following iterative procedure:

$$\boldsymbol{a_i^\ell} = \begin{cases} \bar{\boldsymbol{p}}_{\boldsymbol{d_i}}, & \text{if } \ell = 0, \\ (1-\alpha)\boldsymbol{a_{i-1}^{\ell-1}} + \alpha \boldsymbol{a_i^{\ell-1}}, & \text{otherwise,} \end{cases}$$

for $\ell = 0, \ldots, n$ and $i = \ell, \ldots, n$. The curves $\boldsymbol{q_d}(\zeta)$ and $\boldsymbol{r_d}(\zeta)$ represent the first and second subdivisions of $\boldsymbol{p_d}(\zeta)$ which are illustrated in Figure 3.4, and satisfy

$$\boldsymbol{q_d}(\zeta) = \boldsymbol{p_d}(\zeta\alpha), \qquad \boldsymbol{r_d}(\zeta) = \boldsymbol{p_d}(\alpha + \zeta(1-\alpha)), \qquad \text{with} \qquad \zeta \in [0, 1].$$

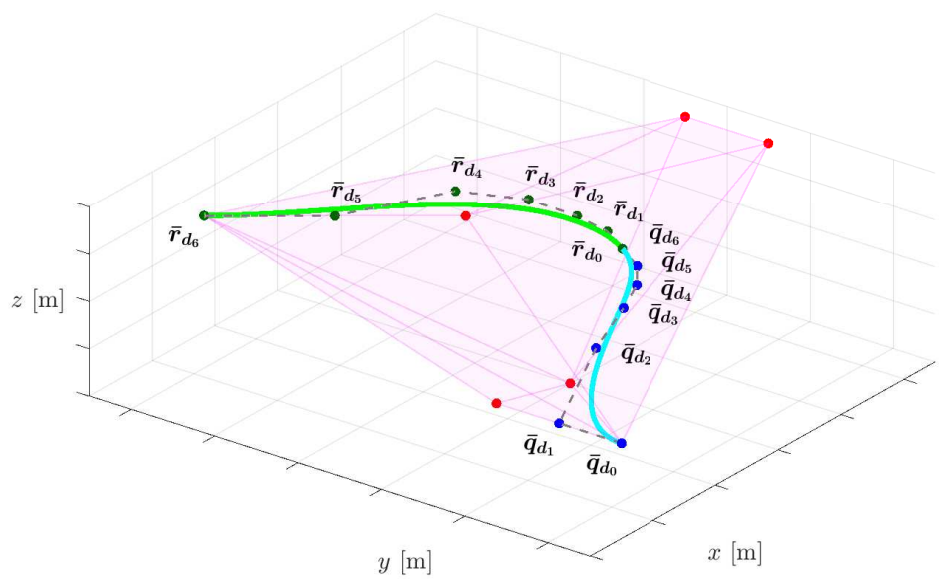


Figure 3.4: Curve subdivision using the de Casteljau algorithm.

x) **Variation-diminishing property**: given a Bézier curve $u(\zeta)$ with control points $\bar{u}_k \in \mathbb{R}$, the number of real roots n_r on the interval $(0, 1)$ is less or equal to the number of sign changes in the control points n_c by an even amount

$$n_r = n_c - 2m,$$

where $m \in \mathbb{N}$. Thus, if $n_c = 1$ then $n_r = 1$. The combination of this property with the de Casteljau algorithm and the Newton-Raphson method is an efficient tool for fast isolation and approximation of real roots for n-degree polynomials [66, 67, 37].

The following section uses some of these properties and algorithms for polytopes and Bézier curves to perform proximity queries among objects in the geometric model presented.

3.2 Proximity Queries

Let $\mathcal{A}$ and $\mathcal{B}$ be two objects in the geometric model described in Section 3.1. Thus, the pair $(\mathcal{A}, \mathcal{B})$ can either be composed of two polytopes, a polytope and a Bézier curve, or a pair of Bézier curves. Then, to answer questions regarding the relative distance between these objects, define the *separating distance* between $\mathcal{A}$ and $\mathcal{B}$ as

$$d(\mathcal{A}, \mathcal{B}) := \min_{\boldsymbol{x} \in \mathcal{A},\, \boldsymbol{y} \in \mathcal{B}} \|\boldsymbol{x} - \boldsymbol{y}\|.$$

For practical reasons, the scope of proximity queries studied here is limited to the following types:

i) **Separation distance queries** between $\mathcal{A}$ and $\mathcal{B}$ return an *approximate separating distance* $\hat{d}$ within some small error ϵ of the true separating distance, that is

$$d(\mathcal{A}, \mathcal{B}) - \epsilon \leq \hat{d} \leq d(\mathcal{A}, \mathcal{B}) + \epsilon.$$

Generally, finite precision machines cannot return the true separating distance, due to the approximation errors introduced by the floating-point arithmetic [68]. However, given the expected range for the input arguments of the distance query, the software in [69, 70] can be leveraged in some cases to find an over-approximation of the accumulated absolute error $\bar{\epsilon} \geq \epsilon$ that a query may incur in.

ii) **Tolerance verification queries** between $\mathcal{A}$ and $\mathcal{B}$ determine whether the approximate distance $\hat{d}$ is greater than a *safety distance* $d_s(\mathcal{A}, \mathcal{B})$. This involves evaluating the inequality

$$\hat{d} > d_s(\mathcal{A}, \mathcal{B}) + \bar{\epsilon},$$

and returning *true* or *false* accordingly. In practice, computing $\hat{d}$ can be computationally expensive, and algorithms often resort to computing upper and lower bounds $\underline{\hat{d}} \leq \hat{d} \leq \bar{\hat{d}}$ to conclude that

$$\begin{aligned} \underline{\hat{d}} > d_s(\mathcal{A}, \mathcal{B}) + \bar{\epsilon} &\implies \hat{d} > d_s(\mathcal{A}, \mathcal{B}) + \bar{\epsilon}, \\ \bar{\hat{d}} \leq d_s(\mathcal{A}, \mathcal{B}) + \bar{\epsilon} &\implies \hat{d} \leq d_s(\mathcal{A}, \mathcal{B}) + \bar{\epsilon}. \end{aligned} \tag{3.1}$$

In the context of the geometric model presented in Section 3.1, the selection of an appropriate $d_s(\mathcal{A}, \mathcal{B})$ must be informed by the uncertainty associated with $\mathcal{A}$ and $\mathcal{B}$. For instance, let $\mathcal{A}$ be jth trajectory segment assigned to the ith vehicle $\boldsymbol{p}^{\boldsymbol{j}}_{\boldsymbol{d},\boldsymbol{i}}(\zeta_i^j)$ with associated uncertainty distance $d_{p,i}$, and let $\mathcal{B}$ be the kth obstacle in the workspace $\mathcal{O}_k$ with associated uncertainty distance $d_{o,k}$. Then, to ensure that the ith vehicle maintains a safe separation with the kth obstacle choose a safety distance

$$d_s(\boldsymbol{p}^{\boldsymbol{j}}_{\boldsymbol{d},\boldsymbol{i}}, \mathcal{O}_k) = c_s \left(d_{p,i} + d_{o,k} \right),$$

where $c_s > 1$ is a *safety factor* that provides an additional margin for unmodelled errors.

iii) **Collision queries** determine whether objects $\mathcal{A}$ and $\mathcal{B}$ intersect. This involves evaluating the inequality

$$\hat{d} > \bar{\epsilon},$$

and returning *true* or *false* accordingly. In practice, the value for $\hat{d}$ is rarely computed and algorithms resort to the same strategy described for tolerance verification queries to conclude that

$$\begin{aligned} \underline{\hat{d}} > \bar{\epsilon} &\implies \hat{d} > \bar{\epsilon}, \\ \bar{\hat{d}} \leq \bar{\epsilon} &\implies \hat{d} \leq \bar{\epsilon}. \end{aligned} \tag{3.2}$$

Currently, collision queries are the most widespread type of proximity query used in robotics to discard paths that are deemed unsafe. However, given the formulation in Section 3.1, this query cannot be leveraged to safely deconflict $\mathcal{A}$ and $\mathcal{B}$, unless the corresponding tubes and/or

blown-out safety buffers illustrated in Figure 3.1 are explicitly constructed. As discussed previously, this is computationally expensive and may only be beneficial if the execution time of collision queries clearly outperforms that of tolerance verification queries.

Proximity queries between convex polytopes have been widely studied in the literature. The software packages in [71, 72, 73] perform some of the aforementioned proximity queries for pairs of polytopes with efficacy. Among the existing methods, the Gilbert-Johnson-Keerthi (GJK) algorithm [58] exhibits linear-time performance in practice, and utilizes the fact that the minimum distance between two polytopes $\mathcal{A}$ and $\mathcal{B}$ is the same as the minimum distance between the origin and the Minkowski sum set $\mathcal{A} \oplus (-\mathcal{B})$. The work in [74] proposes a modification of the GJK algorithm to improve performance in coherent environments, and achieves almost-constant time complexity. The Lin-Canny algorithm [75] was one of the first to exploit motion coherence, and exhibits constant runtime in coherent environments. In addition, not only algorithms and frame coherence affect the runtime performance of these queries, but also the polytope representation used. The authors in [76] present an optimal algorithm that takes $O(\log n)$ time, where n is the number of faces of the polyhedra. It relies on the bounded Dobkin-Kirkpatrick hierarchical representation [77], which requires preprocessing the polyhedra.

Proximity queries among convex polytopes can be formulated as a convex optimization problem. However, in general, proximity queries involving Bézier curves define a non-convex problem. Existing literature uses a wide variety of strategies to tackle this problem. Two common approaches are analytic and subdivision methods. Analytic techniques often involve curve implicitation, and require the solution of systems of non-linear equations, as described in [78, 79]. Subdivision methods, on the other hand, perform successive partitions of the curve domain and examine the relative distance between these partitions, often coupled with a bounding method. The authors in [61] use this approach to compute the distance between a pair of Bézier curves and surfaces, and propose an adaptive subdivision scheme. The algorithm in [80] uses swept spheres to iteratively cull smaller curve segments and compute the distance between a pair of Bézier curves. The authors in [81] develop a branch and bound method for absolutely continuous parametric curves. In this case, they devised an elegant bounding method for this class of curves that results in shrinking ellipsoids. In this thesis, we leverage the bounding properties of the control polytope of a Bézier curve, differ-

ent variations of the GJK algorithm, and a static subdivision scheme to tackle proximity queries involving Bézier curves.

As described in Section 3.1, the obstacles $\mathcal{O}_k$ in the geometric model are static. As a result, proximity queries between an obstacle $\mathcal{O}_k$ and a trajectory segment $\boldsymbol{p}^{\boldsymbol{j}}_{\boldsymbol{d},\boldsymbol{i}}(\zeta^j_i)$ are also static[5]. Trajectory segments are inherently dynamic objects, and can be deconflicted using the spatial or temporal dimensions. If one wishes to ensure *spatial separation* between a pair of Bézier curves

$$\|\boldsymbol{p}^{\boldsymbol{j_1}}_{\boldsymbol{d},\boldsymbol{i_1}}(\zeta^{j_1}_{i_1}) - \boldsymbol{p}^{\boldsymbol{j_2}}_{\boldsymbol{d},\boldsymbol{i_2}}(\zeta^{j_2}_{i_2})\| > d_s, \quad \forall \zeta^{j_1}_{i_1}, \zeta^{j_2}_{i_2} \in [0,1], \quad i_1 \neq i_2, \quad i_1, i_2 \in \mathcal{I}, \quad j_1 \in \mathcal{J}_{i_1}, \quad j_2 \in \mathcal{J}_{i_2},$$

then a static proximity query can be utilized to check whether the inequality above is met. In this case, for the sake of simplicity curves are often expressed in terms of their dimensionless parameters. However, if one wishes to enforce *temporal separation* between a pair of Bézier curves

$$\|\boldsymbol{p}^{\boldsymbol{j_1}}_{\boldsymbol{d},\boldsymbol{i_1}}(t_d) - \boldsymbol{p}^{\boldsymbol{j_2}}_{\boldsymbol{d},\boldsymbol{i_2}}(t_d)\| > d_s, \quad \forall t_d \in [t_{init}, t_{end}], \quad i_1 \neq i_2, \quad i_1, i_2 \in \mathcal{I}, \quad j_1 \in \mathcal{J}_{i_1}, \quad j_2 \in \mathcal{J}_{i_2},$$

which is a common approach in cluttered scenarios due to the lack of alternative routes, then the following procedure is used. First, the curve segments are clipped using the de Casteljau algorithm so that the initial and final times of the curve segments being analyzed coincide, as in the equation above. Second, the difference trajectory $\boldsymbol{p}^{\boldsymbol{j_1}}_{\boldsymbol{d},\boldsymbol{i_1}}(t_d) - \boldsymbol{p}^{\boldsymbol{j_2}}_{\boldsymbol{d},\boldsymbol{i_2}}(t_d)$ is computed. Next, a static proximity query between the difference trajectory and the origin determines whether the segments are temporally deconflicted

$$d(\boldsymbol{p}^{\boldsymbol{j_1}}_{\boldsymbol{d},\boldsymbol{i_1}}(t_d) - \boldsymbol{p}^{\boldsymbol{j_2}}_{\boldsymbol{d},\boldsymbol{i_2}}(t_d), \boldsymbol{0}) = \min_{t_d \in [t_{init}, t_{end}]} \|\boldsymbol{p}^{\boldsymbol{j_1}}_{\boldsymbol{d},\boldsymbol{i_1}}(t_d) - \boldsymbol{p}^{\boldsymbol{j_2}}_{\boldsymbol{d},\boldsymbol{i_2}}(t_d)\|.$$

Hence, what initially was a dynamic proximity query between two curves is reformulated as a static proximity query between the difference trajectory and the origin.

Subsection 3.2.1 describes the method developed in this thesis to ensure safe separation between

[5]If one wishes to extend the geometric model to include dynamic objects in the workspace there are two options. First, if the obstacles cannot be represented as a closed-form function of time, static proximity queries can be executed on a fixed-time interval. Unfortunately, this discrete-time approach can lead to missed collisions. Second, if the obstacles can be parameterized as a closed-form function of time, then novel continuous proximity queries must be developed for the geometric objects in question. The algorithm in [82] can serve as a starting point, since it provides a fast and continuous collision detection method for polyhedra.

the tubes associated with a trajectory and the safety buffer around each of the obstacles in the workspace, see Figure 3.1. Subsection 3.2.2 presents the method developed to ensure safe separation between the tubes associated with a pair of trajectories.

3.2.1 Tolerance Verification Query for a Polytope and a Bézier curve

Given the $\mathcal{V}$-representation of a convex polytope $\mathcal{O} \subset \mathbb{R}^3$ with the types of boundary representations described in Subsection 3.1.1, a polynomial parametric curve $\boldsymbol{p_d}(\zeta) : [0, 1] \to \mathbb{R}^3$ of degree n expressed in a Bernstein basis, and a safety distance d_s, the objective is to design an algorithm that returns the boolean flag

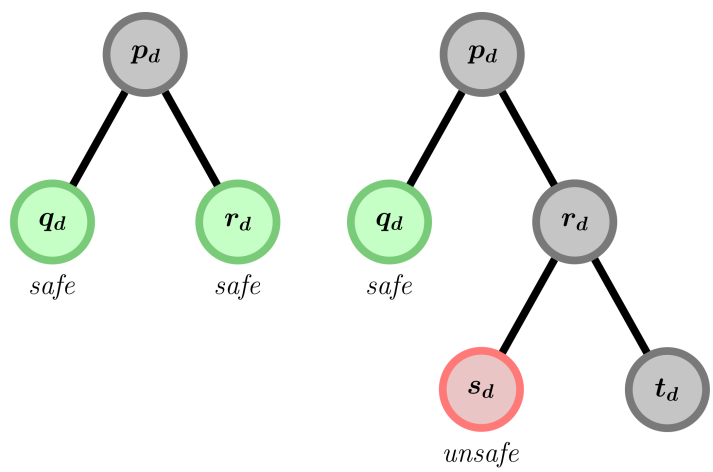
$$v_t = \begin{cases} \textit{true}\,, & \text{if } \hat{d}(\mathcal{O}, \boldsymbol{p_d}) > d_s + \bar{\epsilon}, \\ \textit{false}\,, & \text{otherwise.} \end{cases}$$

Algorithm 1 in Appendix A contains the pseudo-code proposed to solve the tolerance verification problem for a polytope and a Bézier curve. The algorithm relies on two fundamental methods: the de Casteljau algorithm for the subdivision of Bézier curves, and a modification of the GJK algorithm [58] designed to perform tolerance verification queries between polytopes, referred to as GJKTolerance in Appendix A.

Algorithm 1 in Appendix A also leverages three fundamental properties of Bézier curves: first, the control polytope of a Bézier curve defines a bounding region where the curve is fully contained; second, the curve starts and finishes at its first and last control points; and third, the control polytopes generated through recursive subdivisions of $\boldsymbol{p_d}(\zeta)$ converge to $\boldsymbol{p_d}(\zeta)$ itself, as illustrated in Figure 3.4. The algorithm proposed constructs depth-first a binary tree structure, as illustrated in Figure 3.5, until a solution is found. It returns $v_t = \textit{true}$ if the control polytopes $\mathcal{P}_k$ of the curve subdivisions located at all the leaf nodes satisfy $\hat{d}(\mathcal{O}, \mathcal{P}_k) > d_s + \bar{\epsilon}$, as shown in Figure 3.5a. On the other hand, the algorithm returns $v_t = \textit{false}$ as soon as a point on the curve fails a tolerance verification query with $\mathcal{O}$, as shown in Figure 3.5b. Figures 3.6 and 3.7 depict the geometric operations that the algorithm performs on the curve for two particular cases where $v_t = \textit{true}$ and $v_t = \textit{false}$. The captions in these figures provide an interpretation of the different steps within Algorithm 1. The polyhedron in magenta represents the convex hull of the control points of the

original curve $\boldsymbol{p_d}(\zeta)$, whereas the polyhedron in cyan represents an obstacle $\mathcal{O}$ in the workspace. The polyhedra in gray represent the boundary of the control polytopes after a curve subdivision. These polyhedra are colored green when the tolerance verification query with $\mathcal{O}$ returns *true*, and are colored red when the initial or final points of the curve subdivision violate the safe separation constraints. Notice that the binary tree in Figure 3.5a corresponds to the example in Figure 3.6; whereas the tree in Figure 3.5b matches the example in Figure 3.7.

The pseudo-code in Algorithm 1 proposes a recursive implementation. In practice, this can lead to excessive memory allocation, or even to a stack overflow due to an excessively deep recursion. Hence, it is recommended that this algorithm be implemented either using an equivalent iterative structure[6], or with a mechanism that limits the depth of the recursion. Conversely, this recursive structure can be easily exploited to develop an inductive proof.



(a) $v_t = \mathit{true}$ (b) $v_t = \mathit{false}$

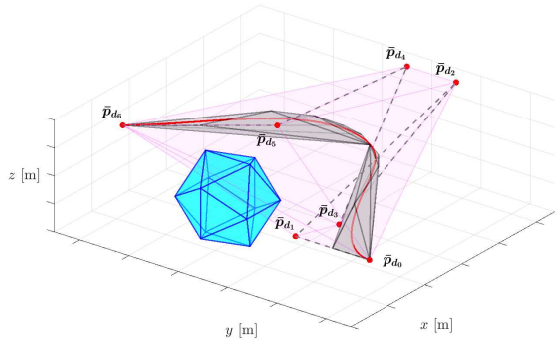
Figure 3.5: Binary tree representation of the tolerance verification algorithm for a polytope and a Bézier curve.

3.2.2 Tolerance Verification Query for a pair of Bézier curves

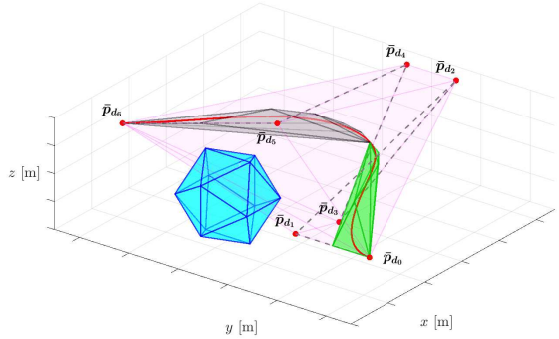
Given two polynomial parametric curves $\boldsymbol{p_d^1}(\zeta_1) : [0,1] \to \mathbb{R}^3$ and $\boldsymbol{p_d^2}(\zeta_2) : [0,1] \to \mathbb{R}^3$ of degree n_1 and n_2 expressed in a Bernstein basis, and a safety distance d_s, the objective is to design an algorithm that returns a boolean flag

$$v_t = \begin{cases} \mathit{true}\,, & \text{if } \hat{d}(\boldsymbol{p_d^1}, \boldsymbol{p_d^2}) > d_s + \bar{\epsilon}, \\ \mathit{false}\,, & \text{otherwise.} \end{cases}$$

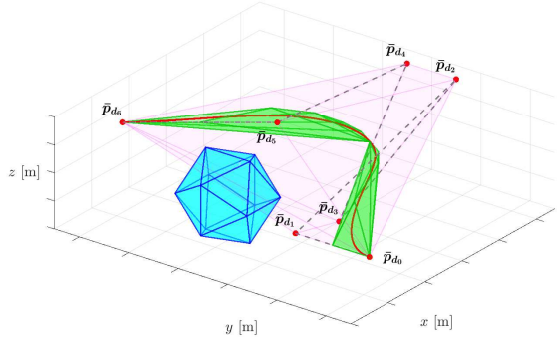
[6] A recursive tree search can always be implemented using an iterative structure, which is often more efficient [83]. In fact, the algorithm proposed in [81] uses an iterative structure that relies on a priority queue to perform tolerance verification queries between a polyhedral object and a broader class of parametric curves.



(a) $\boldsymbol{p_d}(\zeta)$ is divided into $\boldsymbol{q_d}(\zeta)$ and $\boldsymbol{r_d}(\zeta)$.

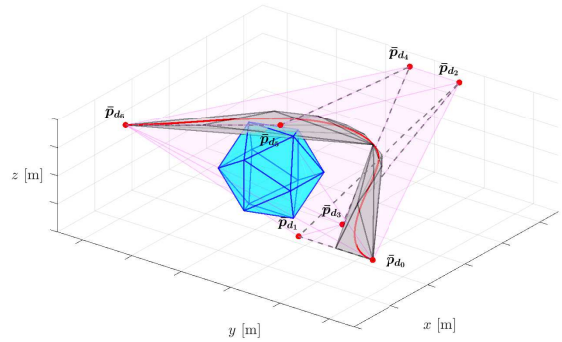


(b) The control polytope of $\boldsymbol{q_d}(\zeta)$ is deemed safe.

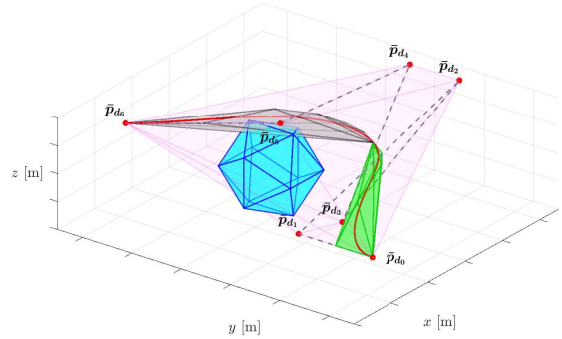


(c) The control polytope of $\boldsymbol{r_d}(\zeta)$ is deemed safe.

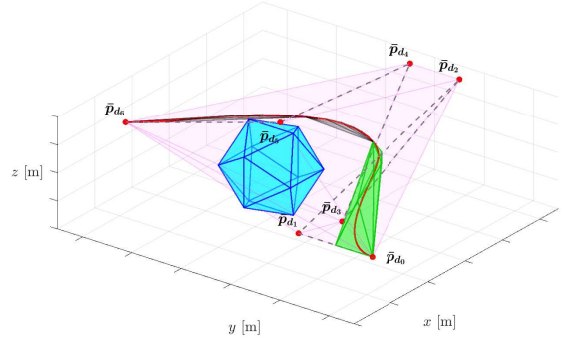
Figure 3.6: Tolerance verification algorithm, $v_t = true$.



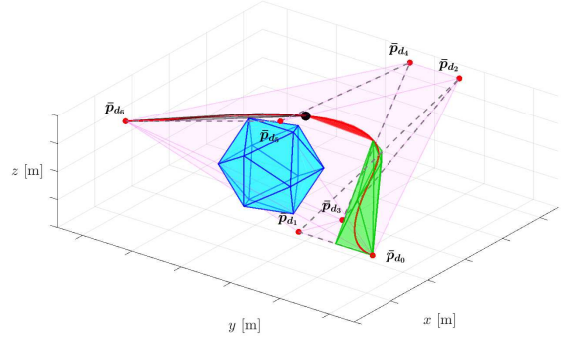
(a) $\boldsymbol{p_d}(\zeta)$ is divided into $\boldsymbol{q_d}(\zeta)$ and $\boldsymbol{r_d}(\zeta)$.



(b) The control polytope of $\boldsymbol{q_d}(\zeta)$ is deemed safe.



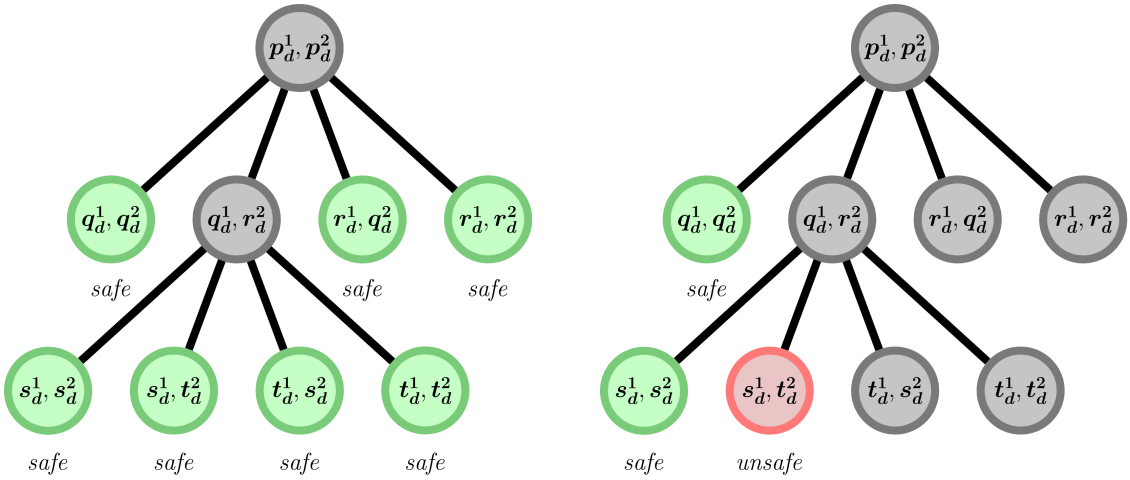
(c) $\boldsymbol{r_d}(\zeta)$ is divided into $\boldsymbol{s_d}(\zeta)$ and $\boldsymbol{t_d}(\zeta)$.



(d) The curve $\boldsymbol{s_d}(\zeta)$ is deemed unsafe.

Figure 3.7: Tolerance verification algorithm, $v_t = false$.

Algorithm 2 in Appendix A contains the pseudo-code proposed to solve the tolerance verification problem for a pair of Bézier curves. The algorithm relies on the same methods and properties as Algorithm 1. The main difference is that Algorithm 2 generates a quadtree, instead of a binary tree, as illustrated in Figure 3.8. The proposed method returns $v_t = \mathit{true}$ if the control polytopes $\mathcal{P}^1_{k_1}$ and $\mathcal{P}^2_{k_2}$ of all the curve subdivision pairs located at the leaf nodes satisfy $\hat{d}(\mathcal{P}^1_{k_1}, \mathcal{P}^2_{k_2}) > d_s + \bar{\epsilon}$, see Figure 3.8a. On the other hand, the algorithm returns $v_t = \mathit{false}$ as soon as it finds a pair of points in $\boldsymbol{p^1_d}$ and $\boldsymbol{p^2_d}$ that violate the safe separation constraints, as shown in Figure 3.8b. As in Algorithm 1, the recursive structure in Algorithm 2 is useful to develop proofs by induction. However, in practice an iterative implementation is recommended.



(a) $v_t = \mathit{true}$ (b) $v_t = \mathit{false}$

Figure 3.8: Quadtree representation of the tolerance verification algorithm for a pair of Bézier curves.

3.3 Silhouette

Given a convex polytope $\mathcal{O}$, a facet f_k of the polytope $\mathcal{O}$ is *visible* from a point of view $\boldsymbol{p_v} \notin \mathcal{O}$ if

$$(\boldsymbol{p_v} - \bar{\boldsymbol{v}}_{\boldsymbol{i}}) \cdot \boldsymbol{n_k} > \epsilon_v,$$

where $\boldsymbol{n_k}$ is the unit normal vector to facet f_k pointing outwards of $\mathcal{O}$, $\bar{\boldsymbol{v}}_{\boldsymbol{i}}$ are the coordinates of any vertex v_i incident to facet f_k, and ϵ_v is the *visibility threshold*. The visibility threshold is usually a

small number and can either have a physical interpretation when working with particular sensors, or could be chosen to ensure numerical stability of the visibility test when operating on a finite precision machine. The following definition employs this notion of facet visibility to characterize the silhouette of a polytope.

Definition 9 *The silhouette of a convex polytope $\mathcal{O}$ from a point of view $\boldsymbol{p_v} \notin \mathcal{O}$ is the closed sequence of edges that defines the boundary between the facets of $\mathcal{O}$ that are visible from $\boldsymbol{p_v}$ and those that are not.*

The silhouette of a polytope contains local geometric information that can be used for both simulation and path-planning purposes. In simulation, the silhouette can be leveraged to determine the region of an obstacle that is visible to a UAS, as illustrated in Figure 3.9, which can be useful to model the UAS perception of the environment. For path planning purposes, the silhouette captures relevant geometric features that can be utilized to avoid a collision with a particular obstacle. Note that in Figure 3.9 if a vehicle were to fly in a straight line along a direction contained within the convex hull of all vectors $\boldsymbol{u_{v,i}}$ then the UAS would eventually collide with the obstacle, where $i \in \mathcal{I}_s$ and $\mathcal{I}_s$ is the set of identification numbers of the vertices in the silhouette. This section presents the methods used in this thesis to compute the silhoutte for the three boundary representations described in Section 3.1.1:

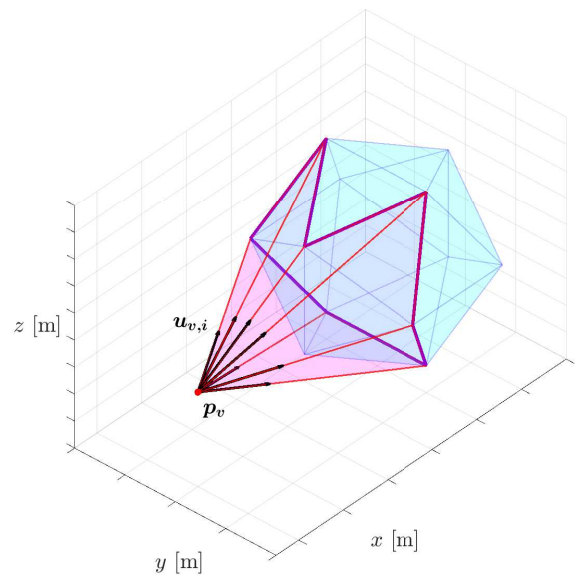


Figure 3.9: Silhouette of a polytope.

i) Given the complete boundary representation of an obstacle $\mathcal{O}$, the method implemented to retrieve the silhouette information is the flood fill scheme described in [55]. The algorithm leverages the DCEL data structure, and exhibits $O(\text{card}(\mathcal{I}_s))$ runtime performance.

ii) Given an incomplete boundary representation of an obstacle $\mathcal{O}$, Algorithm 3 in Appendix A

was designed to leverage the available boundary information, and construct sufficient boundary information to retrieve the silhouette.

iii) Given an obstacle with no boundary information $\mathcal{O}$, Algorithm 4 in Appendix A is used to first define a non-degenerate tetrahedron with four vertices located on the boundary of $\mathcal{O}$. Then, similar to Algorithm 3, sufficient boundary information is constructed to extract the silhouette.

Algorithms 3 and 4 avoid computing a complete boundary representation to return the silhouette. These methods rely on the Expanding Polytope Algorithm (EPA) to inflate a polytope $\mathcal{P}$ inside $\mathcal{O}$. When Algorithms 3 and 4 finish, the inflated polytope $\mathcal{P}$ contains at least the facets of $\mathcal{O}$ that are incident to the set of vertices that define the silhouette. The EPA is typically used to compute the penetration depth between two colliding polyhedra [55]. Equivalent methods to Algorithms 3 and 4 could be designed leveraging the gift-wrapping techniques that are often seen in convex hull algorihtms [84], instead of the EPA. In fact, such modifications of Algorithms 3 and 4 are expected to be simpler and have better runtime performance.

Once a non-degenerate polytope $\mathcal{P} \subset \mathcal{O}$ with vertices on the boundary of $\mathcal{O}$ has been obtained in Algorithm 4 through the function call nonDegenerateTetrahedon, then Algorithms 3 and 4 can be broken down into four identical steps:

i) The algorithms update $\mathcal{P}$ through multiple expansions until they find a visible facet of $\mathcal{P}$ on the boundary of $\mathcal{O}$. To do this, the algorithms select a visible facet f_k on $\mathcal{P}$, and expand repeatedly along the direction of its normal vector $\boldsymbol{n_k}$ using the EPA. This is performed by the function reachVisibleBoundary in Algorithms 3 and 4.

ii) When $\mathcal{P}$ contains a visible facet on the boundary of $\mathcal{O}$, then the algorithms expand $\mathcal{P}$ until they find two incident facets on the boundary of $\mathcal{O}$, one visible and one not visible. By definition, the edge that these facets share is on the silhouette. This is executed by the function reachSilhouette in Algorithms 3 and 4.

iii) Once the algorithms have identified a single edge on the silhouette, they use the EPA to expand $\mathcal{P}$ until it contains all the facets on the boundary of $\mathcal{O}$ that are incident to the vertices

that define the silhouette. The silhouette is completed when it forms a closed sequence of edges, and can be easily retrieved through a local search over the boundary between the facets of $\mathcal{P}$ that are visible and those that are not. This step is performed by the method closeSilhouette in Algorithms 3 and 4.

3.4 Performance Evaluation

This section summarizes the results for the runtime tests performed on the algorithms proposed within this chapter.

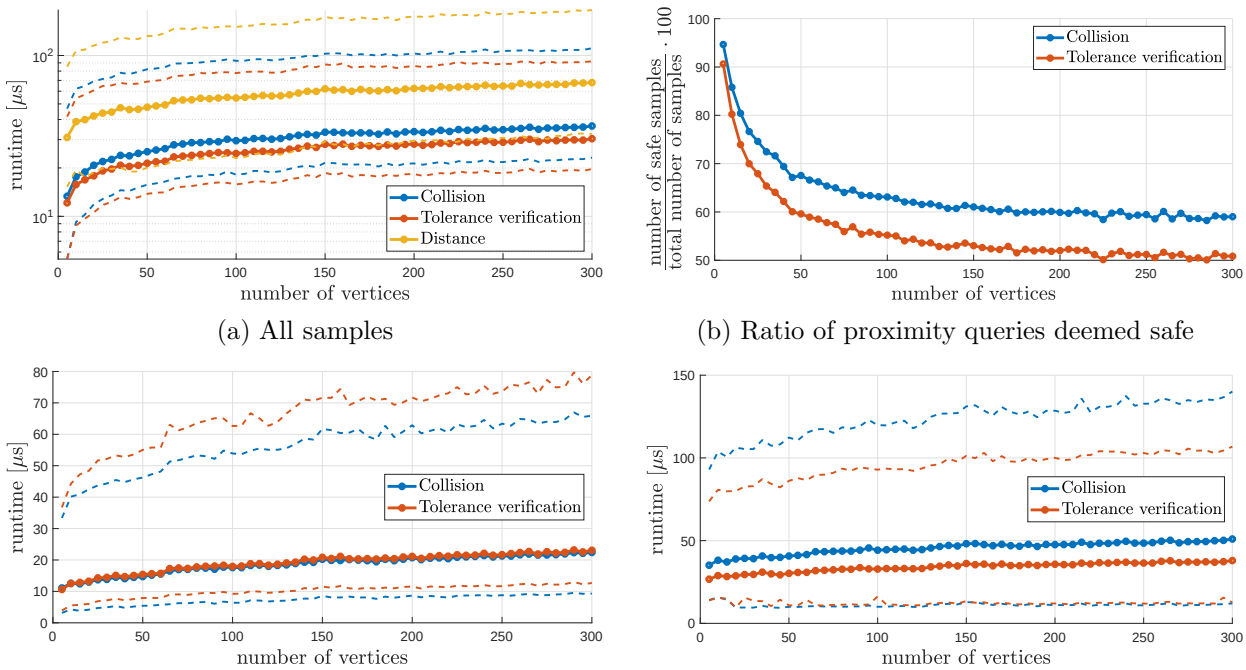
3.4.1 Proximity Queries

Figures 3.10 through 3.12 present the performance results for the nine proximity query algorithms used within this thesis to deconflict the objects in the geometric model described in Section 3.1. These nine algorithms result from the combination of the three proximity queries—collision, tolerance verification, and distance computation—with the three types of geometric pairs included in the model—a pair of polytopes, a polytope and a Bézier curve, and two Bézier curves. All the algorithms were implemented in Matlab® 2018b, and run on a portable computer equipped with 32.00 GB of RAM, an Intel(R) Core(TM) i7-7820HQ processor, and a 64-bit operating system. Matlab® is an interpreted programming platform, and thus the expected runtime is greater than that of a compiled programming language. However, relevant information can still be distilled by comparing the runtime results of the nine proximity query algorithms mentioned.

Pairs of Polytopes

Figure 3.10 shows the performance results for the different variations of the GJK algorithm designed to perform collision, tolerance verification, and distance queries for a pair of polytopes in $\mathbb{R}^3$. These algorithms will be referred to as GJKCollision, GJKTolerance, and GJKDistance, respectively. To evaluate the runtime of these algorithms, all queries were provided the same pairs of polytopes. These were randomly generated using the following procedure. First, the vertices of each polytope were sampled uniformly within a closed ball[7] $\beta^2_{\mathbf{0},r}$ with radius $r = 10\,\mathrm{m}$. Then, each polytope

[7] A closed ball in $\mathbb{R}^3$ of radius r centered at $\boldsymbol{c} \in \mathbb{R}^3$ with norm p is the set $\beta^p_{\boldsymbol{c},r} := \left\{\boldsymbol{x} \in \mathbb{R}^3 \,\middle|\, \|\boldsymbol{x} - \boldsymbol{c}\|_p \leq r\right\}$.



(a) All samples

(b) Ratio of proximity queries deemed safe

(c) Samples that returned $v_t = \mathit{true}$

(d) Samples that returned $v_t = \mathit{false}$

Figure 3.10: Performance results for proximity queries between polytopes.

was translated within a box using a translation vector that was sampled uniformly within a closed ball $\beta^{\infty}_{\mathbf{0},r}$ with radius $r = 25\,\mathrm{m}$. Internal vertices were maintained and counted towards the number of vertices shown in the abscissa of Figures 3.10a through 3.10d. No boundary representation was generated for these polytopes. The safety distance was set to $d_s = 2\,\mathrm{m}$, and the over-approximation for the accumulated round-off error was set to $\bar{\epsilon} = 10^{-6}$. These parameters were selected to ensure the numerical stability of the algorithms, and maintain the percentage of safe pairs in Figure 3.10b above 50%. Recall that a pair of objects is deemed safe in a collision query if $\hat{d} > \bar{\epsilon}$, whereas for tolerance verification queries the safe criterion is $\hat{d} > d_s + \bar{\epsilon}$.

Each data point in Figure 3.10a shows the average runtime for a batch of $10,000$ pairs of random polytopes. The lower and upper dashed lines represent the 5 and 95 percentile curves, respectively. Notice the logarithmic scale of the ordinate in Figure 3.10a. It is clear that distance queries are computationally more expensive than collision or tolerance verification queries. At first glance, Figure 3.10a seems to indicate that tolerance verification queries are faster than collision queries. However, when the data is separated based on the result of the tolerance verification query, as in Figures 3.10c and 3.10d, two conclusions follow. First, the average runtimes when $v_t = \mathit{false}$ are

larger than when $v_t = \mathit{true}$, compare Figures 3.10c and 3.10d. Second, on average collision queries are faster than tolerance verification queries when $v_t = \mathit{true}$, see Figure 3.10c; whereas tolerance verification queries are faster than collision queries when $v_t = \mathit{false}$, as depicted in Figure 3.10d. This is a direct result of the return mechanisms detailed in Equations (3.1) and (3.2). If the deciding factor is the lower bound $\hat{\underline{d}}$ then collision queries tend to be faster. However, if the deciding factor is the upper bound $\hat{\bar{d}}$ then tolerance verification queries tend to be faster.

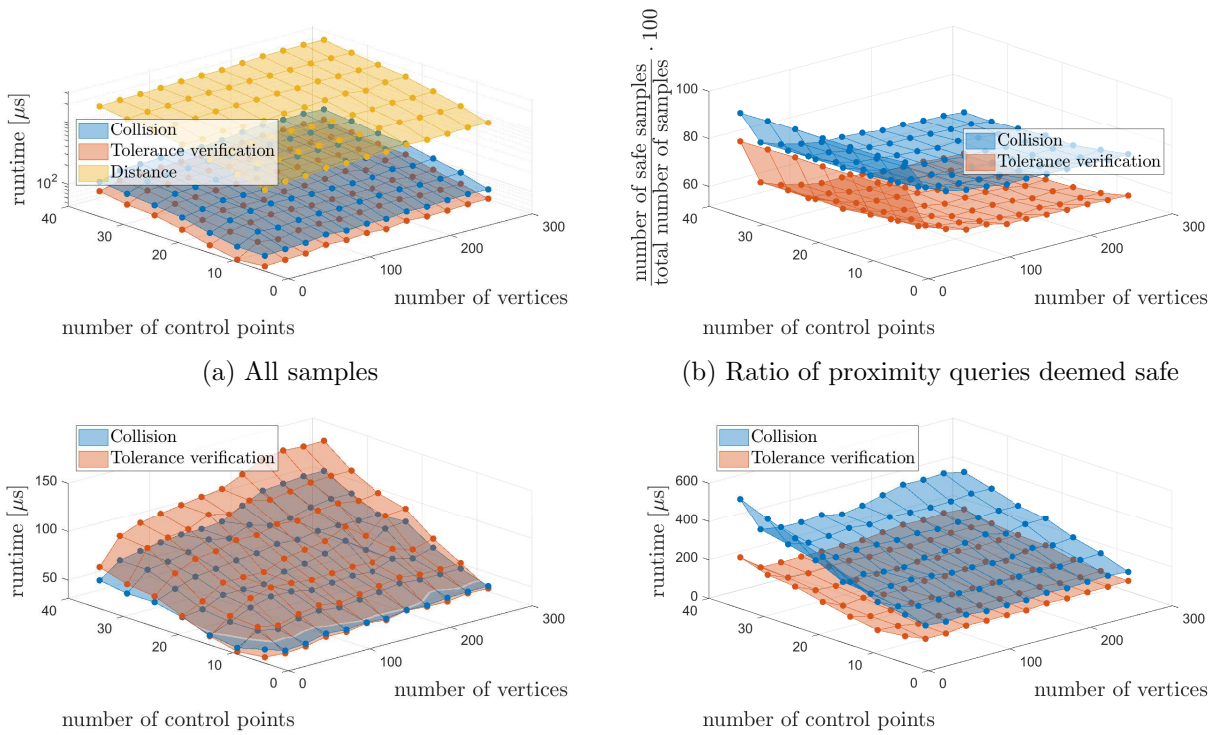
Both conclusions are relevant to understand and tackle the complexity of cluttered scenarios, which are widely recognized to be challenging for two main reasons. First, a larger number of obstacles in the workspace requires a greater number of proximity queries, which are the major bottleneck in existing motion-planning algorithms [85]. Second, narrow passages are created among the obstacles in cluttered environments. These are particularly difficult to find for existing sample-based motion-planning algorithms, due to the low probability of sampling within these regions [86, 87]. In addition, a greater portion of the proximity queries in a cluttered scenario is expected to classify the candidate paths generated by the motion-planning algorithm as unsafe. The results in Figure 3.10 suggest that this can lead to a further increase in the computational cost. In these scenarios, using tolerance verification queries, instead of collision queries, holds potential to reducing the overall planning times.

Polytopes and Bézier curves

Figure 3.11 shows the performance results for the proximity queries between a polytope and a Bézier curve in $\mathbb{R}^3$. The distance query implemented for these tests is a modification of the method proposed in [61] for a polytope and a Bézier curve, instead of a pair of Bézier curves, and relies on the algorithm GJKDistance. The tolerance verification query implements Algorithm 1, and the collision query is a modification of Algorithm 1 that uses GJKCollision, given in [55], instead of GJKTolerance. All proximity queries were provided the same pairs of polytopes and Bézier curves. These pairs were generated randomly using the following procedure. First, the vertices of the polytope and the control points of the curves were sampled uniformly within a closed ball $\beta^2_{\mathbf{0},r}$ with radius $r = 10\,\mathrm{m}$. Then, each set of vertices and control points were translated within a box using a translation vector sampled uniformly within a ball $\beta^\infty_{\mathbf{0},r}$ with radius $r = 11\,\mathrm{m}$. Again, the internal

vertices within the polytoped were maintained, and counted towards the number of vertices shown in Figures 3.11a through 3.11d. No boundary representation was generated for these polytopes. The safety distance was set to $d = 2\,\mathrm{m}$, and the over-approximation round-off error was set to $\bar{\epsilon} = 10^{-6}$. These parameters were selected to ensure the numerical stability of the algorithms, and maintain the percentage of safe pairs in Figure 3.11b above 50%.

Each data point in Figure 3.11a shows the average runtime for a batch of $10,000$ pairs of random polytopes and Bézier curves. Notice the logarithmic scale of the runtime axis in Figure 3.11a. The conclusions highlighted for the different variations of the GJK algorithm in Figure 3.10 also apply to Figure 3.11. Notice also that increasing the number of control points has a more costly effect than an increase in the number of vertices. The reason is that the de Casteljau algorithm requires $O(n^2)$ arithmetic operations, whereas the different variants of the GJK algorithm exhibit linear-time performance in practice [58, 88].



(a) All samples

(b) Ratio of proximity queries deemed safe

(c) Samples that returned $v_t = \mathit{true}$

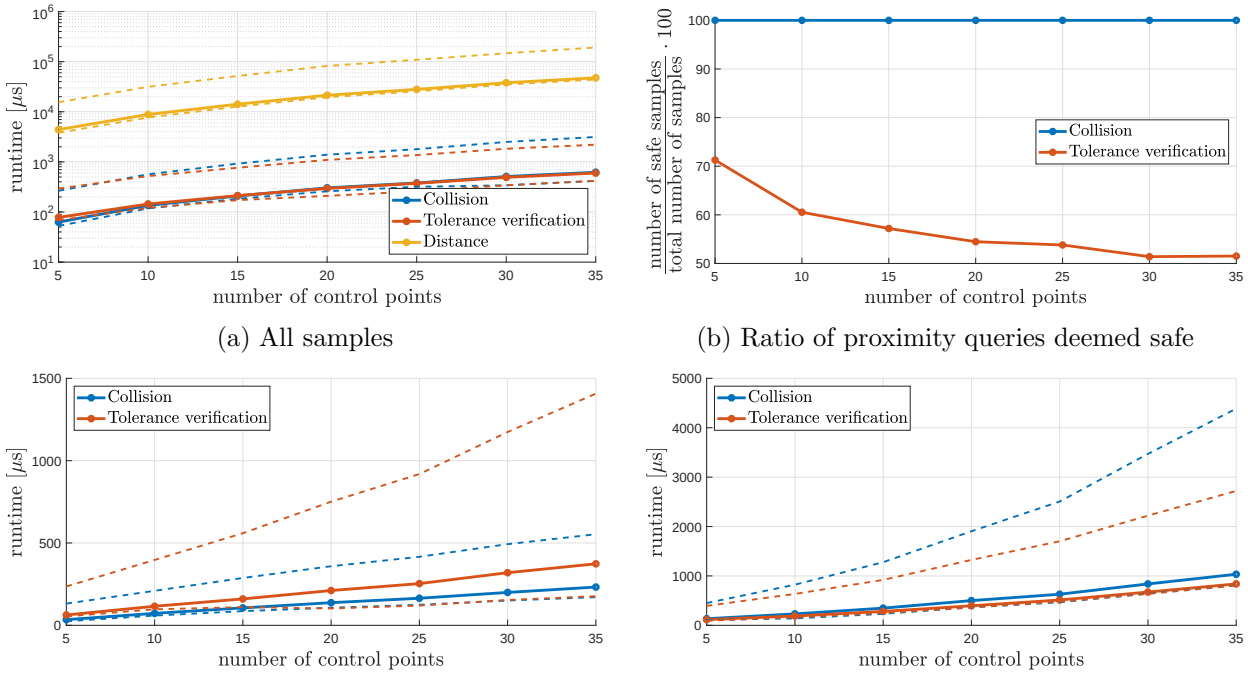
(d) Samples that returned $v_t = \mathit{false}$

Figure 3.11: Performance results for proximity queries between a polytope and a Bézier curve.

Pairs of Bézier curves

Figure 3.12 shows the performance results for the proximity queries between a pair of Bézier curves in $\mathbb{R}^3$. The distance query implemented for these tests is the algorithm proposed in [61] with a static subdivision scheme. The tolerance verification method is described in Algorithm 2, and the collision query implements a modification of Algorithm 2 that leverages GJKCollision, given in [55], instead of GJKTolerance. All proximity queries were provided the same pairs of Bézier curves. The Bézier curves were sampled randomly using the following procedure. First, the control points were sampled uniformly within a closed ball $\beta^2_{\mathbf{0},r}$ with radius $r = 10\,\mathrm{m}$. Then, each set of control points was translated within a box using a translation vector that was sampled randomly within a closed ball $\beta^\infty_{\mathbf{0},r}$ with radius $r = 4.6\,\mathrm{m}$. The safety distance was set to $d_s = 2\,\mathrm{m}$, and the over-approximation of the accumulated round-off error was set to $\bar{\epsilon} = 10^{-6}$. These parameters were selected to ensure the numerical stability of the algorithms, and maintain the percentage of samples in Figure 3.12b above 50%. Figure 3.12b indicates that 100% of the collision queries returned no collision. This is expected as the probability of randomly sampling two Bézier curves that satisfy $d(\boldsymbol{p_d^1}, \boldsymbol{p_d^2}) < \bar{\epsilon}$ is very low.

Each data point in Figure 3.12a shows the average runtime for a batch of $10,000$ pairs of Bézier curves. The lower and upper dashed lines represent the 5 and 95 percentile curves, respectively. Notice the logarithmic scale of the ordinate in Figure 3.12a. Again, the conclusions inferred for the different variations of the GJK algorithm in Figure 3.10 also apply to Figure 3.12. Notice also that, in general, queries between polytopes are faster than for a polytope and a Bézier curve, which in turn are also faster than queries between a pair of Bézier curves. The reason behind these results is not only that de Casteljau requires $O(n^2)$ arithmetic operations, but more importantly that queries involving Bézier curves are no longer a convex problem, and the associated algorithms require the application of branch and bound strategies, as shown in Algorithms 1 and 2.



(a) All samples

(b) Ratio of proximity queries deemed safe

(c) Samples that returned $v_t = true$

(d) Samples that returned $v_t = false$

Figure 3.12: Performance results for proximity queries between Bézier curves.

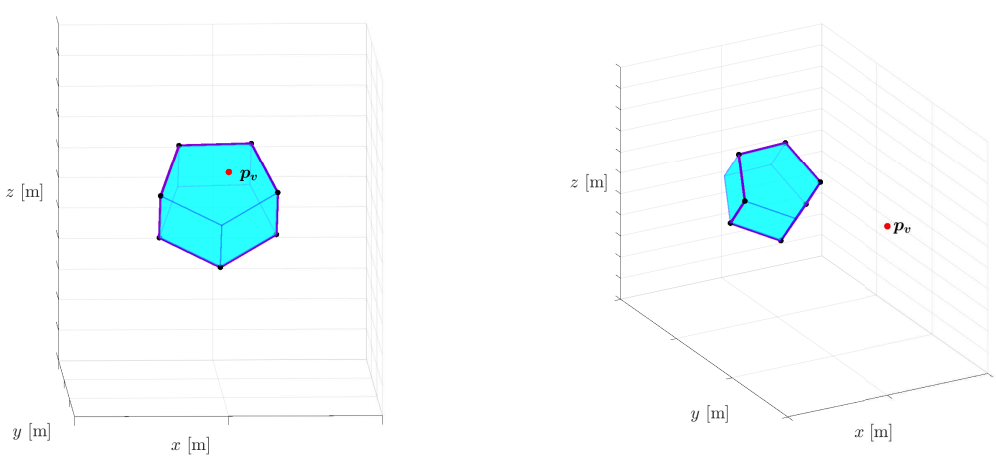
Chapter 4

Planning in Cluttered Environments

This chapter proposes the silhouette of an obstacle as an attention mechanism to increase the probability of generating paths through the narrow passages in a cluttered environment. In addition, the chapter presents a sampling-based motion-planning algorithm that utilizes silhouette information, as well as path-smoothing and temporal assignment methods to generate cooperative trajectories that satisfy the constraints specified in Chapter 2.

4.1 Silhouette as an Attention Mechanism

As discussed in Section 3.3, the silhouette captures meaningful geometric information that can be used to avoid a particular obstacle. Figure 4.1 shows a convex obstacle $\mathcal{O}$ in cyan, and its silhouette in purple as seen from a particular point of view $\boldsymbol{p_v}$. The results in Figure 4.1 can be computed using the three algorithms introduced in Section 3.3 for the three different boundary representations of a polytope described in Section 3.1.1.



(a) 3D view 1

(b) 3D view 2

Figure 4.1: Silhouette of an obstacle $\mathcal{O}$ as seen from point of view $\boldsymbol{p_v}$.

To leverage the silhouette for motion planning, an area of interest around the obstacle is defined. This area is referred to as the *expanded silhouette*, and is highlighted in purple in Figure 4.2. To compute this surface the vertices of the silhouette are expanded outwards. The coordinates of the inner vertices, generated through the expansion of edge $e_{i,j} = \{v_i, v_j\}$ in the silhouette, are

$$\bar{\boldsymbol{v}}_{\boldsymbol{I},\boldsymbol{i}} := \bar{\boldsymbol{v}}_{\boldsymbol{i}} + d_s \boldsymbol{n}_{\boldsymbol{e}_{\boldsymbol{i},\boldsymbol{j}}}, \qquad \text{and} \qquad \bar{\boldsymbol{v}}_{\boldsymbol{I},\boldsymbol{j}} := \bar{\boldsymbol{v}}_{\boldsymbol{j}} + d_s \boldsymbol{n}_{\boldsymbol{e}_{\boldsymbol{i},\boldsymbol{j}}},$$

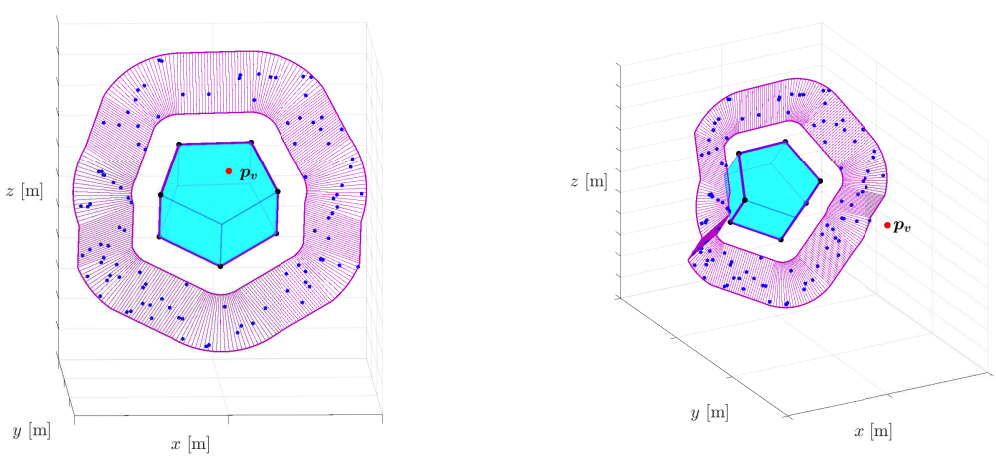
whereas the coordinates of the outer vertices associated to edge $e_{i,j}$ are

$$\bar{\boldsymbol{v}}_{\boldsymbol{O},\boldsymbol{i}} := \bar{\boldsymbol{v}}_{\boldsymbol{i}} + d_e \boldsymbol{n}_{\boldsymbol{e}_{\boldsymbol{i},\boldsymbol{j}}}, \qquad \text{and} \qquad \bar{\boldsymbol{v}}_{\boldsymbol{O},\boldsymbol{j}} := \bar{\boldsymbol{v}}_{\boldsymbol{j}} + d_e \boldsymbol{n}_{\boldsymbol{e}_{\boldsymbol{i},\boldsymbol{j}}},$$

where $\bar{\boldsymbol{v}}_{\boldsymbol{i}}$ and $\bar{\boldsymbol{v}}_{\boldsymbol{j}}$ are the coordinates of vertices v_i and v_j; d_s is the safe separation distance; $d_e > d_s$ is the outer expansion distance, which is a tuning parameter; $\boldsymbol{n}_{\boldsymbol{e}_{\boldsymbol{i},\boldsymbol{j}}}$ is a unit vector defined as

$$\boldsymbol{n}_{\boldsymbol{e}_{\boldsymbol{i},\boldsymbol{j}}} := \frac{(\bar{\boldsymbol{v}}_{\boldsymbol{i}} - \boldsymbol{p}_{\boldsymbol{v}}) \times (\bar{\boldsymbol{v}}_{\boldsymbol{j}} - \bar{\boldsymbol{v}}_{\boldsymbol{i}})}{\|(\bar{\boldsymbol{v}}_{\boldsymbol{i}} - \boldsymbol{p}_{\boldsymbol{v}}) \times (\bar{\boldsymbol{v}}_{\boldsymbol{j}} - \bar{\boldsymbol{v}}_{\boldsymbol{i}})\|};$$

and the sequence of edges $e_{i,j}$, $e_{j,k}$, ..., $e_{\ell,i}$ that define the silhouette traverse it counterclockwise as observed from $\boldsymbol{p}_{\boldsymbol{v}}$. The idea is that the path-planning effort will focus its attention on the expanded silhouette to design paths that get around the obstacle. This concentrates the search for candidate paths near the narrow passages that exist in the surroundings of obstacles in cluttered scenarios.



(a) 3D view 1

(b) 3D View 2

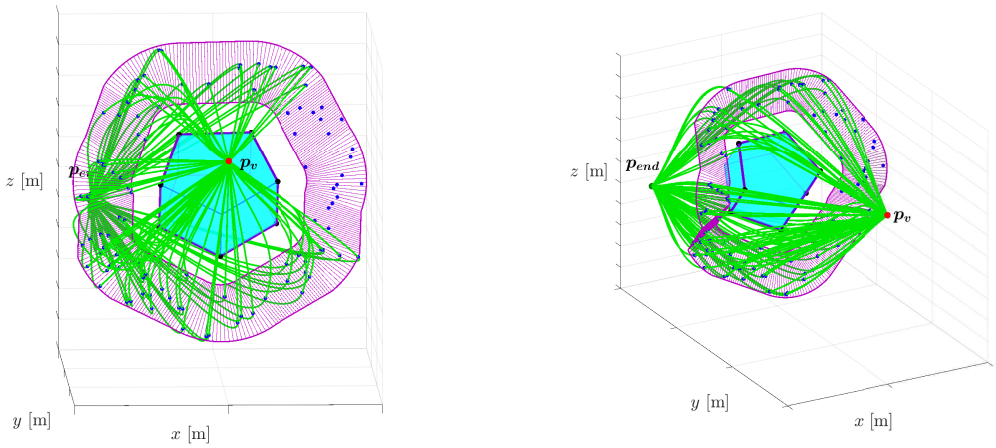
Figure 4.2: Expanded silhouette and randomly sampled points.

Once the expanded silhouette is constructed, the motion-planning algorithm can sample points on this surface, as depicted in Figure 4.2. Like a human would when trying to avoid an obstacle, these points focus the attention in the vicinity of the obstacle in conflict, yet not so close as to render all candidate paths unsafe. The narrow passages that occur in the neighborhood of obstacles in cluttered environments are particularly difficult to explore for existing sampling-based motion-planning algorithms. This attention mechanism increases the probability of sampling within those narrow passages. Unlike existing sample-based methods [15, 10, 41, 42], points sampled using the silhouette inherently incorporate additional information obtained from the analysis of the local geometry of the obstacle.

Finally, to build a set of paths around the obstacle, we define a goal point $\boldsymbol{p_{end}}$, depicted by a green marker in Figure 4.3. The design of a silhouette-informed path around this obstacle is divided into two steps:

i) A first path segment from $\boldsymbol{p_v}$ to each of the sampled points $\boldsymbol{p_{rand}}$ on the expanded silhouette is sketched. Note that the expanded silhouette is within line of sight of $\boldsymbol{p_v}$. Therefore, it should be relatively simple to design a path that maintains safe separation with the obstacle.

ii) A second path segment from each $\boldsymbol{p_{rand}}$ to $\boldsymbol{p_{end}}$ is designed. However, $\boldsymbol{p_{end}}$ may not be within line of sight of each $\boldsymbol{p_{rand}}$. If the second path segment does not keep a safe separation with the obstacle, one can recompute the silhouette using $\boldsymbol{p_{rand}}$ as the new point of view.



(a) 3D view 1

(b) 3D view 2

Figure 4.3: Safe paths generated around an obstacle using silhouette information.

Thus, adding additional path segments as needed until a safe path that ends in $\boldsymbol{p_{end}}$ is generated. Figure 4.3 only shows paths that were designed in a single iteration, without recomputing the silhouette.

Algorithm 1 in Appendix A is used to discard unsafe path segments, ensuring that all curves in Figure 4.3 maintain a safe separation with the obstacle. However, computing a path around a single obstacle is much simpler than through a cluttered environment. The next section proposes an algorithm that leverages the idea presented here to generate paths through complex scenarios.

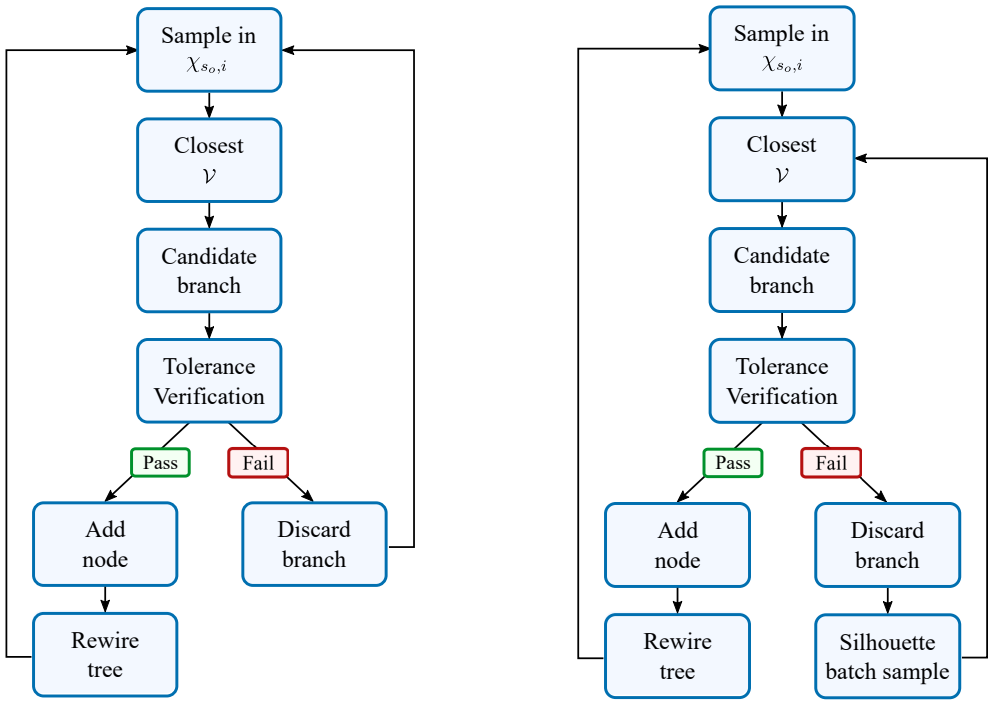
4.2 Path Planning

The path-planning method is composed of a silhouette-informed sample-based motion-planning algorithm and a path-smoothing algorithm. The motion-planning algorithm produces a set of line segments that connect the initial position of each vehicle with their goal position. Then, the path-smoothing algorithm uses this sequence of line segments to produce a set of $\mathcal{G}^2$ continuous Bézier curves that meet the specifications in Definition 4, Chapter 2. The sample-based motion-planning algorithm could have been designed to explore the workspace using $\mathcal{G}^2$ continuous Bézier curves straightaway. However, this was decided against based on the runtime performance results presented in Section 3.4. Recall that proximity queries between a pair of polytopes, such as a line segment and an obstacle, are less computationally expensive than between a polytope and a Bézier curve. In the path-planning phase all vehicles compute their paths independently of their cooperating peers. Once their paths have been computed, they share this information with the cooperating UASs to design speed profiles that guarantee safe separation among the cooperating agents, thus extending the idea of decoupling space and time in [31] to the trajectory generation phase.

4.2.1 Silhouette-Informed Trees (SIT)

Algorithm 5 in Appendix A contains the pseudo-code of the proposed silhouette-informed sample-based motion-planning method. Note that Algorithm 5 is similar to RRT* described in [10]. Indeed, lines 13 through 34 are identical to the corresponding lines in [10]. Like RRT*, Algorithm 5 builds a tree $\mathcal{T} = (\mathcal{V}, \mathcal{E})$, where its vertices $\mathcal{V}$ and edges $\mathcal{E}$ maintain a safe separation with all obstacles in the

workspace. The algorithm differs from RRT* in that it uses silhouette information to sample around an obstacle when a candidate branch fails a tolerance verification test, see Figure 4.4. Note also that in Figure 4.4 collision queries have been replaced by tolerance verification queries to consider the uncertainty in the geometric model described in Section 3.1, as well as the dimensions of the vehicle frames in this TBO framework. Henceforth, Algorithm 5 is referred to as the Silhouette-Informed Trees (SIT). To provide a coherent structure, the analysis of SIT has been divided into four subsections: sampling logic, tolerance verification, connections along a minimum-cost path, and tree rewiring.



(a) RRT*

(b) SIT

Figure 4.4: Comparison of the schematics of of RRT* and SIT.

Sampling logic

The first difference between Algorithm 5 and RRT* is the definition of a sampling-mode flag s_m in line 3. This flag determines the sampling approach that will be used in each iteration of SIT:

i) If $s_m = \mathit{true}$, then a batch of samples is picked from the expanded silhouette of an obstacle that violated safe separation constrains with the previous candidate branch.

ii) If $s_m = \mathit{false}$, then SIT samples the configuration space as RRT* would. The function sample in line 9 may also contain some of the sample-biasing methods proposed in [89, 13], as well as other heuristics with the potential to reduce the number of samples required to reach $\boldsymbol{p_{goal}}$. In this thesis, the sampling was biased towards $\boldsymbol{p_{goal}}$, as suggested in [90, 13], to steer the tree in that direction.

Note that only lines 35 and 37 in Algorithm 5 modify the value of the sampling-mode flag. This is designed to ensure that the algorithm does not focus all the attention on a cul-de-sac. It also intends to balance the narrow passage search—provided by the silhouette—with the exploration of the configuration space that RRT* performs. Additional fields can be included in the tree nodes $\mathcal{V}$ to limit the number of times a node is used as the point of view to compute the silhouette. Lines 11 and 12 contain the steps that let RRT* build the tree incrementally. The function nearest returns the node in $\mathcal{T}$ that is closest to $\boldsymbol{p_{rand}}$, while steer$(\boldsymbol{x}, \boldsymbol{y})$ returns the point

$$\boldsymbol{p_{new}} = \begin{cases} \boldsymbol{y}\,, & \text{if } \|\boldsymbol{x} - \boldsymbol{y}\| \leq \eta \\ \operatorname{argmin}_{\boldsymbol{z} \in \mathcal{B}^2_{\boldsymbol{x},\eta}} \|\boldsymbol{z} - \boldsymbol{y}\|\,, & \text{otherwise} \end{cases},$$

where $\mathcal{B}^2_{\boldsymbol{x},\eta}$ is a ball centered at $\boldsymbol{x}$ of radius η defined using the 2-norm, and η is the *steering distance*, which is a tuning parameter.

Tolerance verification

Once a candidate branch $\{\boldsymbol{p_{nearest}}, \boldsymbol{p_{new}}\}$ has been generated, the function toleranceVerification evaluates whether the candidate branch lies in $\mathcal{X}_{s_o,i}$, where i denotes the index of the vehicle for which the path is being generated. It loops through the different obstacles in the workspace and utilizes the modification of the GJK algorithm for tolerance verification, GJKTolerance in Section 3.2. If the function detects a conflict, it returns $v_t = \mathit{false}$, and a pointer to the corresponding obstacle id_c. If $\{\boldsymbol{p_{nearest}}, \boldsymbol{p_{new}}\} \in \mathcal{X}_{s_o,i}$, then toleranceVerification returns $v_t = \mathit{true}$, and $id_c = \mathit{nil}$.

Connections along a minimum-cost path

As described in [10], the construction of the tree considers possible connections with all the neighboring vertices $\mathcal{P}_{near}$ within a ball of radius r centered around $\boldsymbol{p_{new}}$. The value of r is computed

at each iteration through the function radius that implements the following expression:

$$r = \min\left\{\gamma_{RRT^*}\left(\frac{\log\left(\mathrm{card}(\mathcal{V})\right)}{\mathrm{card}(\mathcal{V})}\right)^{\frac{1}{d}}, \eta\right\},$$

where γ_{RRT^*} is a tuning parameter, and $d = 3$ is the dimension of the configuration space. The vertex with the smallest cost $\boldsymbol{p_{min}}$ is initialized with the nearest vertex $\boldsymbol{p_{nearest}}$ in line 18, where the function cost returns the cost associated with the input vertex, and the function costLine returns the cost associated with the line segments that connects the input vertices. Notice that only the edge in $\mathcal{X}_{s_o,i}$ with the minimum cost is added to the tree in lines 19 through 25. The cost function used in this thesis is the arclength.

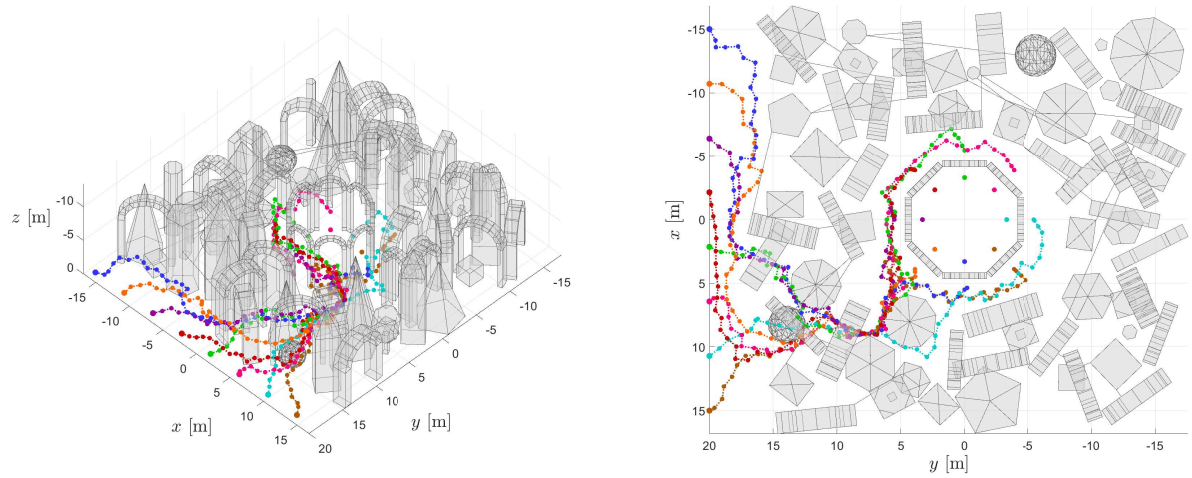
Tree rewiring

Finally, if one can create an edge from $\boldsymbol{p_{new}}$ to some $\boldsymbol{p_{near}} \in \mathcal{P}_{near}$ with a lower cost than the original path to $\boldsymbol{p_{near}}$, then $\boldsymbol{p_{near}}$ is disconnected from its parent node and replaced by the new edge $\{\boldsymbol{p_{new}}, \boldsymbol{p_{near}}\}$. To this end, the function parent in line 31 extracts the original parent node of $\boldsymbol{p_{near}}$. This step is necessary to ensure asymptotic convergence to the optimal solution.

When a solution is found, SIT returns an ordered set of m_i points $\mathcal{P}_i := \{\boldsymbol{p_{1,i}}, \ldots, \boldsymbol{p_{m_i,i}}\}$ that connect the initial and goal points, with $\boldsymbol{p_{1,i}} = \boldsymbol{p_{d_{init},i}}$, $\boldsymbol{p_{m_i,i}} = \boldsymbol{p_{d_{end},i}}$, and $i \in \mathcal{I}$. The line segments defined by consecutive points in $\mathcal{P}_i$ maintain safe separation constraints with all the obstacles in the workspace. Figure 4.5 shows the set of ordered points $\mathcal{P}_i$ for a group of eight agents that are tasked to fly through an urban-like environment. This particular cluttered scenario contains a variety of polyhedral obstacles such as prisms, arches, tree-like objects, icosahedrons, as well as line segments that simulate cables. The fleet is asked to fly from the periphery of the environment to the central plaza. For this example, the tuning parameters within SIT are set to

$$\eta = 1.00\,\mathrm{m}, \qquad n_{sil} = 25, \qquad n_{\max} = 5500, \qquad \gamma_{RRT^*} = 1.5, \qquad d_s = 1.00\,\mathrm{m}, \qquad d_e = 1.50\,\mathrm{m},$$

and the cost function selected is the arclength. The following section explores how to process and smooth this solution.



(a) 3D view

(b) Top view

Figure 4.5: Ordered set of points $\mathcal{P}_i$ returned by SIT for each UAS, $i \in \mathcal{I}$.

4.2.2 Path Smoothing

As illustrated in Figure 4.5, in cluttered scenarios sampling-based methods are often given a computational budget that is insufficient to make the solution converge near the optimal. In this case, the solution provided can be post-processed to improve its quality and cost based on some knowledge of the system. In this thesis, the proposed path-smoothing algorithm is divided into two steps:

i) An edge-reduction technique is implemented with two objectives. First, reduce the arclength of the solution proposed by SIT, if possible. Second, reduce the number of line segments that safely connect the initial and end points assigned to the ith UAS[1]. For brevity, we drop the subscript i from the ordered set $\mathcal{P}_i$ and all the points within, and refer to the set and points returned by SIT as $\mathcal{P} := \{\boldsymbol{p_1}, \ldots, \boldsymbol{p_m}\}$. Define now the variables $\omega_{j,k}$ associated with points $\boldsymbol{p_j}$ and $\boldsymbol{p_k}$ with $j \in \{1, \ldots, m-1\}$ and $k \in \{j+1, \ldots, m\}$, as well as the arclength of the associated line segment $\ell_{j,k} := \|\boldsymbol{p_k} - \boldsymbol{p_j}\|$. Then, define $\mathcal{K}_\ell$ as the set of indices $k > \ell$ that satisfy $\{\boldsymbol{p_\ell}, \boldsymbol{p_k}\} \in \mathcal{X}_{s_o,i}$ for some $\ell < m$, where $\{\boldsymbol{p_\ell}, \boldsymbol{p_k}\}$ represents the line segment with endpoints $\boldsymbol{p_\ell}$ and $\boldsymbol{p_k}$. Similarly, define $\mathcal{J}_\ell$ as the set of indices $j < \ell$ that satisfy $\{\boldsymbol{p_j}, \boldsymbol{p_\ell}\} \in \mathcal{X}_{s_o,i}$ for some $\ell > 1$. Then, the edge-reduction problem can be formulated as a

[1]Later, this step will lower the computational burden of the algorithm that the cooperating fleet uses to design their speed profiles.

mixed integer linear programming problem with cost function

$$\min \sum_{j=1}^{m-1} \sum_{k \in \mathcal{K}_j} \omega_{j,k}\, \ell_{j,k},$$

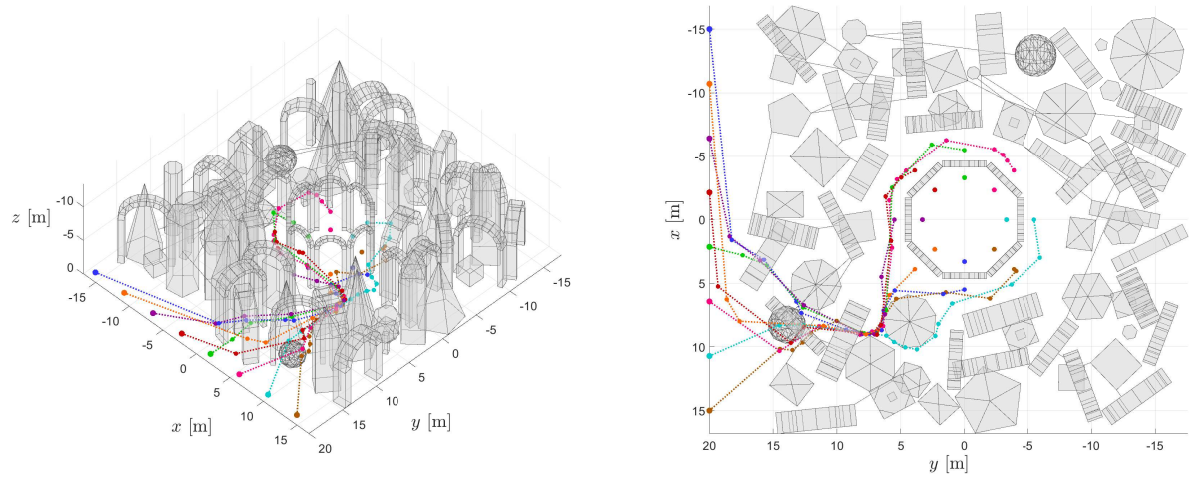
subject to linear constraints

$$\begin{aligned} \sum_{k \in \mathcal{K}_1} \omega_{1,k} &= 1, \\ \sum_{j \in \mathcal{J}_m} \omega_{j,m} &= 1, \\ \sum_{j \in \mathcal{J}_\ell} \omega_{j,\ell} - \sum_{k \in \mathcal{K}_\ell} \omega_{\ell,k} &= 0, \qquad \forall \ell \in \{2, \ldots, m-1\}, \end{aligned}$$

and integer constraints $\omega_{j,k} \in \{0, 1\}$. Fortunately, the constraint matrix of the linear portion of this problem is totally unimodular [91]. Consequently, the integer constraints can be dropped, and the problem above can be solved as a *linear programming problem*. This returns the reduced ordered set $\mathcal{P}_{r,i} := \{\boldsymbol{p_{1,i}}, \ldots, \boldsymbol{p_{p_i,i}}\}$ where $p_i \leq m_i$ is the total number of points, $\boldsymbol{p_{1,i}} = \boldsymbol{p_{d_{init},i}}$, $\boldsymbol{p_{p_i,i}} = \boldsymbol{p_{d_{end},i}}$, and the total arclength associated with $\mathcal{P}_{r,i}$ is smaller or equal than that of $\mathcal{P}_i$, with $i \in \mathcal{I}$. Additional post-processing methods could be applied to control other characteristics of these points, such as height, while maintaining a safe separation with all the obstacles.

Figure 4.6 illustrates the results of the edge-reduction algorithm when applied to the urban-like environment introduced in Figure 4.5. As expected, the number of points that connect the initial and goal points for each UAS has decreased significantly, and the set $\mathcal{P}_{r,i}$ defines a shorter path than the set $\mathcal{P}_i$ for all $i \in \mathcal{I}$.

ii) An extension of an efficient cornering motion algorithm [92, 93], originally designed for Computerized Numerical Control (CNC) machines, is used in this thesis to smooth the path defined by $\mathcal{P}_{r,i}$ for all the UASs in the fleet. The result, is a sequence of $\mathcal{G}^2$ continuous *Pythagorean-Hodograph* (PH) Bézier curves. The *hodograph* of a curve $\boldsymbol{p_d}(\zeta)$ is defined as its parametric derivative $\boldsymbol{p'_d}(\zeta)$. Hence, a spatial PH Bézier curve of degree n defines a map



(a) 3D view

(b) Top view

Figure 4.6: Ordered set of points $\mathcal{P}_{r,i}$ returned by the edge-reduction method for each UAS.

$\boldsymbol{p_d}(\zeta) : [0,1] \to \mathbb{R}^3$, whose hodograph satisfies the *Pythagorean condition*

$$\frac{\mathrm{d}\boldsymbol{p_d}^\top(\zeta)}{\mathrm{d}\zeta}\frac{\mathrm{d}\boldsymbol{p_d}(\zeta)}{\mathrm{d}\zeta} = \sigma^2(\zeta),$$

where $\sigma(\zeta) : [0,1] \to \mathbb{R}$ is the parametric speed of $\boldsymbol{p_d}(\zeta)$, and can be expressed in a Bernstein basis of degree $n-1$ with control points $\bar{\sigma}_k$

$$\sigma(\zeta) = \sum_{k=0}^{n-1} \bar{\sigma}_k b_k^{n-1}(\zeta).$$

The integration of the parametric speed yields the arc length

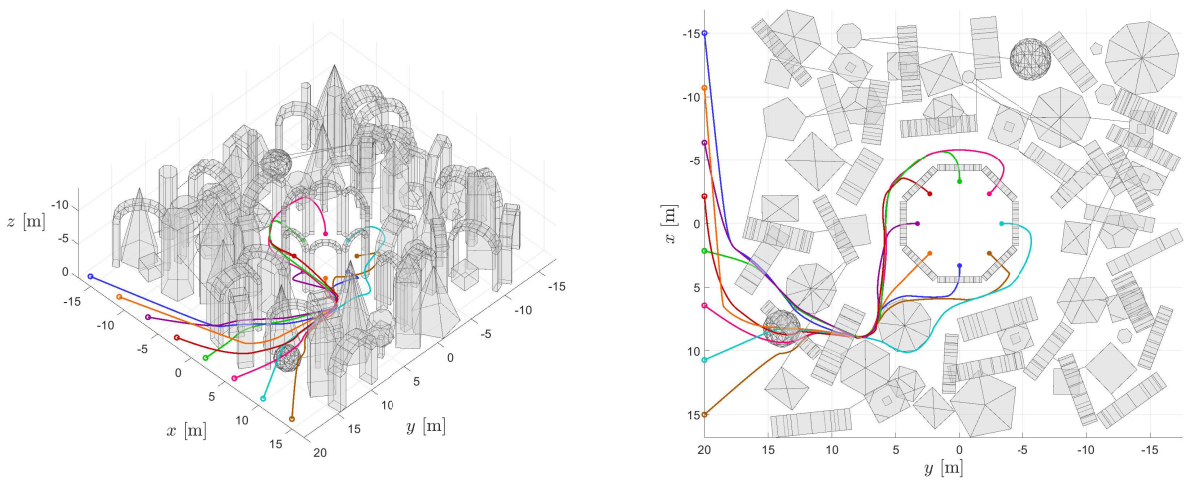
$$\ell_d(\zeta) = \int_0^\zeta \sigma(\epsilon)\,\mathrm{d}\epsilon = \sum_{k=0}^{n} \bar{\ell}_{d,k} b_k^n(\zeta)\,,$$

where $\bar{\ell}_{d,0} = 0$ and $\bar{\ell}_{d,k} = \frac{1}{n}\sum_{j=0}^{k-1}\bar{\sigma}_j$. Thus, the total length of a PH Bézier curve is

$$L = \frac{1}{n}\sum_{k=0}^{n-1}\bar{\sigma}_k. \tag{4.1}$$

PH Bézier curves were chosen as the solution for the smoothing method for three main reasons.

First, as highlighted in Equation (4.1), their arclength has a closed-form expression. Since the arclength is the cost function considered to optimize path smoothing, PH Bézier curves avoid approximate numerical integration, decreasing the computational load and increasing the accuracy of the arclength computations. Second, curve framing is used to define the long-track and cross-track path-following errors, which are used by the underlying speed-tracking algorithms in the UASs. PH Bézier curves yield exact rotation-minimizing frames[2] that can be obtained through the integration of a rational function [95], and hence facilitate the computation of the different components of the path-following error. Last, PH Bézier curves yield a polynomial expression for the left-hand side in Equation (2.1), which simplifies, to some extent, the reparameterization of the path as a function of the time t_d.



(a) 3D view

(b) Top view

Figure 4.7: $\mathcal{G}^2$ continuous piecewise Bézier curves $\boldsymbol{p_{d,i}}(\zeta_i)$ returned by the smoothing method.

Figure 4.7 shows the paths assigned to the cooperating fleet tasked to fly through the urban-like environment. Notice that the curves are smooth, and maintain a safe separation with all the obstacles in the environment. To this end, the method uses the tolerance verification algorithm proposed in Section 3.2.1. The result in Figure 4.7 often alternates PH Bézier curves of degree 7 and degree 1—line segments—to smooth $\mathcal{P}_{r,i}$. This occurs when the algorithm identifies that placing a line segment between two Bézier curves of degree 7 yields a shorter

[2]These curve frames are well defined at inflection points, as opposed to Frenet-Serret frames. The frame used in the speed-tracking controllers of the UASs in this thesis is the Rotation Minimizing Osculating Frame (RMOF) [94].

path. At the contact points, these curves have the same tangent direction, zero curvature, and satisfy the additional geometric continuity constraints derived in Section 2.2.1.

4.3 Speed Assignment

Once vehicles have computed their piecewise-defined spatial paths $\boldsymbol{p_{d,i}}(\zeta_i)$, agents share their geometric description with their cooperating peers. The fleet uses this information to compute speed profiles that guarantee safe separation among all agents. To solve the speed-assignment problem, this thesis uses a modification of RRT to generate a tree $\mathcal{T}_v = (\mathcal{V}_v, \mathcal{E}_v)$ with vertices $\mathcal{V}_v$ and edges $\mathcal{E}_v$[3]. Each node contains n speed profiles, one for each vehicle participating in the cooperative effort. To this end, the algorithm samples in the space of polynomial functions of degree 4 to design the speed profile $v^j_{d,i}(t_d)$ associated with jth path segment $\boldsymbol{p^j_{d,i}}(\zeta^j_i)$ of the ith vehicle that meets the dynamic constraints in Equation (2.7). To check that the speed profile meets the desired dynamic constraints, the algorithm leverages the properties of Bézier curves introduced in Section 3.1.2. The speed profiles associated with the first and last path segments of each UAS are designed to meet the boundary constraints in Equation (2.3), whereas intermediate speed profiles are designed to meet the $\mathcal{C}^2$ continuity constraints in Equation (2.4). In addition, the reparameterization of the path as a function of time in Equation (2.1) defines a surface with two independent variables ζ^j_i and t_d for each path segment. The intersection with the zero plane renders the relationship between the ζ^j_i and t_d. This is solved using polynomial approximation methods [78], and yields an estimate of the evolution of the curve parameter $\hat{\zeta}^j_i$ as a function of the time t_d. The approximation method stops when the difference between the actual and the estimated arclength across the entire path segment is below a predefined threshold δ_ℓ, that is

$$\int_{t^j_{d_{init},i}}^{t_d} v^j_{d,i}(\tau)\,\mathrm{d}\tau - \ell^j_{d,i}(\hat{\zeta}^j_i(t_d)) \leq \delta_\ell.$$

This is used to ensure safe separation constraints among all the trajectory segments in a node, as well as with all the trajectory segments in parent nodes

[3]RRT is not asymptotically optimal, and hence this step does not contemplate a cost function. However, reducing flight time is often a desired mission objective. As a result, the proposed method sometimes yields long mission times, and should be modified in future work. Suitable extensions of RRT* or RRT*-Connect [96] could be useful for this purpose.

$$\|\boldsymbol{p}^{\boldsymbol{j_1}}_{\boldsymbol{d,i_1}}(\hat{\zeta}^{j_1}_{i_1}(t_d)) - \boldsymbol{p}^{\boldsymbol{j_2}}_{\boldsymbol{d,i_2}}(\hat{\zeta}^{j_2}_{i_2}(t_d))\| \geq d_s(\boldsymbol{p_{d,i_1}}, \boldsymbol{p_{d,i_2}}) + 2\delta_\ell, \qquad \forall i_1 \neq i_2 \in \mathcal{I}. \tag{4.2}$$

The current formulation and implementation of the speed-assignment algorithm is centralized. Again, the tolerance verification queries developed in Section 3.2.1 are used to discard candidate branches that do not meet the condition in Equation (4.2). This step also relies on temporal logic to avoid applying proximity queries on trajectory segments that do not overlap in time. Similarly, the algorithm also analyzes which path segments are separated in space, to avoid checking whether they are separated in time and reduce the computational load of the algorithm.

Figure 4.8 shows the speed profile designed by the speed-assignment algorithm for the urban-like environment introduced in Figure 4.5. Here, the speed and acceleration limits are set to

$$v_{d_{\min},i} = 0.00\,\mathrm{m/s}, \qquad v_{d_{\max},i} = 1.00\,\mathrm{m/s}, \qquad a_{d_{\min},i} = -2.00\,\mathrm{m/s^2}, \qquad a_{d_{\max},i} = 2.00\,\mathrm{m/s^2},$$

for all $i \in \mathcal{I}$. The safe separation distance among vehicles is set to $d_s = 0.50\,\mathrm{m}$ for all pairs of vehicles. The initial and final conditions are designed for a smooth departure and arrival

$$v_{d_{init},i} = 0.00\,\mathrm{m/s}, \qquad v_{d_{end},i} = 0.00\,\mathrm{m/s}, \qquad a_{d_{init},i} = 0.00\,\mathrm{m/s^2}, \qquad a_{d_{end},i} = 0.00\,\mathrm{m/s^2},$$

for all $i \in \mathcal{I}$. The starting mission time is $t_{d_{init},i} = 0.00\,\mathrm{s}$ for all $i \in \mathcal{I}$, whereas the arrival time to the center plaza for each agent is decided by the speed-assignment algorithm. Figure 4.9 illustrates the planned distance among the centers of mass of the cooperating vehicles. Note that the distance is greater than d_s at all times, which is highlighted by a red dashed line.

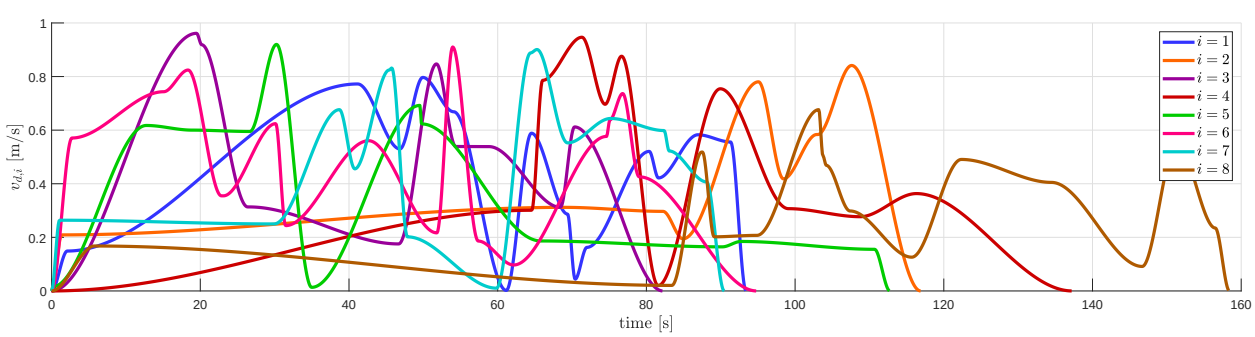


Figure 4.8: Speed profiles.

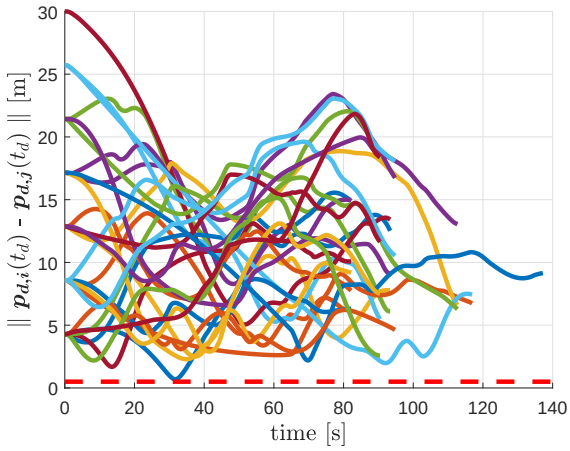


Figure 4.9: Planned distance among vehicles.

Chapter 5

Tight Coordination Protocols

This chapter proposes a distributed time-critical control law for tight coordination under a variety of temporal constraints. Steady-state and transient performance guarantees are derived assuming perfect virtual-target-tracking capabilities. These results are theoretical, and will be leveraged in Chapter 6 to infer performance guarantees for a fleet of heterogeneous vehicles with realistic speed-tracking controllers. To illustrate and corroborate the performance of these protocols under ideal target-tracking conditions, simulation results for a group of eight cooperating peers are provided for unenforced, relaxed, and strict temporal specifications.

5.1 Distributed Coordination Control Law

To solve the multi-objective problem for tight coordination defined in Equation (2.15), the following distributed protocol is proposed for the link peers:

$$u_{c,i}(t) = -k_P \sum_{j \in \mathcal{N}_i} (\xi_i(t) - \xi_j(t)) - k_R\, \omega_{R_i} (\xi_i(t) - \xi_R(t)) + \dot{\xi}_R, \qquad i \in \mathcal{I}_\ell, \tag{5.1}$$

while the control law for the end peers is

$$\begin{aligned} u_{c,i}(t) &= -k_P \sum_{j \in \mathcal{N}_i} (\xi_i(t) - \xi_j(t)) + \chi_i(t), && i \in \mathcal{I}_e, \\ \dot{\chi}_i(t) &= -k_I \sum_{j \in \mathcal{N}_i} (\xi_i(t) - \xi_j(t)), \qquad \chi_i(0) = \chi_{i_0}, && i \in \mathcal{I}_e, \end{aligned} \tag{5.2}$$

where k_R, k_P, and k_I are control gains, ω_{R_i} is a link weight that implements the different types of temporal constraints, and $\chi_i(t)$ is an integral state responsible for learning the mission rate $\dot{\xi}_R$. As shown in Table 5.1, for unenforced or strict temporal constraints ω_{R_i} are identically equal to 0

Table 5.1: Link-weight logic under different types of temporal constraints.

Unenforced	**Relaxed**	**Strict**
$\omega_{R_i}(t) \equiv 0$	$\omega_{R_i}(t) = \begin{cases} 1, & \text{if } \lvert\xi_i(t) - \xi_R(t)\rvert \geq \Delta_t(t) \ \wedge \ t - t_s^i > \tau_{R_0} \\ 0, & \text{if } \lvert\xi_i(t) - \xi_R(t)\rvert < \Delta_t(t) \ \wedge \ t - t_s^i > \tau_{R_1} \\ \omega_{R_i}^+(t_s^i), & \text{otherwise} \end{cases}$	$\omega_{R_i}(t) \equiv 1$

or 1, respectively, thus permanently "ignoring" or "listening" to the information provided by the reference agent. However, for relaxed temporal constrains Table 5.1 defines a state-dependent switching logic that selectively "listens" to the reference agent. In this context, t_s^i denotes the last time the ith link peer switched the value of ω_{R_i}, and $\omega_{R_i}^+(t_s^i)$ is the limit from the right at t_s^i, as shown in Figure 5.1. To avoid Zeno behavior, changes in the link-weight values are subject to slow switching constraints, where dwell times τ_{R_0} and τ_{R_1} define the minimum times ω_{R_i} will be set to 0 or 1, respectively.

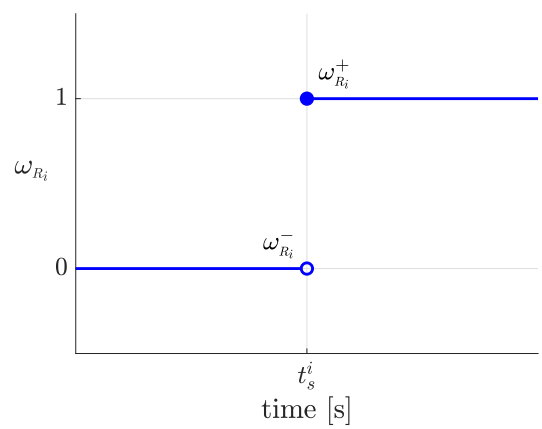


Figure 5.1: Link-weight switch.

For brevity, the control law in (5.1) and (5.2) can be rewritten in matrix form as

$$\begin{aligned} \boldsymbol{u_c}(t) &= -k_P \mathbf{L}(t)\,\boldsymbol{\xi}(t) - k_R \boldsymbol{\Omega}(t)\left(\boldsymbol{\xi}(t) - \xi_R(t)\mathbf{1_n}\right) + \begin{bmatrix} \rho \mathbf{1_{n_\ell}} \\ \boldsymbol{\chi}(t) \end{bmatrix}, \\ \dot{\boldsymbol{\chi}}(t) &= -k_I \mathbf{C_e}^\top \mathbf{L}(t)\,\boldsymbol{\xi}(t), \end{aligned} \tag{5.3}$$

where $\boldsymbol{u_c}(t)$, $\boldsymbol{\xi}(t)$, $\boldsymbol{\chi}(t)$, and matrices $\boldsymbol{\Omega}(t)$, and $\mathbf{C_e}$ are

$$\begin{aligned} \boldsymbol{u_c}(t) &:= [u_{c,1}(t), u_{c,2}(t), \ldots, u_{c,n}(t)]^\top \in \mathbb{R}^n, \\ \boldsymbol{\xi}(t) &:= [\xi_1(t), \xi_2(t), \ldots, \xi_n(t)]^\top \in \mathbb{R}^n, \\ \boldsymbol{\chi}(t) &:= [\chi_{n_\ell+1}(t), \chi_{n_\ell+2}(t), \ldots, \chi_n(t)]^\top \in \mathbb{R}^{n-n_\ell}, \\ \boldsymbol{\Omega}(t) &:= \operatorname{diag}(\omega_{R_1}(t), \omega_{R_2}(t), \ldots, \omega_{R_{n_\ell}}(t), \mathbf{0}) \in \mathbb{R}^{n\times n}, \\ \mathbf{C_e} &:= [\,\mathbf{0} \ \ \mathbb{I}_{\mathbf{n-n_\ell}}\,]^\top \in \mathbb{R}^{n\times n-n_\ell}. \end{aligned}$$

Since the link-weight logic is embedded in $\boldsymbol{\Omega}(t)$, the expression in (5.3) simultaneously encompasses the protocol for the three types of temporal constraints detailed in Table 5.1. In addition, to help interpret $\mathbf{C_e}$, note that the product $\mathbf{C_e}^\top \boldsymbol{\xi}(t)$ returns the coordination states of the end peers.

5.2 Coordination Dynamics

The single integrator dynamics in Equation (2.12) and the control law in (5.3) yield the following closed-loop collective dynamics:

$$\begin{aligned} \dot{\boldsymbol{\xi}}(t) &= -k_P \mathbf{L}\,(t)\,\boldsymbol{\xi}(t) - k_R \boldsymbol{\Omega}(t)\,(\boldsymbol{\xi}(t) - \xi_R(t)\mathbf{1_n}) + \begin{bmatrix} \rho \mathbf{1}_{\mathbf{n}_\ell} \\ \boldsymbol{\chi}(t) \end{bmatrix} + \boldsymbol{u}_{\mathcal{T}_\models}(t), \\ \dot{\boldsymbol{\chi}}(t) &= -k_I \mathbf{C_e}^\top \mathbf{L}\,(t)\,\boldsymbol{\xi}(t), \end{aligned} \tag{5.4}$$

where the collective target-tracking error feedback $\boldsymbol{u}_{\mathcal{T}_\models}(t)$ is defined as follows:

$$\boldsymbol{u}_{\mathcal{T}_\models}(t) := \left[u_{\mathcal{T}_\models,1}, \ldots, u_{\mathcal{T}_\models,n}\right] \in \mathbb{R}^n. \tag{5.5}$$

Then, to analyze the convergence properties of the collective system dynamics, one can reformulate the dynamics in Equation (5.4) into a stabilization problem. To this end, define the collective error state $\boldsymbol{\zeta}(t) := [\zeta_t(t), \boldsymbol{\zeta_c}^\top(t), \boldsymbol{\zeta_r}^\top(t)]^\top$, where the collective temporal, coordination, and rate errors are

$$\begin{aligned} \zeta_t(t) &:= \xi_R(t) - \frac{1}{n}\mathbf{1_n}^\top \boldsymbol{\xi}(t) \in \mathbb{R}\,, \\ \boldsymbol{\zeta_c}(t) &:= \mathbf{Q}\boldsymbol{\xi}(t) \in \mathbb{R}^{n-1}, \\ \boldsymbol{\zeta_r}(t) &:= \boldsymbol{\chi}(t) - \dot{\xi}_R \mathbf{1}_{\mathbf{n}-\mathbf{n}_\ell} \in \mathbb{R}^{n-n_\ell}, \end{aligned} \tag{5.6}$$

respectively. Note that the collective error state is defined such that

$$\boldsymbol{\zeta}(t) = \mathbf{0} \iff \boldsymbol{\xi}(t) = \xi_R(t)\mathbf{1_n} \;\wedge\; \dot{\boldsymbol{\xi}}(t) = \dot{\xi}_R \mathbf{1_n}.$$

Then, the resulting collective error dynamics are

$$\dot{\boldsymbol{\zeta}}(t) = \boldsymbol{A}(t)\boldsymbol{\zeta}(t) + \boldsymbol{B}\boldsymbol{u}_{\mathcal{T}_\models}(t), \qquad \boldsymbol{\zeta}(0) = \boldsymbol{\zeta_0}, \tag{5.7a}$$

$$\boldsymbol{A}(t) := \begin{bmatrix} -\frac{k_R}{n}\tilde{\omega}(t) & \frac{k_R}{n}\boldsymbol{\omega}^\top(t)\,\mathbf{Q}^\top & -\frac{1}{n}\mathbf{1}_{\mathbf{n}-\mathbf{n}_\ell}^\top \\ k_R\mathbf{Q}\,\boldsymbol{\omega}(t) & -k_P\widehat{\mathbf{L}}\,(t) & \mathbf{Q}\mathbf{C}_\mathbf{e} \\ \mathbf{0} & -k_I\mathbf{C}_\mathbf{e}^\top\boldsymbol{\Pi}\,\mathbf{L}\,(t)\,\mathbf{Q}^\top & \mathbf{0} \end{bmatrix}, \qquad \boldsymbol{B} := \begin{bmatrix} -\frac{1}{n}\mathbf{1}_\mathbf{n}^\top \\ \mathbf{Q} \\ \mathbf{0} \end{bmatrix}, \tag{5.7b}$$

where $\boldsymbol{A}(t) \in \mathbb{R}^{(2n-n_\ell)\times(2n-n_\ell)}$, $\boldsymbol{B} \in \mathbb{R}^{(2n-n_\ell)\times n}$, and $\tilde{\omega}(t)$, $\boldsymbol{\omega}(t)$, $\widehat{\mathbf{L}}\,(t)$, and $\boldsymbol{\Pi}$ are given by

$$\begin{aligned} \tilde{\omega}(t) &:= \sum_{i=1}^{n_\ell} \omega_{R_i}(t), \\ \boldsymbol{\omega}(t) &:= [\omega_{R_1}(t), \omega_{R_2}(t) \ldots, \omega_{R_{n_\ell}}(t), \mathbf{0}^\top]^\top \in \mathbb{R}^n, \\ \widehat{\mathbf{L}}\,(t) &:= \mathbf{Q}\left(\mathbf{L}\,(t) + \frac{k_R}{k_P}\,\boldsymbol{\Omega}(t)\right)\mathbf{Q}^\top, \\ \boldsymbol{\Pi} &:= \mathbf{Q}^\top\mathbf{Q} = \mathbb{I}_\mathbf{n} - \frac{1}{n}\mathbf{1}_\mathbf{n}\mathbf{1}_\mathbf{n}^\top. \end{aligned} \tag{5.8}$$

The derivation of (5.7b) relies on the symmetry[1] of the Laplacian matrix, and uses properties $\mathbf{Q}\mathbf{1}_\mathbf{n} = \mathbf{0}$, $\mathbf{Q}\mathbf{Q}^\top = \mathbb{I}_{\mathbf{n}-\mathbf{1}}$, $\mathbf{L}\,(t)\,\boldsymbol{\Pi} = \boldsymbol{\Pi}\,\mathbf{L}\,(t) = \mathbf{L}\,(t)$, and $\mathbf{1}_\mathbf{n}^\top\mathbf{L}\,(t) = \mathbf{L}\,(t)\,\mathbf{1}_\mathbf{n} = \mathbf{0}$. For further details see Appendix B.1. The following sections investigate whether the system dynamics in (5.7) meet the control objectives specified in Equation (2.15).

5.3 Stability with Ideal Target Tracking

For simplicity, all stability analyses within this section assume perfect target-tracking capabilities. Consequently, the individual position errors for all coordinating agents are $\boldsymbol{e}_{\boldsymbol{p},\boldsymbol{i}}(t) \equiv \mathbf{0}$, and the collective long-track target-tracking error feedback in Equation (5.7) is $\boldsymbol{u}_{\boldsymbol{\mathcal{T}}_\models}(t) \equiv \mathbf{0}$. The following subsections address unenforced, relaxed, and strict temporal constraints with tight coordination.

5.3.1 Unenforced Temporal Constraints

The link-weight logic in Table 5.1 for unenforced temporal specifications ($\omega_{R_i}(t) \equiv 0$) decouples the collective coordination and rate errors $\boldsymbol{\zeta}_{\boldsymbol{u}}(t) := [\boldsymbol{\zeta}_{\boldsymbol{c}}^\top(t), \boldsymbol{\zeta}_{\boldsymbol{r}}^\top(t)]^\top$ from the collective temporal

[1]The Laplacian of an undirected graph is symmetric. Hence, this step leverages Assumption 4 in Chapter 2.

error $\zeta_t(t)$ in Equation (5.7), which leads to the following set of dynamics:

$$\dot{\zeta}_t(t) = \boldsymbol{A_t}\boldsymbol{\zeta_u}(t), \qquad \zeta_t(0) = \zeta_{t_0}, \tag{5.9a}$$

$$\dot{\boldsymbol{\zeta}}_{\boldsymbol{u}}(t) = \boldsymbol{A_u}(t)\boldsymbol{\zeta_u}(t), \qquad \boldsymbol{\zeta_u}(0) = \boldsymbol{\zeta_{u_0}}, \tag{5.9b}$$

where $\boldsymbol{A_t}$ and $\boldsymbol{A_u}(t)$ are

$$\boldsymbol{A_t} := \begin{bmatrix} \mathbf{0}^\top & -\frac{\mathbf{1}_{\mathbf{n}-\mathbf{n}_\ell}^\top}{n} \end{bmatrix}, \qquad \boldsymbol{A_u}(t) := \begin{bmatrix} -k_P\bar{\boldsymbol{L}}(t) & \mathbf{QC_e} \\ -k_I\mathbf{C_e}^\top\boldsymbol{\Pi}\,\mathbf{L}(t)\,\mathbf{Q}^\top & \mathbf{0} \end{bmatrix}, \tag{5.9c}$$

and $\bar{\boldsymbol{L}}(t) := \mathbf{QL}(t)\,\mathbf{Q}^\top$. The following theorem leverages algebraic graph and Lyapunov stability theory to prove that (5.9a) is Uniformly Bounded (UB), and the origin of the system dynamics in (5.9b) is Globally Uniformly Exponentially Stable (GUES). This result is an extension of Theorem 6 in [26], and is necessary to draw conclusions for relaxed temporal constraints.

Theorem 1 *Assume ideal target-tracking capabilities $\boldsymbol{u}_{\mathcal{T}_\models}(t) \equiv \mathbf{0}$, and the information flow $\mathcal{G}(t)$ satisfies Assumptions 3 through 5. Then, there exist known control gains k_P and k_I such that*

$$|\xi_i(t) - \xi_R(t)| \leq \kappa_{t,u_1}\|\boldsymbol{\zeta_{u_0}}\|e^{-\lambda_u t} + \kappa_{t,u_2}\|\boldsymbol{\zeta_{u_0}}\| + |\zeta_{t_0}|, \tag{5.10a}$$

$$|\xi_i(t) - \xi_j(t)| \leq \kappa_{c,u}\|\boldsymbol{\zeta_{u_0}}\|\, e^{-\lambda_u t}, \tag{5.10b}$$

$$|\dot{\xi}_i(t) - \rho| \leq \kappa_{r,u}\|\boldsymbol{\zeta_{u_0}}\|\, e^{-\lambda_u t}, \tag{5.10c}$$

for all $t \geq 0$, and all i, $j \in \mathcal{I}$, where κ_{t,u_1}, κ_{t,u_2}, $\kappa_{c,u}$, $\kappa_{r,u}$ are know constants, and λ_u is the guaranteed rate of convergence of the coordination and rate errors

$$\lambda_u := \frac{k_P n\mu}{(1+k_P nT)^2}\left(1 + \eta_{I,u}\frac{n}{n_\ell}\right)^{-1},$$

with design parameter $\eta_{I,u} \geq 2$.

Proof. The proof and expressions for κ_{t,u_1}, κ_{t,u_2}, $\kappa_{c,u}$, $\kappa_{r,u}$ are given in Appendix C.2. This proof is constructive and provides explicit bounds for the control gains k_P and k_I. □

Remark 2 *Equation* (5.10a) *shows that* $|\xi_i(t) - \xi_R(t)|$ *is UB. However, this bound can grow arbitrarily large as the initial errors* $\boldsymbol{\zeta_{u_0}}$ *and* ζ_{t_0} *increase. As expected, this eliminates the possibility of controlling the temporal error through this coordination strategy.*

Remark 3 *The choice of control gains specified in the proof of Theorem 1*

$$k_P > 0, \qquad \frac{k_I}{k_P} = \eta_{I,u}\frac{n}{n_\ell}\lambda_u$$

can be used to determine that the maximum guaranteed rate of convergence

$$\lambda_u^* := \frac{1}{4T}\left(1 + \eta_{I,u}\,\frac{n}{n_\ell}\right)^{-1}$$

is achieved when $k_P = \frac{1}{nT}$*. Notice that* λ_u^* *scales well with the number of agents, as long as the ratio of link to end peers is maintained. However, this choice of* k_P *requires knowing the number of peers. Thus, if agents were to join and leave the cooperative effort inadvertently, then the guaranteed rate of convergence would deviate from the maximum.*

The following section addresses the stability of the system when a specific bound on $|\xi_i(t) - \xi_R(t)|$ must be imposed.

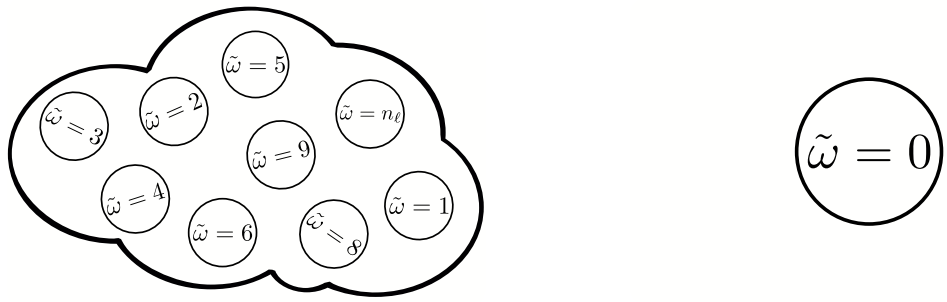
5.3.2 Relaxed Temporal Constraints

To analyze the system under relaxed temporal constraints, the dynamics in Equation (5.7) are separated into two modes, depicted in Figure 5.2. Mode ☁ encompasses all cases where at least one link peer has an active link weight $\omega_{R_i}(t) = 1$, whereas in mode Ø all link weights are 0. As a result, the dynamics in (5.7) can be rewritten as a system that alternates between modes Ø and ☁, with switching signal $\gamma(t) : [0, \infty) \mapsto \mathcal{S}$, a piecewise constant function with a finite number of discontinuities on every bounded time interval, and $\mathcal{S} := \{\emptyset, ☁\}$

$$\dot{\boldsymbol{\zeta}}(t) = \boldsymbol{A}_\gamma(t)\boldsymbol{\zeta}(t), \qquad \boldsymbol{\zeta}(0) = \boldsymbol{\zeta_0}, \qquad \gamma \in \mathcal{S}, \tag{5.11}$$

$$\boldsymbol{A}_{\emptyset}(t) := \begin{bmatrix} 0 & \mathbf{0}^\top & -\frac{1}{n}\mathbf{1}_{\mathbf{n}-\mathbf{n}_\ell}^\top \\ \mathbf{0} & -k_P\bar{\boldsymbol{L}}(t) & \mathbf{Q}\mathbf{C}_\mathbf{e} \\ \mathbf{0} & -k_I\mathbf{C}_\mathbf{e}^\top\boldsymbol{\Pi}\,\mathbf{L}(t)\,\mathbf{Q}^\top & \mathbf{0} \end{bmatrix}, \quad \text{and} \quad \boldsymbol{A}_{\text{☁}}(t) := \boldsymbol{A}(t) \quad \text{with} \quad \tilde{\omega} \geq 1.$$

The discontinuities in $\gamma(t)$ occur at the switching times t_m, with $m \in \{0, 1, \ldots, n_s\}$, and $n_s \in \mathbb{N}$ being the total number of switches.



(a) At least one link peer "listens" to the reference agent, $\tilde{\omega} \geq 1$

(b) No link peers "listen" to the reference agent, $\tilde{\omega} = 0$

Figure 5.2: Modes under relaxed temporal constraints.

Note that in mode Ø the coordination dynamics are the same as for unenforced temporal constraints, since all peers intentionally ignore the reference information. The notation chosen alludes to the fact that mode ☁ encompasses a multitude of cases, whereas in mode Ø the set of link peers that listens to the reference is empty. In addition, since ω_{R_i} is independent of the weight values of other peers, the switching behavior between modes ☁ and Ø can only be partially controlled as follows:

i) if a link peer maintains $\omega_{R_i}(t) = 1$ for τ_{R_1} seconds, then the system stays in mode ☁ for at least that time;

ii) however, if an agent keeps $\omega_{R_i}(t) = 0$ for τ_{R_0} seconds, the system may not remain or even be in mode Ø, since other peers may set their weights to 1 at anytime.

Thus, only a dwell time from mode ☁ to mode Ø can be enforced, but not in the opposite direction. While this may complicate the analysis, the advantage of defining a switching logic where $\omega_{R_i}(t)$ is independent of other link weights is that link peers need not exchange additional information over the network. Next, without loss of generality, assume $\gamma(t_0) = \emptyset$, as depicted in Figure 5.3.

As a result, all odd m define a time t_m where the system switches from mode Ø to mode ☁, whereas even m correspond to a switch in the opposite direction, see Figure 5.3. Then, expressing m in terms of k, the number of Ø-☁ cycles completed, the logic in Table 5.1 for relaxed temporal constraints can only enforce the following slow switching constraints:

$$\left.\begin{aligned} t_{2k} - t_{2k-1} &> \tau_{R_1}, \\ t_{2k-1} - t_{2k-2} &\geq 0, \end{aligned}\right\} \qquad \text{with} \qquad k \in \left\{1, \ldots, \left\lfloor \frac{n_s}{2} \right\rfloor\right\}. \tag{5.12}$$

Notice that the dwell time τ_{R_0}, introduced in Table 5.1, does not appear in Equation (5.12). Moreover, τ_{R_0} forces agents to ignore the reference information for some time while $|\xi_i(t) - \xi_R(t)| > \Delta_t(t)$. This delays the convergence of $|\xi_i(t) - \xi_R(t)|$ to the desired temporal window. For these reasons, hereafter $\tau_{R_0} = 0$. Figure 5.3 serves as an aid to understand Equation (5.12), and shows a Gantt chart with the temporal evolution of the link weights for a fleet with 3 link peers. The identification numbers of these agents are shown in the ordinate, whereas the abscissa represents time. The horizontal bars are highlighted in blue if $\omega_{R_i}(t) = 0$, and are colored red if $\omega_{R_i}(t) = 1$. Transitions from mode Ø to mode ☁, and vice versa, are marked by vertical lines. The vertical zigzag pattern denotes that a period of time is not represented in the chart.

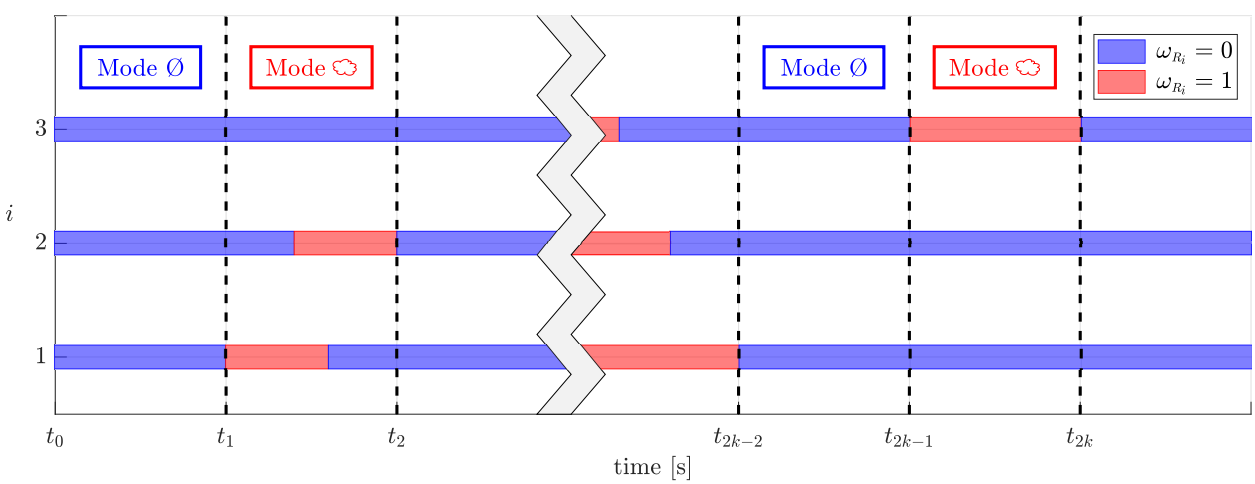


Figure 5.3: Example of the temporal evolution of three link weights.

The following lemma combines algebraic graph and Lyapunov theory to analyze the system dynamics in mode ☁, and concludes that the origin of (5.11) is GUES when $\tilde{\omega}(t) \geq 1$.

Lemma 2 *Assume ideal target-tracking capabilities* $\boldsymbol{u}_{\mathcal{T}_{\models}}(t) \equiv \mathbf{0}$*, the information flow* $\mathcal{G}(t)$ *satisfies Assumptions 3 through 5, and* $\tilde{\omega}(t) \geq 1$*. Then, there exist known control gains* k_R*,* k_P*, and* k_I *such that*

$$|\xi_i(t) - \xi_R(t)| \leq \kappa_{t,r} \|\boldsymbol{\zeta_0}\| \, e^{-\lambda_r t},$$
$$|\xi_i(t) - \xi_j(t)| \leq \kappa_{c,r} \|\boldsymbol{\zeta_0}\| \, e^{-\lambda_r t},$$
$$|\dot{\xi}_i(t) - \rho| \leq \kappa_{r,r} \|\boldsymbol{\zeta_0}\| \, e^{-\lambda_r t},$$

for all $t \geq 0$*, and all* i*,* $j \in \mathcal{I}$*, where* $\kappa_{t,r}$*,* $\kappa_{c,r}$*,* $\kappa_{r,r}$ *are known constants, and* λ_r *is the guaranteed rate of convergence*

$$\lambda_r := \frac{k_P n \mu}{(1 + (k_P n + k_R)\, T)^2} \left(1 + \eta_{I,r} + f_r\right)^{-1},$$

with design parameters $\eta_{I,r} \geq 2$*,* $\eta_{R,r} > 1$*, and known function* $f_r(\eta_{I,r}, \eta_{R,r})$*.*

Proof. The proof and expressions for $\kappa_{t,r}$, $\kappa_{c,r}$, $\kappa_{r,r}$, and f_r are given in Appendix C.3.1. This proof is constructive and provides explicit bounds for the control gains k_R, k_P, and k_I. □

Remark 4 *The definition of* λ_r *given in Lemma 2 is suitable to identify the major trends that affect this rate of convergence. However, the value of* λ_r *is not as straightforward to compute as it may appear from this definition. The reason is the following choice of control gains specified in the proof of Lemma 2:*

$$k_P > 0, \qquad \frac{k_I}{k_P} = \eta_{I,r} \lambda_r, \qquad k_R = \eta_{R,r} n \xi_r \lambda_r,$$

with $\xi_r := \eta_{I,r}\left(1 - \frac{n_\ell}{n}\right) + 1$*. Note that* k_R *is given as a function of* λ_r*. However, the definition of* λ_r *in Lemma 2 includes* k_R*. Appendix C.3.1 proves that for every valid choice of* $\eta_{I,r}$ *and* $\eta_{R,r}$ *there exists a valid pair* k_R *and* λ_r*. The value of* λ_r *is the only positive real root of the equation*

$$\lambda_r \left(1 + k_P n T + \eta_{R,r} n \xi_r T \lambda_r\right)^2 - \frac{k_P n \mu}{1 + \eta_{I,r} + f_r} = 0.$$

The existence of a single positive real root for the equation above is proven using Routh's criterion.

So far, modes Ø and ☁ have been analyzed separately in Theorem 1 and Lemma 2, respectively. The following theorem combines these results with switched systems theory to find τ_{R_1}, and applies

state machine theory to merge these results with the link-weight logic in Table 5.1. This yields finite-time convergence of the temporal error $|\xi_i(t) - \xi_R(t)|$ to the desired temporal window $\Delta_t(t)$, and exponential convergence to the origin of the coordination error $|\xi_i(t) - \xi_R(t)|$, and rate error $|\dot{\xi}_i(t) - \dot{\xi}_R|$.

Theorem 2 *Assume ideal target-tracking capabilities $\boldsymbol{u}_{\mathcal{T}_{\models}}(t) \equiv \mathbf{0}$, and the information flow $\mathcal{G}(t)$ satisfies Assumptions 3 through 5. Then, there exist known control gains k_R, k_P, k_I, and dwell time*

$$\tau_{R_1} = \eta_\tau + \max\left\{0, \frac{1}{2\lambda_r}\ln a_\tau\right\},$$

such that the dynamics in (5.11) *switch between modes $\emptyset$ and* ☁ *finitely many times with*

$$n_s \leq 2\left\lceil \frac{1}{\lambda_r \eta_\tau} \ln\left(\kappa_{n_s}\frac{\|\boldsymbol{\zeta_0}\|}{\underline{\Delta}_t}\right)\right\rceil,$$

satisfying

$$\begin{aligned}
|\xi_i(t_{2k}) - \xi_R(t_{2k})| &\leq \kappa_{t,r}\|\boldsymbol{\zeta_0}\|\, e^{-\lambda_r\eta_\tau k},\\
|\xi_i(t_{2k}) - \xi_j(t_{2k})| &\leq \kappa_{c,r}\|\boldsymbol{\zeta_0}\|\, e^{-\lambda_r\eta_\tau k},\\
|\dot{\xi}_i(t_{2k}) - \rho| &\leq \kappa_{r,r}\|\boldsymbol{\zeta_0}\|\, e^{-\lambda_r\eta_\tau k},
\end{aligned}$$

for all $k \in \left\{1, \ldots, \left\lfloor \frac{n_s}{2}\right\rfloor\right\}$, and all i, $j \in \mathcal{I}$. After some time t_{Δ_t}, the system remains in mode $\emptyset$ for all $t > t_{\Delta_t}$, and satisfies

$$\begin{aligned}
|\xi_i(t) - \xi_R(t)| &< \Delta_t(t),\\
|\xi_i(t) - \xi_j(t)| &\leq \kappa_{c,u}\|\boldsymbol{\zeta_u}(t_{\Delta_t})\|\, e^{-\lambda_u(t-t_{\Delta_t})},\\
|\dot{\xi}_i(t) - \rho| &\leq \kappa_{r,u}\|\boldsymbol{\zeta_u}(t_{\Delta_t})\|\, e^{-\lambda_u(t-t_{\Delta_t})},
\end{aligned}$$

for all i, $j \in \mathcal{I}$, where a_τ and κ_{n_s} are known constants, $\lambda_u := \nu\lambda_u$ is the guaranteed rate of convergence of the coordination and rate errors in mode $\emptyset$, and λ_r is the guaranteed rate of convergence in mode ☁*, with design parameters $\eta_\tau > 0$ and ν.*

Proof. The proof and expressions for a_τ and κ_{n_s} are given in Appendix C.3. Constants $\kappa_{t,r}$, $\kappa_{c,r}$, and $\kappa_{r,r}$ are the same as in Lemma 2, and $\kappa_{c,u}$ and $\kappa_{r,u}$ are the same as in Theorem 1. This proof is constructive and provides explicit bounds for the control gains k_R, k_P, and k_I, as well as ν. □

Remark 5 *The proof of Theorem 2 can be extended to agent-specific temporal constraints*

$$\xi_i(t) - \xi_R(t) \overset{t\to\infty}{\longrightarrow} [-\Delta_{t,i}(t), \Delta_{t,i}(t)], \quad \textit{with} \quad \Delta_{t,i}(t) > 0, \quad \forall i \in \mathcal{I}_\ell.$$

In this case, all peers converge to the narrowest temporal window, $\Delta_t(t) = \min_{i\in\mathcal{I}_\ell} \Delta_{t,i}(t)$.

Remark 6 *The system dynamics* (5.11) *in mode* $\emptyset$ *are equivalent to the dynamics for unenforced temporal constraints in* (5.9)*. However, Theorem 2 indicates that in mode* $\emptyset$ *the rate of convergence is* $\lambda_u := \nu\lambda_u$*, with known* $\nu \leq 1$*, and not just* λ_u *as in Theorem 1. This is the result of ensuring that the choice of control gains* k_P *and* k_I *is simultaneously feasible for modes* $\emptyset$ *and* ☁*. Since this is a distributed switching system, link peers have no knowledge of the switching signal* $\gamma(t)$*, and thus cannot change the control gains accordingly.*

The next section addresses the stability of the system when $|\xi_i(t) - \xi_R(t)|$ must converge to the origin.

5.3.3 Strict Temporal Constraints

The link-weight logic in Table 5.1 for strict temporal constraints ($\omega_{R_i}(t) \equiv 1$) implies that $\tilde{\omega}(t) \equiv n_\ell$ and $\boldsymbol{\omega}(t) \equiv \boldsymbol{v} := [\mathbf{1}_{\mathbf{n}_\ell}^\top, \mathbf{0}]^\top$, which can be leveraged to simplify the system dynamics in Equation (5.7). Then, the following theorem uses algebraic graph and Lyapunov theory to prove that the origin of the system dynamics in (5.7) is GUES.

Theorem 3 *Assume ideal target-tracking capabilities* $\boldsymbol{u}_{\mathcal{T}_\models}(t) \equiv \mathbf{0}$*, and the information flow* $\mathcal{G}(t)$ *satisfies Assumptions 3 through 5. Then, there exist known control gains* k_R*,* k_P *and* k_I*, such that*

$$|\xi_i(t) - \xi_R(t)| \leq \kappa_{t,s}\|\boldsymbol{\zeta_0}\|\, e^{-\lambda_s t},$$
$$|\xi_i(t) - \xi_j(t)| \leq \kappa_{c,s}\|\boldsymbol{\zeta_0}\|\, e^{-\lambda_s t},$$
$$|\dot{\xi}_i(t) - \rho| \leq \kappa_{r,s}\|\boldsymbol{\zeta_0}\|\, e^{-\lambda_s t},$$

for all $t \geq 0$*, and all* i*,* $j \in \mathcal{I}$*, where* $\kappa_{t,s}$*,* $\kappa_{c,s}$*,* $\kappa_{r,s}$ *are known constants, and* λ_s *is the guaranteed rate of convergence*

$$\lambda_s := \frac{k_P n \mu}{(1 + (k_P n + k_R)\, T)^2} \left(1 + \eta_{I,s} + f_s\right)^{-1},$$

with design parameters $\eta_{I,s} \geq 2$, $\eta_{R,s} > 1$, and known function $f_s(\eta_{I,s}, \eta_{R,s})$.

Proof. The proof and expressions for $\kappa_{t,s}$, $\kappa_{c,s}$, $\kappa_{r,s}$, and f_s are given in Appendix C.4. This proof is constructive and provides explicit bounds for the control gains k_R, k_P, and k_I. □

Remark 7 *The definition of λ_s given in Theorem 3 is suitable to identify the major trends that affect this rate of convergence. However, the value of λ_s is not as straightforward to compute as it may appear from this definition. The reason is the following choice of control gains specified in the proof of Theorem 3:*

$$k_P > 0, \qquad \frac{k_I}{k_P} = \eta_{I,s}\lambda_s, \qquad k_R = \eta_{R,s}\frac{n}{n_\ell}\xi_s\lambda_s,$$

with $\xi_s := \eta_{I,s}\left(1 - \frac{n_\ell}{n}\right) + 1$. Note that k_R is given as a function of λ_s. However, the definition of λ_s in Theorem 3 includes k_R. Appendix C.4 proves that for every valid choice of $\eta_{I,s}$ and $\eta_{R,s}$ there exists a valid pair k_R and λ_s. The value of λ_s is the only positive real root of the equation

$$\lambda_s\left(1 + k_P nT + \eta_{R,s} n\xi_s T\lambda_s\right)^2 - \frac{k_P n\mu}{1 + \eta_{I,s} + f_s} = 0.$$

The existence of a single positive real root for the equation above is proven using Routh's criterion.

Remark 8 *Lemma 2 and Theorem 3 bear a close resemblance to each other. The reason is that the proof of Theorem 3 is a particular case of the proof of Lemma 2. On the other hand, the fundamental differences rely on two elements:*

i) In mode ☁, k_R is proportional to the number of agents n; whereas for strict temporal constraints k_R is proportional to the ratio $\frac{n}{n_\ell}$.

ii) A similar trend presents with f_r and f_s, which appear in the expressions of λ_r and λ_s

$$f_r := \frac{\left(\eta_{R,r} n\xi_r u_\omega + \eta_{I,r} u_v\right)^2}{n\xi_r(\eta_{R,r} - 1)}, \qquad f_s := \frac{u_v^2}{n}\frac{\left(\eta_{R,s}\frac{n}{n_\ell}\xi_s + \eta_{I,s}\right)^2}{\xi_s(\eta_{R,s} - 1)},$$

with ξ_r, ξ_s, $u_\omega \leq n_\ell$, and $u_v \leq n_\ell$ defined in Appendices C.3 and C.4.

This is a result of the switching logic for ω_{R_i}. The variability introduced in mode ☁ requires the proof of Lemma 2 to consider the worst-case scenario, often $\tilde{\omega}(t) = 1$; while for strict temporal

constraints $\tilde{\omega}(t) \equiv n_\ell$. This yields more conservative rates of convergence for mode ☁. However, in simulation the slower rate of convergence often corresponds to strict temporal constraints, as shown in the following section.

5.4 Simulation Results

Figures 5.4 through 5.9 show the simulation results for a group of eight peers with ideal virtual-target-tracking capabilities $\boldsymbol{u}_{\mathcal{T}_\mathbb{E}}(t) \equiv \mathbf{0}$; and subject to the three general coordination strategies presented in this chapter. All simulations were run with three link peers, the same time-varying communication network, same initial conditions, and tuning parameters. Following the convention for $\mathcal{I}_\ell$ and $\mathcal{I}_e$ established in Section 2.4.1, agents 1 through 3 are link peers, and the remaining agents are end peers. The QoS of the network was estimated with the following integral connectivity measure:

$$\hat{\mu}(t) = \frac{1}{n}\frac{1}{T}\int_t^{t+T} \mathbf{QL}\,(\tau)\,\mathbf{Q}^\top \,\mathrm{d}\tau, \tag{5.13}$$

and a fixed $T = 4\,s$. To ensure $\mathcal{G}(t)$ satisfies Assumption 5, a pseudo-random sequence of Laplacian matrices that switches every 0.50 s was generated to maintain $\hat{\mu}$ within the range shown in Figure 5.4. The initial conditions for the coordination states were generated randomly

$$\boldsymbol{\xi}(0) = [-2.38,\ -2.55,\ -0.21,\ -2.54,\ -3.73,\ -1.48,\ -0.02,\ -1.25]^\top \text{ s},$$

and the negative values denote that the agents started the mission later than originally planned. The initial value for the integral states was set to $\chi_{i0} = 1\,\mathrm{s/s}$. The control gains and dwell times are[2]

$$k_P = 0.32, \quad k_I = 0.10, \quad k_R = 0.10, \quad \tau_{R_0} = 0.00\,\mathrm{s}, \quad \text{and} \quad \tau_{R_1} = 0.50\,\mathrm{s}.$$

As shown in Figure 5.8, $\dot{\xi}_R$ is increased by 20% mid-simulation for all scenarios. This effectively acts as a disturbance, since all end peers must relearn the new reference rate through the integral states $\chi_i(t)$. Equations (2.10) and (2.11) indicate that $\dot{\xi}_i(t)$ scales the desired speed profile $\boldsymbol{v}_{\boldsymbol{d},\boldsymbol{i}}(t)$ to generate an online speed command $\boldsymbol{v}_{\boldsymbol{cmd},\boldsymbol{i}}(t)$, and achieve closed-loop coordination. Assum-

[2]The tuning parameters given in Remarks 2, 4 and 7 are conservative. For comparison and performance purposes some of the parameters chosen do not satisfy these bounds.

ing the speed profile $\boldsymbol{v_{d,i}}(t)$ has been designed to satisfy an optimal criterion, any $\dot{\xi}_i(t) \neq \dot{\xi}_R(t)$ steers the fleet away from optimality. The following measure is proposed to evaluate the collective coordination control effort exerted by the fleet

$$\epsilon_u(t) = \|\boldsymbol{u_c}(t) - \dot{\xi}_R(t)\mathbf{1_n}\|_{\mathcal{L}_2} = \int_0^t \|\dot{\boldsymbol{\xi}}(\tau) - \dot{\xi}_R(\tau)\mathbf{1_n}\|^2 \, \mathrm{d}\tau. \tag{5.14}$$

Figure 5.5 shows the collective control effort for the three coordination strategies presented in this chapter. Although anecdotal evidence is provided, since results for a single set of initial conditions are presented, the control effort $\epsilon_u(t)$ associated with unenforced temporal constraints tends to be smaller than that of relaxed specifications, which at the same time tends to be smaller than that of strict temporal constraints. Consequently, if utilized appropriately this framework has the potential to reduce the collective control effort, leading to a reduction in fuel consumption. Hence, operators with supervisory control over the fleet should avoid over-constraining the system by imposing the least restrictive temporal specifications that the mission can safely admit.

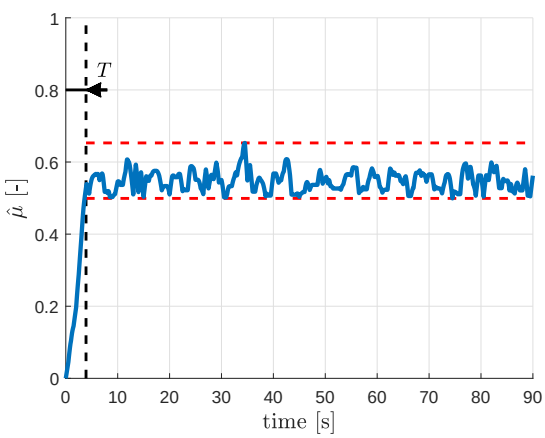


Figure 5.4: QoS of the network.

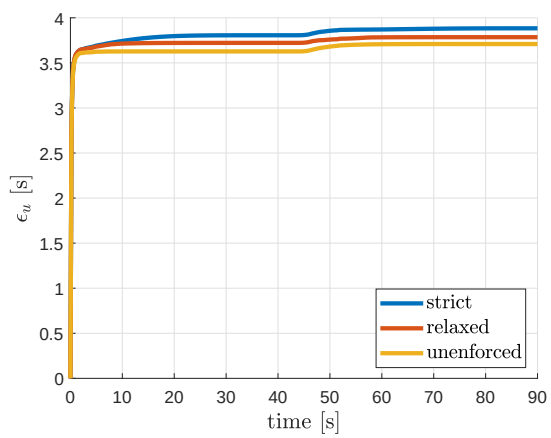


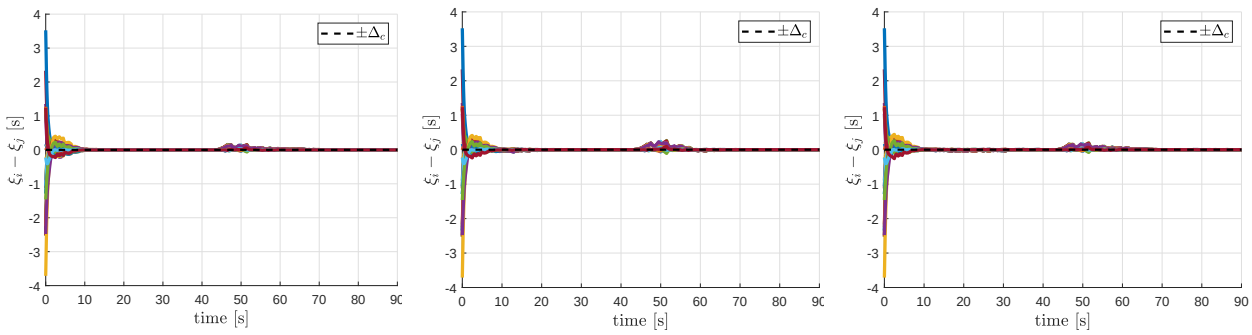
Figure 5.5: Coordination control effort.

Figure 5.6 shows the coordination error for unenforced, relaxed, and strict temporal constraints. As expected, $|\xi_i(t) - \xi_j(t)|$ converges to the origin, regardless of the temporal specifications imposed.

Figure 5.7 shows the temporal error and illustrates how the protocols successfully impose the desired spectrum of temporal constraints:

i) For unenforced constraints $|\xi_i(t) - \xi_R(t)|$ is bounded and away from 0, see Figure 5.7a.

ii) For relaxed constraints $|\xi_i(t) - \xi_R(t)|$ converges to a time-varying window, depicted by dashed

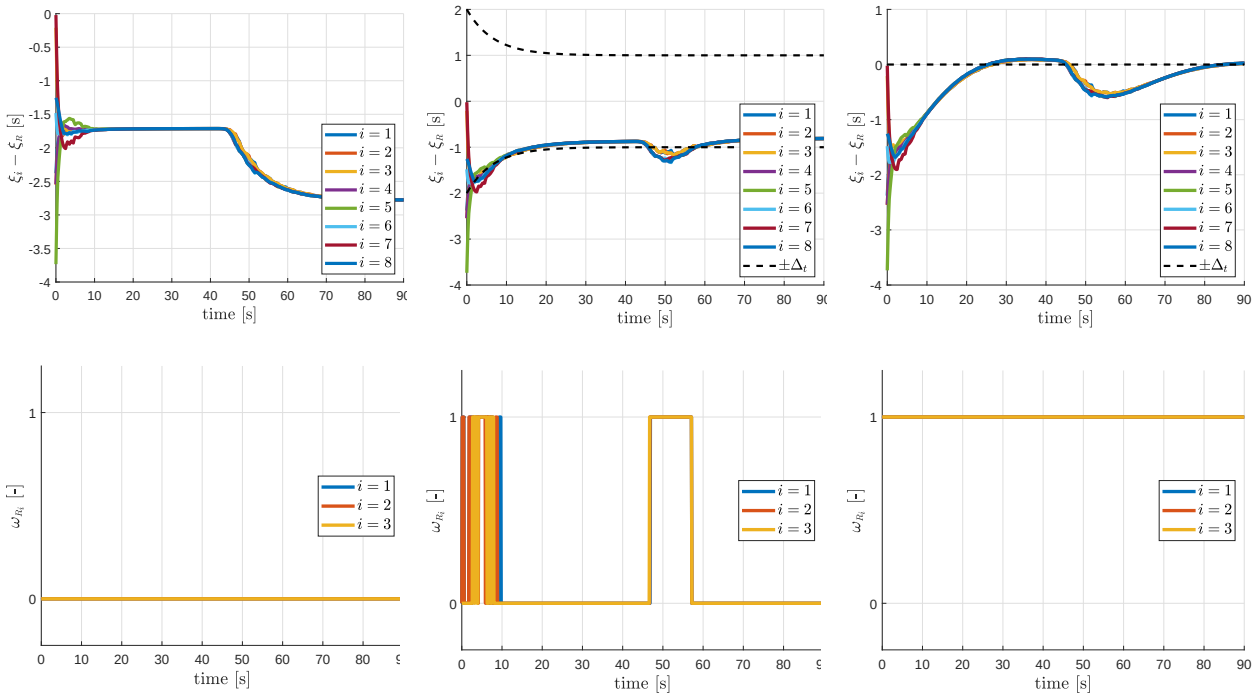


(a) Unenforced (b) Relaxed (c) Strict

Figure 5.6: Coordination errors.

lines in Figure 5.7b. In this scenario, the peers selectively modify their weights to converge to the temporal window. Notice that the disturbance introduced by the change in $\dot{\xi}_R$ is sufficient to push some of the link peers outside the temporal window, which activates their temporal link weights ω_{R_i}.

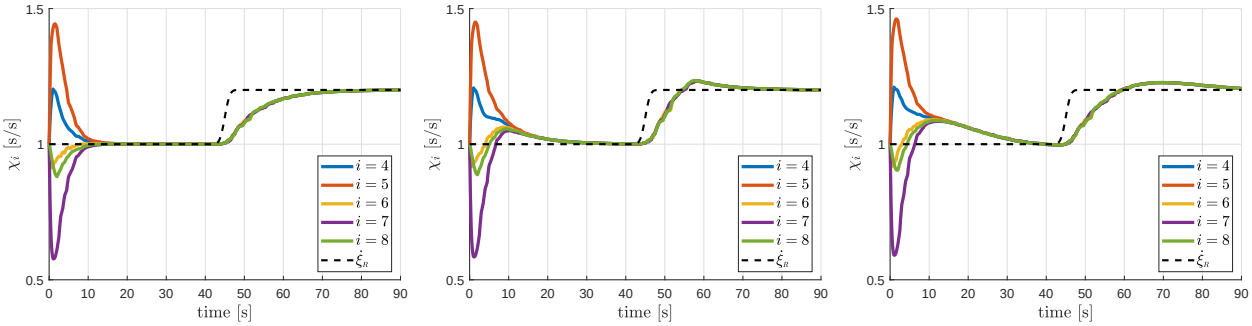
iii) For strict temporal constraints the end peers permanently "listen" to the reference agent, see Figure 5.7c, which forces all agents to track the reference state.



(a) Unenforced (b) Relaxed (c) Strict

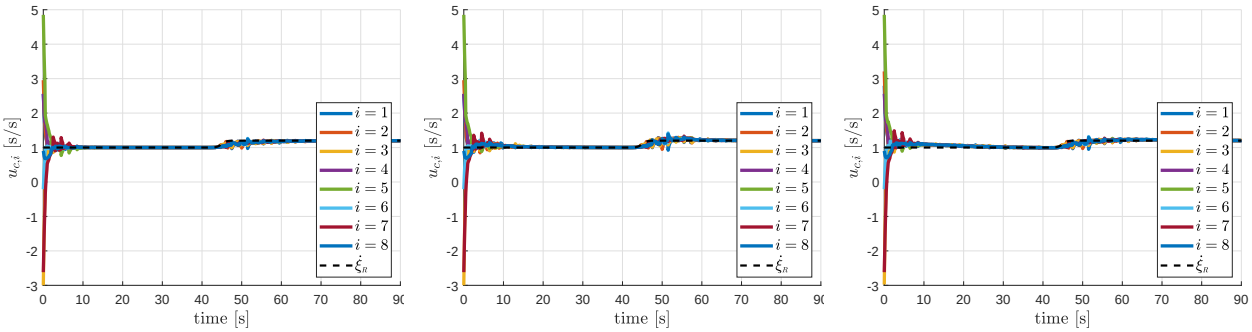
Figure 5.7: Temporal errors (top) and temporal link weights (bottom).

Figure 5.8 shows the evolution of the integral states under unenforced, relaxed, and strict temporal constraints. Observe that convergence to the reference rate becomes slower as the temporal constraints become more stringent. If one wishes to make the rate of convergence of the integral states invariant with respect to the type of temporal specifications imposed, one possible solution is to implement a distributed reference state estimator, as suggested in Chapter 7. However, this comes at the expense of exchanging an additional variable per vehicle over the network. Finally, Figure 5.9 shows the coordination control signal for all vehicles. The piecewise continuous nature of these signals is not only due to the switching logic for ω_{R_i}, but also the switching network topology.



(a) Unenforced (b) Relaxed (c) Strict

Figure 5.8: Integral states.



(a) Unenforced (b) Relaxed (c) Strict

Figure 5.9: Coordination control signal.

Chapter 6

Tight Cooperative Path Following

This chapter utilizes the distributed protocols proposed in Chapter 5, and builds upon the theorems derived for ideal target-tracking capabilities. Assuming a fleet of heterogeneous cooperating vehicles equipped with realistic speed-tracking controllers, this chapter analyzes the structure of the collective target-tracking feedback. This feedback structure is then propagated through the system dynamics as a perturbation to infer transient and steady-state guarantees under non-ideal target-tracking conditions. To illustrate and corroborate the performance of these protocols, simulation results are provided for a group of eight heterogeneous multirotors in a cluttered scenario subject to unenforced, relaxed, and strict temporal constraints.

6.1 Collective Target-Tracking Feedback

To characterize the system performance with realistic and heterogeneous speed-tracking controllers, this section considers Assumption 1 and analyzes the effects of the initial position errors $\boldsymbol{e_{p,i_0}}$ and speed-tracking precision $\bar{e}_{v,i}$ on $\boldsymbol{u}_{\mathcal{T}_\models}(t)$. To this end, Equation (5.5), Lemma 1, and the fact that $\hat{\boldsymbol{t}}_{\boldsymbol{i}}$ is a unit vector yield the following bound for target-tracking feedback:

$$\|\boldsymbol{u}_{\mathcal{T}_\models}(t)\| \leq \bar{u}_{\mathcal{T}_\models}(t) := k_{e_p}\|\boldsymbol{e_{p_0}}\|e^{-\underline{k}_{PF}t} + \frac{k_{e_p}}{\underline{k}_{PF}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|\left(1 - e^{-\bar{k}_{PF}t}\right), \qquad \forall t \geq 0, \tag{6.1}$$

where $\boldsymbol{e_{p_0}} := \left[\boldsymbol{e}_{\boldsymbol{p_0,1}}^\top, \boldsymbol{e}_{\boldsymbol{p_0,2}}^\top, \ldots, \boldsymbol{e}_{\boldsymbol{p_0,n}}^\top\right]^\top$, $\bar{\boldsymbol{e}}_{\boldsymbol{v}} := [\bar{e}_{v,1}, \bar{e}_{v,2}, \ldots, \bar{e}_{v,n}]^\top$, $\underline{k}_{PF} := \min_{\forall i \in \mathcal{I}} k_{PF,i}$, and $\bar{k}_{PF} := \max_{\forall i \in \mathcal{I}} k_{PF,i}$. Note that the coordination control law in Equation (5.3) uses the same gains for all cooperating agents. However, the target-tracking feedback bound above includes individual gains $k_{PF,i}$, and speed-tracking precision bounds $\bar{e}_{v,i}$ for each agent, thus acknowledging that the algorithms and vehicle dynamics beneath the coordination layer can be heterogeneous.

Notice also, that the bound in Equation (6.1) has two components, depicted in Figure 6.1:

i) An exponentially decaying term, proportional to $\|\boldsymbol{e_{p_0}}\|$, shown in orange, and induced by the speed command in (2.11).

ii) A uniformly bounded term, proportional to $\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|$, shown in yellow, and caused by the underlying non-ideal speed-tracking controllers.

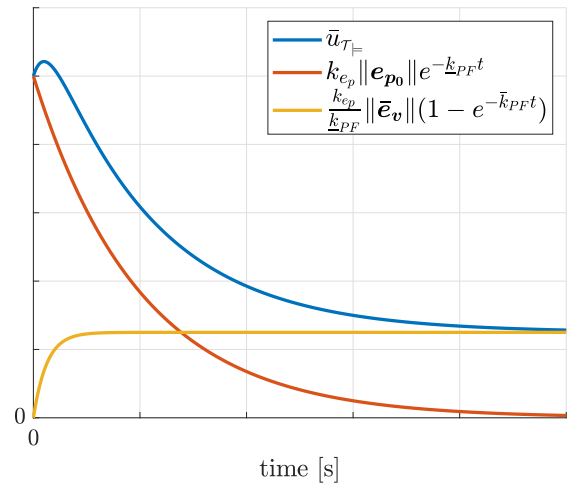


Figure 6.1: Target-tracking feedback bound.

The following section explores how the perturbation in Equation (6.1) propagates through the system dynamics for unenforced, relaxed, and strict temporal constraints under tight coordination.

6.2 Stability with Non-Ideal Target Tracking

Next, Theorems 1, 2, and 3 are extended to account for non-ideal speed-tracking controllers, and infer conditions that ensure the maximum speed command for each UAS is not exceeded.

6.2.1 Unenforced Temporal Constraints

As in Section 5.3.1, the link-weight logic in Table 5.1 for unenforced temporal specifications ($\omega_{R_i}(t) \equiv \mathbf{0}$) decouples the collective coordination and rate errors $\boldsymbol{\zeta_u}(t)$ from the temporal error $\zeta_t(t)$, and yields

$$\dot{\zeta}_t(t) = \boldsymbol{A_t}\boldsymbol{\zeta_u}(t) + \boldsymbol{B_t}\boldsymbol{u}_{\mathcal{T}_{\models}}(t), \qquad \zeta_t(0) = \zeta_{t_0}, \tag{6.2a}$$

$$\dot{\boldsymbol{\zeta}}_{\boldsymbol{u}}(t) = \boldsymbol{A_u}(t)\boldsymbol{\zeta_u}(t) + \boldsymbol{B_u}\boldsymbol{u}_{\mathcal{T}_{\models}}(t), \qquad \boldsymbol{\zeta_u}(0) = \boldsymbol{\zeta_{u_0}}, \tag{6.2b}$$

where $\boldsymbol{A_t}$ and $\boldsymbol{A_u}(t)$ are given in (5.9c), $\boldsymbol{B_t} := -\frac{1}{n}\mathbf{1}_{\mathbf{n}}^{\top}$, and $\boldsymbol{B_u} := \left[\mathbf{Q}^{\top}, \mathbf{0}\right]^{\top}$. The following theorem uses the proof of Theorem 1, and perturbation theory to conclude that the temporal error in (6.2a) is integral Input-to-State Stable (iISS) and can grow at most linearly with time, whereas the coordination and rate errors in (6.2b) are λ_u-weighted iISS with respect to $\boldsymbol{u}_{\mathcal{T}_{\models}}(t)$.

Theorem 4 *Assume the underlying speed-tracking controller for all agents satisfies Assumption 1, the information flow $\mathcal{G}(t)$ satisfies Assumptions 3 through 5, and the speed profiles assigned to each agent by the trajectory generation algorithms satisfy Assumption 2. If the collective speed-tracking precision satisfies*

$$\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| < \min_{i\in\mathcal{I}} \frac{v_{max,i} - \rho v_{d_{max},i}}{1 + \left(\kappa_{r,u_v} + \frac{k_{e_p}}{\underline{k}_{PF}}\right) v_{d_{\max},i}},$$

then there exist known control gains k_P, k_I, and $k_{PF,i} > 0$ for all $i \in \mathcal{I}$, such that for all initial conditions $(\boldsymbol{\zeta_{u_0}}, \boldsymbol{e_{p_0}}) \in \Omega_{u_0}$, the speed command in (2.11), with the protocol for unenforced temporal constraints in (5.1) and (5.2), ensures that

$$\|\boldsymbol{v_{cmd,i}}(t)\| \leq v_{max,i}, \qquad \forall t \geq 0, \qquad \forall i \in \mathcal{I},$$

and the individual temporal, coordination, and rate errors satisfy

$$\begin{bmatrix} |\xi_i(t) - \xi_R(t)| \\ |\xi_i(t) - \xi_j(t)| \\ |\dot{\xi}_i(t) - \rho| \end{bmatrix} \leq \boldsymbol{K_u}(t) \begin{bmatrix} |\zeta_{t_0}| \\ \|\boldsymbol{\zeta_{u_0}}\| \\ \|\boldsymbol{e_{p_0}}\| \\ \|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \end{bmatrix}, \qquad \forall t \geq 0, \qquad \forall i, j \in \mathcal{I},$$

where $\boldsymbol{K_u}(t) \in \mathbb{R}^{3\times 4}$ is defined as

$$\boldsymbol{K_u} := \begin{bmatrix} 1 & \kappa_{t,u_1} e^{-\lambda_u t} + \kappa_{t,u_2} & \frac{\kappa_{t,u_{p_1}}}{\lambda_u - \underline{k}_{PF}} \left(e^{-\underline{k}_{PF} t} - e^{-\lambda_u t}\right) + \kappa_{t,u_{p_2}} & \kappa_{t,u_{v_1}} + \kappa_{t,u_{v_2}} t \\ 0 & \kappa_{c,u} e^{-\lambda_u t} & \frac{\kappa_{c,u_p}}{\lambda_u - \underline{k}_{PF}} \left(e^{-\underline{k}_{PF} t} - e^{-\lambda_u t}\right) & \kappa_{c,u_v} \\ 0 & \kappa_{r,u} e^{-\lambda_u t} & \frac{\kappa_{r,u_p}}{\lambda_u - \underline{k}_{PF}} \left(e^{-\underline{k}_{PF} t} - e^{-\lambda_u t}\right) + k_{e_p} e^{-\underline{k}_{PF} t} & \kappa_{r,u_v} + \frac{k_{e_p}}{\underline{k}_{PF}} \end{bmatrix},$$

Ω_{u_0} is a known non-empty set, κ_{t,u_1}, κ_{t,u_2}, $\kappa_{t,u_{p_1}}$, $\kappa_{t,u_{p_2}}$, $\kappa_{t,u_{v_1}}$, $\kappa_{t,u_{v_2}}$, $\kappa_{c,u}$, κ_{c,u_p}, κ_{c,u_v}, $\kappa_{r,u}$, κ_{r,u_p}, and κ_{r,u_v} are known constants, and $\lambda_u = \nu \underline{\lambda}_u$ is the guaranteed rate of convergence with $\nu = 1$.

Proof. Constants κ_{t,u_1}, κ_{t,u_2}, $\kappa_{c,u}$, and $\kappa_{r,u}$, the bounds for the control gains k_P and k_I, and the rate of convergence $\underline{\lambda}_u$ are the same as in Theorem 2. The proof and expressions for Ω_{u_0}, $\kappa_{t,u_{p_1}}$, $\kappa_{t,u_{p_2}}$, $\kappa_{t,u_{v_1}}$, $\kappa_{t,u_{v_2}}$, κ_{c,u_p}, κ_{c,u_v}, κ_{r,u_p}, and κ_{r,u_v} are given in Appendix C.5. $\square$

To help in the interpretation of Theorem 4, the elements in matrix $\boldsymbol{K_u}(t)$ have been color coded and organized into groups with similar behavior and origin:

i) The elements in blue are exponentially decaying and are induced by the dynamics of $\boldsymbol{\zeta_u}(t)$ in Equation (5.9).

ii) The block in purple represents the interaction between the dynamics of $\boldsymbol{\zeta_u}(t)$ in Equation (6.2) with the position error dynamics in Equation (2.9). Note that these terms are always positive semidefinite for all $t \geq 0$ since

$$\left.\begin{array}{l} \text{if } \ \underline{\lambda}_u - \underline{k}_{PF} > 0 \ \Rightarrow \ e^{-\underline{k}_{PF}t} - e^{-\underline{\lambda}_u t} \geq 0 \\ \text{if } \ \underline{\lambda}_u - \underline{k}_{PF} < 0 \ \Rightarrow \ e^{-\underline{k}_{PF}t} - e^{-\underline{\lambda}_u t} \leq 0 \end{array}\right\} \ \Rightarrow \ \frac{1}{\underline{\lambda}_u - \underline{k}_{PF}}\left(e^{-\underline{k}_{PF}t} - e^{-\underline{\lambda}_u t}\right) \geq 0, \quad \forall t \geq 0.$$

Considering also the case when $\underline{\lambda}_u = \underline{k}_{PF}$ yields

$$\lim_{\underline{k}_{PF} \to \underline{\lambda}_u} \frac{1}{\underline{\lambda}_u - \underline{k}_{PF}}\left(e^{-\underline{k}_{PF}t} - e^{-\underline{\lambda}_u t}\right) = t\, e^{-\underline{\lambda}_u t} \geq 0, \qquad \forall t \geq 0.$$

iii) The terms in orange define ultimate bounds that are proportional to $\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|$, whereas the terms in yellow denote ultimate bounds that are proportional to $\|\boldsymbol{\zeta_{u_0}}\|$ and $\|\boldsymbol{e_{p_0}}\|$. Two important conclusions follow:

- Tight coordination cannot be attained unless the speed tracking error of each agent converges to the origin, $\|\boldsymbol{e_{v,i}}(t)\| \to 0$ as $t \to \infty$ for all $i \in \mathcal{I}$.
- The coordinating agents cannot accurately learn the mission rate unless the speed tracking error of each agent converges to the origin, $\|\boldsymbol{e_{v,i}}(t)\| \to 0$ as $t \to \infty$ for all $i \in \mathcal{I}$.

This emphasizes the importance of the control laws underneath the coordination layer. The implementation of a control law that tracks the speed command in Equation (2.11) with accuracy will tighten the ultimate bounds for $|\xi_i(t) - \xi_j(t)|$ and $|\dot{\xi}_i(t) - \rho|$. In this regard, the protocol in Equation (5.3) is to some extent naive, since it attempts to drive the aforementioned errors to the origin, and incurs in a control effort that cannot achieve this goal. Chapter 7 presents a distributed protocol that considers this fundamental limitation, and

ceases its effort at coordination once $|\xi_i(t) - \xi_j(t)|$ falls withing a specified neighborhood of the origin.

iv) The element in red indicates that the temporal error can drift away over time, and this drift is proportional to $\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|$.

v) The terms in green are a direct effect of $\boldsymbol{u}_{\mathcal{T}_{\models}}(t)$, and can be easily identified with the elements in Equatiom (6.1).

The following section derives transient and steady-state performance guarantees for relaxed temporal constraints with non-ideal target-tracking capabilities.

6.2.2 Relaxed Temporal Constraints

Similar to Section 5.3.2, the dynamics in (5.7) are rewritten as a system that switches between modes Ø and ☁, with switching signal $\gamma(t) : [0, \infty) \mapsto \mathcal{S}$, and $\mathcal{S} := \{\emptyset, ☁\}$

$$\dot{\boldsymbol{\zeta}}(t) = \boldsymbol{A}_{\gamma}(t)\boldsymbol{\zeta}(t) + \boldsymbol{B}\boldsymbol{u}_{\mathcal{T}_{\models}}(t), \qquad \boldsymbol{\zeta}(0) = \boldsymbol{\zeta_0}, \qquad \gamma \in \mathcal{S}, \tag{6.3}$$

where $\boldsymbol{A}_{\emptyset}$ and $\boldsymbol{A}_{☁}$ are defined in Section 5.3.2. The discontinuities in $\gamma(t)$ occur at the switching times t_m, with $m \in \mathbb{N}$. Recall Theorem 4 proves $|\xi_i(t) - \xi_R(t)|$ can drift away in mode Ø. Then, anticipating that τ_{R_1} may now be a function of the time spent in mode Ø, a subtle change in the link-weight logic is introduced

$$\omega_{R_i}(t) = \begin{cases} 1, & \text{if } |\xi_i(t) - \xi_R(t)| \geq \Delta_t(t) \ \wedge \ t - t_s^i > \tau_{R0}, \\ 0, & \text{if } |\xi_i(t) - \xi_R(t)| < \Delta_t(t) \ \wedge \ t - t_s^i > \tau_{R_1}^i(\Delta t_0^i(t)), \\ \omega_{R_i}^+(t_s^i), & \text{otherwise}, \end{cases} \tag{6.4}$$

where $\Delta t_0^i(t)$ is defined as follows:

$$\Delta t_0^i(t) := \begin{cases} t_s^i - t_{s-1}^i, & \text{if } \omega_{R_i}(t) = 1, \\ 0, & \text{if } \omega_{R_i}(t) = 0. \end{cases}$$

As depicted in Figure 6.2, t_s^i and t_{s-1}^i respectively denote the last and second to last times the ith link peer switched the value of $\omega_{R_i}(t)$. Part of the challenge is that link peers ignore the time spent in mode $\emptyset$, and hence can only use $\Delta t_0^i(t)$ to counteract the drift of the collective system. Then, assuming $\gamma(t_0) = \emptyset$ and expressing m in terms of the number of $\emptyset$-☁ cycles completed k, the logic above can only enforce the following slow switching constraints for the system dynamics:

$$\left.\begin{aligned} t_{2k} - t_{2k-1} &> \tau_{R_{1,k}}, \\ t_{2k-1} - t_{2k-2} &\geq 0, \end{aligned}\right\} \tag{6.5}$$

where $\tau_{R_{1,k}} := \max_{i \in \mathcal{I}_\ell} \tau_{R_1}^i(t_{2k-1})$. As opposed to Section 5.3.2, each agent implements a different dwell time $\tau_{R_1}^i(t)$, and the collective dwell time $\tau_{R_{1,k}}$ varies with each $\emptyset$-☁ cycle. See Figure 5.3 for a visual aid in the interpretation of (6.5).

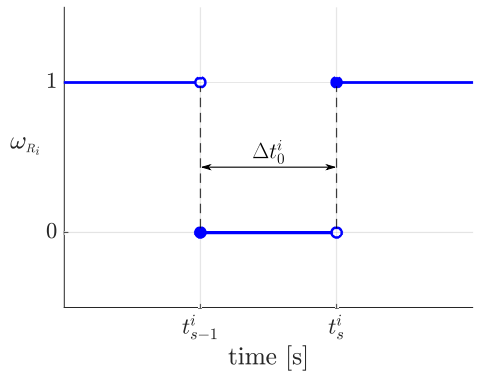


Figure 6.2: Consecutive link-weight switches.

The following lemma uses the proof of Lemma 2 and perturbation theory to analyze the system in mode ☁, and concludes that the dynamics in (6.3) are λ_r-weighted iISS with respect to $\boldsymbol{u}_{\mathcal{T}_\models}(t)$.

Lemma 3 *Assume the underlying speed-tracking controller for all agents satisfies Assumption 1, the information flow $\mathcal{G}(t)$ satisfies Assumptions 3 through 5, and $\tilde{\omega}(t) \geq 1$. Then, there exist known control gains k_R, k_P, k_I, and $k_{PF,i} > 0$ for all $i \in \mathcal{I}$ such that*

$$\begin{bmatrix} |\xi_i(t) - \xi_R(t)| \\ |\xi_i(t) - \xi_j(t)| \\ |\dot{\xi}_i(t) - \rho| \end{bmatrix} \leq \boldsymbol{K_r}(t) \begin{bmatrix} \|\boldsymbol{\zeta_0}\| \\ \|\boldsymbol{e_{p_0}}\| \\ \|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \end{bmatrix}, \qquad \forall t \geq 0, \qquad \forall i,\, j \in \mathcal{I},$$

where $\boldsymbol{K_r}(t) \in \mathbb{R}^{3\times 3}$ is defined as

$$\boldsymbol{K_r}(t) := \begin{bmatrix} \kappa_{t,r} e^{-\lambda_r t} & \frac{\kappa_{t,r_p}}{\lambda_r - \underline{k}_{PF}} \left(e^{-\underline{k}_{PF} t} - e^{-\lambda_r t}\right) & \kappa_{t,r_v} \\ \kappa_{c,r} e^{-\lambda_r t} & \frac{\kappa_{c,r_p}}{\lambda_r - \underline{k}_{PF}} \left(e^{-\underline{k}_{PF} t} - e^{-\lambda_r t}\right) & \kappa_{c,r_v} \\ \kappa_{r,r} e^{-\lambda_r t} & \frac{\kappa_{r,r_p}}{\lambda_r - \underline{k}_{PF}} \left(e^{-\underline{k}_{PF} t} - e^{-\lambda_r t}\right) + k_{e_p} e^{-\underline{k}_{PF} t} & \kappa_{r,r_v} + \frac{k_{e_p}}{\underline{k}_{PF}} \end{bmatrix},$$

$\kappa_{t,r}$, κ_{t,r_p}, κ_{t,r_v}, $\kappa_{c,r}$, κ_{c,r_p}, κ_{c,r_v}, $\kappa_{r,r}$, κ_{r,r_p}, κ_{r,r_v} *are known constants, and* λ_r *is the guaranteed rate of convergence.*

Proof. Constants $\kappa_{t,r}$, $\kappa_{c,r}$, and $\kappa_{r,r}$, the bounds for the control gains k_R, k_P, and k_I, and the rate of convergence λ_r are the same as in Lemma 2. The proof and expressions for κ_{t,r_p}, κ_{t,r_v}, κ_{c,r_p}, κ_{c,r_v}, κ_{r,r_p}, and κ_{r,r_v} are given in Appendix C.6.1. □

The elements in matrix $\boldsymbol{K_r}(t)$ have been color coded using the same criteria as in Section 6.2.1. The most notable differences between $\boldsymbol{K_u}(t)$ and $\boldsymbol{K_r}(t)$ are that in matrix $\boldsymbol{K_r}(t)$ the ultimate bounds in yellow, and the drifting term in red have vanished. As a result, the steady-state value of the temporal error $|\xi_i(t) - \xi_R(t)|$ in mode ☁ is proportional to $\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|$. The next corollary follows from this analysis.

Corollary 1 *Given the assumptions in Lemma 3, if the collective speed-tracking error satisfies*

$$\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \leq \frac{\underline{\Delta}_t}{\kappa_{t,r_v}},$$

then there exists a time $t_{\Delta_t} \geq 0$ *such that*

$$|\xi_i(t) - \xi_R(t)| \leq \Delta_t(t), \qquad \forall t \geq t_{\Delta_t}, \qquad \forall i \in \mathcal{I}.$$

So far, modes Ø and ☁ have been analyzed separately in Theorem 4 and Lemma 3, respectively. The following theorem combines these results, perturbation and switched systems theory, and convergence properties of infinite series to find $\tau^i_{R_1}(t)$. The theorem indicates that the switched system dynamics in (6.3) are UB and can persistently switch between modes Ø and ☁.

Theorem 5 *Assume the underlying speed-tracking controller for all agents satisfies Assumption 1, the information flow* $\mathcal{G}(t)$ *satisfies Assumptions 3 through 5, and the speed profiles assigned to each agent by the trajectory generation algorithms satisfy Assumption 2. If the desired temporal window satisfies*

$$\bar{\Delta}_t < \min_{i \in \mathcal{I}} \frac{1}{\kappa_{r,r}} \left(\frac{v_{\max,i}}{v_{d_{\max},i}} - \rho \right),$$

and the collective speed-tracking error satisfies

$$\|\boldsymbol{e_v}\| < \min_{i\in\mathcal{I}} \left\{ \frac{v_{\max,i} - \rho v_{d_{\max},i}}{\tilde{\kappa}_{\Omega_{u_v,i}}}, \frac{v_{\max,i} - \left(\rho + \kappa_{r,r}\overline{\Delta}_t\right) v_{d_{\max},i}}{\tilde{\kappa}_{\Omega_{r_v,i}}} \right\},$$

then there exist known control gains k_R, k_P, k_I, and $k_{PF,i} > 0$ for all $i \in \mathcal{I}$, and individual dwell times

$$\tau^i_{R_1}(t) = \max\left\{\epsilon_\tau, \frac{1}{\lambda_r}\ln\Delta t^i_0(t)\right\} + \max\left\{0, \frac{1}{\lambda_r}\ln\kappa_{W_r}\right\}, \qquad \forall \in \mathcal{I}_\ell,$$

such that for all initial conditions $(\boldsymbol{\zeta_0}, \boldsymbol{e_{p_0}}) \in \tilde{\Omega}_0$, the speed command in (2.11), with the protocol in (5.1) and (5.2), and the switching logic in (6.4) ensure that

$$\|\boldsymbol{v_{cmd,i}}(t)\| \leq v_{max,i}, \qquad \forall t \geq 0, \qquad \forall i \in \mathcal{I},$$

and $\gamma(t)$ can switch between $\emptyset$ and ☁ finitely many times in every bounded time interval. The temporal, coordination, and rate errors at the switching times from mode ☁ to $\emptyset$ satisfy

$$\begin{bmatrix} |\xi_i(t_{2k}) - \xi_R(t_{2k})| \\ |\xi_i(t_{2k}) - \xi_j(t_{2k})| \\ |\dot{\xi}_i(t_{2k}) - \rho| \end{bmatrix} \leq \tilde{\boldsymbol{K}}(k) \begin{bmatrix} \|\boldsymbol{\zeta_0}\| \\ \|\boldsymbol{e_{p_0}}\| \\ \|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \end{bmatrix}, \qquad \forall i,\, j \in \mathcal{I}, \qquad k \in \mathbb{N}.$$

In mode $\emptyset$, the temporal, coordination, and rate errors satisfy

$$\begin{bmatrix} |\xi_i(t) - \xi_R(t)| \\ |\xi_i(t) - \xi_j(t)| \\ |\dot{\xi}_i(t) - \rho| \end{bmatrix} \leq \boldsymbol{K_u}(t - t_{2k-2}) \begin{bmatrix} |\zeta_t(t_{2k-2})| \\ \|\boldsymbol{\zeta_u}(t_{2k-2})\| \\ \|\boldsymbol{e_p}(t_{2k-2})\| \\ \|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \end{bmatrix}, \quad \forall\, t_{2k-2} \leq t < t_{2k-1}, \quad \forall i,\, j \in \mathcal{I}, \quad k \in \mathbb{N},$$

whereas in mode ☁ the temporal, coordination, and rate errors satisfy

$$\begin{bmatrix} |\xi_i(t) - \xi_R(t)| \\ |\xi_i(t) - \xi_j(t)| \\ |\dot{\xi}_i(t) - \rho| \end{bmatrix} \leq \boldsymbol{K_r}(t - t_{2k-1}) \begin{bmatrix} \|\boldsymbol{\zeta}(t_{2k-1})\| \\ \|\boldsymbol{e_p}(t_{2k-1})\| \\ \|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \end{bmatrix}, \quad \forall\, t_{2k-1} \leq t \leq t_{2k}, \quad \forall i,\, j \in \mathcal{I}, \quad k \in \mathbb{N},$$

where $\boldsymbol{K_u}(t)$ and $\boldsymbol{K_r}(t)$ are defined in Theorem 4 and Lemma 3, respectively, and $\tilde{\boldsymbol{K}}(k) \in \mathbb{R}^{3\times 3}$ is

$$\tilde{\boldsymbol{K}}(k) := \begin{bmatrix} \kappa_{t,r} e^{-k\lambda_r \epsilon_\tau} & \tilde{\kappa}_{t,p}\, k\, e^{-k\tilde{\lambda}\epsilon_\tau} & & \left(\tilde{\kappa}_{t,v_1} + \tilde{\kappa}_{t,v_2} e^{\lambda_r \epsilon_\tau}\right) \sum_{m=1}^{k} e^{-m\lambda_r \epsilon_\tau} \\ \kappa_{c,r} e^{-k\lambda_r \epsilon_\tau} & \tilde{\kappa}_{c,p}\, k\, e^{-k\tilde{\lambda}\epsilon_\tau} & & \left(\tilde{\kappa}_{c,v_1} + \tilde{\kappa}_{c,v_2} e^{\lambda_r \epsilon_\tau}\right) \sum_{m=1}^{k} e^{-m\lambda_r \epsilon_\tau} \\ \kappa_{r,r} e^{-k\lambda_r \epsilon_\tau} & \tilde{\kappa}_{r,p}\, k\, e^{-k\tilde{\lambda}\epsilon_\tau} & + k_{e_p} e^{-k\, \underline{k}_{PF}\, \epsilon_\tau} & \left(\tilde{\kappa}_{r,v_1} + \tilde{\kappa}_{r,v_2} e^{\lambda_r \epsilon_\tau}\right) \sum_{m=1}^{k} e^{-m\lambda_r \epsilon_\tau} + \frac{k_{e_p}}{\underline{k}_{PF}} \end{bmatrix},$$

$\epsilon_\tau > 0$ is a design parameter, $\tilde{\Omega}_0$ is a known non-empty set, $\tilde{\kappa}_{\Omega_{u_v,i}}$, $\tilde{\kappa}_{\Omega_{r_v,i}}$, κ_{W_r}, $\kappa_{t,r}$, $\tilde{\kappa}_{t,p}$, $\tilde{\kappa}_{t,v_1}$, $\tilde{\kappa}_{t,v_2}$, $\kappa_{c,r}$, $\tilde{\kappa}_{c,p}$, $\tilde{\kappa}_{c,v_1}$, $\tilde{\kappa}_{c,v_2}$, $\kappa_{r,r}$, $\tilde{\kappa}_{r,p}$, $\tilde{\kappa}_{r,v_1}$, and $\tilde{\kappa}_{r,v_2}$, $\tilde{\lambda} := \min\{\underline{k}_{PF}, \lambda_r\}$, and λ_r is the guaranteed rate of convergence in mode ☁.

Proof. Constants $\kappa_{t,r}$, $\kappa_{c,r}$, and $\kappa_{r,r}$, the bounds for the control gains k_R, k_P, and k_I, and the rate of convergence λ_r are the same as in Theorem 2. The proof and expressions for $\tilde{\kappa}_{\Omega_{u_v,i}}$, $\tilde{\kappa}_{\Omega_{r_v,i}}$, κ_{W_r}, $\tilde{\kappa}_{t,p}$, $\tilde{\kappa}_{t,v_1}$, $\tilde{\kappa}_{t,v_2}$, $\tilde{\kappa}_{c,p}$, $\tilde{\kappa}_{c,v_1}$, $\tilde{\kappa}_{c,v_2}$, $\tilde{\kappa}_{r,p}$, $\tilde{\kappa}_{r,v_1}$, and $\tilde{\kappa}_{r,v_2}$ are given in Appendix C.6. □

The elements in matrix $\tilde{\boldsymbol{K}}(k)$ have been color coded using the same criteria as in Section 6.2.1. Note that the propagation of the perturbation in Equation (6.1) through the switched system presents structural similarities with the results in mode ☁. This becomes clear if one compares matrices $\boldsymbol{K_r}(t)$ and $\tilde{\boldsymbol{K}}(k)$. They contain the same-color blocks in the same locations. As shown in Figure 6.3, the elements in $\tilde{\boldsymbol{K}}(k)$ behave as follows:

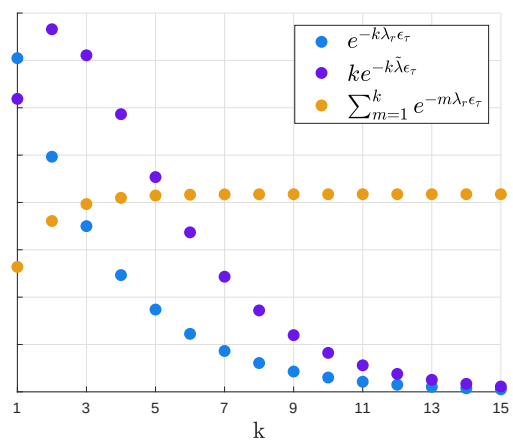


Figure 6.3: Response of the terms in $\tilde{\boldsymbol{K}}$.

i) The blocks in blue decrease exponentially with k, the number of Ø-☁ cycles.

ii) The elements in purple are dominated by a linear growth for small k, and an exponential decay for large k.

iii) The terms in orange are ultimately bounded. Application of D'Alembert's criterion proves convergence of this infinite series.

iv) The terms in green are a direct effect of $\boldsymbol{u}_{\mathcal{T}_{\models}}(t)$ as it propagates through the switched system, and can be easily identified with the elements in Equatiom (6.1).

In all cases, these elements have a similar behavior as their continuous counter parts in $\boldsymbol{K_r}(t)$.

The following corollary follows from Lemma 3 and Theorem 5, and concludes that if the collective speed-tracking error satisfies certain conditions, then two possible behaviors may emerge from the switching logic in Equation (6.4) either

i) After some time t_{Δ_t} the switched system (6.3) stays in mode $\emptyset$, $|\xi_i(t) - \xi_R(t)|$ remains within $\Delta_t(t)$ for all link peers, and $|\xi_i(t) - \xi_R(t)|$ is bounded for all end peers but may lie outside $\Delta_t(t)$; or

ii) The switched system (6.3) persistently switches between modes $\emptyset$ and ☁. In this case, there exists a number of $\emptyset$-☁ cycles k_{Δ_t} beyond which the temporal error $|\xi_i(t_{2k}) - \xi_R(t_{2k})|$ falls inside $\Delta_t(t_{2k})$ for all agents. Between switching times t_{2k} and t_{2k+2}, $|\xi_i(t) - \xi_R(t)|$ remains bounded, but some agents will depart the desired temporal window $\Delta_t(t)$.

Corollary 2 *Given the assumptions in Theorem 5, if the collective speed-tracking error satisfies*

$$\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \leq \underline{\Delta}_t \min\left\{\frac{1}{\kappa_{t,r_v}}, \frac{e^{\lambda_r \epsilon_\tau} - 1}{\tilde{\kappa}_{t,v_1} + \tilde{\kappa}_{t,v_2} e^{\lambda_r \epsilon_\tau}}\right\},$$

then two behaviors may emerge for the switched system (6.3) *either*

i) There exists a time t_{Δ_t} such that $\gamma(t) = \emptyset$ for all $t \geq t_{\Delta_t}$, and

$$\begin{aligned}
&|\xi_i(t) - \xi_R(t)| < \Delta_t(t), \qquad \forall t \geq t_{\Delta_t}, \qquad \forall i \in \mathcal{I}_\ell, \\
&|\xi_i(t) - \xi_R(t)| < \Delta_t(t) + \kappa_{c,u}\|\boldsymbol{\zeta_u}(t_{\Delta_t})\| e^{-\lambda_u(t - t_{\Delta_t})} + \dots \\
&\qquad\qquad + \frac{\kappa_{c,u_p}}{\underline{k}_{PF} - \lambda_u}\|\boldsymbol{e_p}(t_{\Delta_t})\| \left(e^{-\lambda_u(t - t_{\Delta_t})} - e^{-\underline{k}_{PF}(t - t_{\Delta_t})}\right) + \dots \\
&\qquad\qquad + \kappa_{c,u_v}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|, \qquad \forall t \geq t_{\Delta_t}, \qquad \forall i \in \mathcal{I}_e; \qquad \textit{or}
\end{aligned}$$

ii) $\gamma(t)$ persistently switches between modes $\emptyset$ and ☁, and there exists a $k_{\Delta_t} \in \mathbb{N}$ such that

$$|\xi_i(t_{2k}) - \xi_R(t_{2k})| \leq \Delta_t(t_{2k}), \qquad \forall k \geq k_{\Delta_t}, \qquad \forall i \in \mathcal{I},$$

and the temporal errors $|\xi_i(t) - \xi_R(t)|$ between t_{2k} and t_{2k+2} in modes $\emptyset$ and ☁ are bounded for all agents.

Proof. The proof and further details about the bounds for the temporal errors are given in Appendix C.6.4. □

According to Theorem 4 the temporal error can drift away from the origin in mode Ø. However, Theorem 4 only provides an upper bound for the evolution of $|\xi_i(t) - \xi_R(t)|$, and therefore does not guarantee that the temporal errors will indeed drift. This is relevant in the interpretation of Corollary 2 because if $|\xi_i(t) - \xi_R(t)|$ indeed drifts, then the only compatible behavior is item ii). The real implementation of this cooperative system will drift in mode Ø, because the vehicle speed-tracking controllers will invariably incur in small tracking errors. Hence, the real system will exhibit persistent switches between modes Ø and ☁. Therefore, item i) in the corollary is a result of the formality of this proof. One may attempt to remove it assuming the following PE-like conditions involving the speed-tracking error:

$$\frac{1}{T_v}\int_t^{t+T_v} \boldsymbol{e}_{\boldsymbol{v},\boldsymbol{i}}^{\top}(\tau)\boldsymbol{e}_{\boldsymbol{v},\boldsymbol{i}}(\tau)\,\mathrm{d}\tau \geq \mu_v, \qquad \forall t \geq T_v, \qquad i \in \mathcal{I},$$

with $T_v > 0$ and $\mu_v > 0$, and the QoS of the communication network:

$$\frac{1}{n}\frac{1}{T}\int_t^{t+T} \bar{\boldsymbol{L}}\,(\tau)\,\mathrm{d}\tau \leq \bar{\mu}\,\mathbb{I}_{\mathbf{n-1}}\,, \quad \forall\ t \geq T\,,$$

with $T > 0$ and $\mu \leq \bar{\mu} \in (0, 1]$. In accordance with the expected behavior, the following remark analyzes the performance guarantees as the number of Ø-☁ cycles increases.

Remark 9 *As the number of Ø-☁ cycles increases, the switched system* (6.3) *successfully cancels out the effects of the initial error* $\|\boldsymbol{\zeta_0}\|$ *and position-tracking errors* $\|\boldsymbol{e_{p_0}}\|$ *as shown next*

$$\lim_{k\to\infty}\begin{bmatrix} |\xi_i(t_{2k}) - \xi_R(t_{2k})| \\ |\xi_i(t_{2k}) - \xi_j(t_{2k})| \\ |\dot{\xi}_i(t_{2k}) - \rho| \end{bmatrix} \leq \begin{bmatrix} \tilde{\kappa}_{t,v_1} + \tilde{\kappa}_{t,v_2}e^{\lambda_r\epsilon_\tau} \\ \tilde{\kappa}_{c,v_1} + \tilde{\kappa}_{c,v_2}e^{\lambda_r\epsilon_\tau} \\ \tilde{\kappa}_{r,v_1} + \tilde{\kappa}_{r,v_2}e^{\lambda_r\epsilon_\tau} \end{bmatrix} \frac{1}{e^{\lambda_r\epsilon_\tau} - 1}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|.$$

Thus, increasing the value of ϵ_τ *cannot completely eliminate the error induced by* $\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|$*, but can reduce it to some extent, since* $\frac{1}{e^{\lambda_r\epsilon_\tau}-1}$ *and* $\frac{e^{\lambda_r\epsilon_\tau}}{e^{\lambda_r\epsilon_\tau}-1}$ *are monotonically decreasing, and*

$$\lim_{\epsilon_\tau \to \infty} \frac{1}{e^{\lambda_r \epsilon_\tau} - 1} = 0, \qquad \lim_{\epsilon_\tau \to \infty} \frac{e^{\lambda_r \epsilon_\tau}}{e^{\lambda_r \epsilon_\tau} - 1} = 1.$$

When the value of ϵ_τ is increased the contributions of mode Ø to the temporal, coordination, and rate errors diminish, since the system stays in mode ☁ for longer periods of time.

The following section derives transient and steady-state performance guarantees for strict temporal constraints with non-ideal target-tracking capabilities.

6.2.3 Strict Temporal Constraints

As in Section 5.3.3, the link-weight logic in Table 5.1 for strict temporal specifications ($\omega_{R_i}(t) \equiv 1$) implies that $\tilde{\omega}(t) \equiv n_\ell$ and $\boldsymbol{\omega}(t) \equiv \boldsymbol{v} := [\mathbf{1}_{\mathbf{n}_\ell}^\top, \mathbf{0}]^\top$, which can be leveraged to simplify the system dynamics in Equation (5.7). Then, the following theorem uses the proof of Theorem 3 and perturbation theory to prove that the origin of the system dynamics in (5.7) is λ_s-weighted iISS with respect to $\boldsymbol{u}_{\boldsymbol{\mathcal{T}}_\models}(t)$.

Theorem 6 *Assume the underlying speed-tracking controller for all agents satisfies Assumption 1, the information flow $\mathcal{G}(t)$ satisfies Assumptions 3 through 5, and the speed profiles assigned to each agent by the trajectory generation algorithms satisfy Assumption 2. If the collective speed-tracking precision satisfies*

$$\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| < \min_{i \in \mathcal{I}} \frac{v_{max,i} - \rho v_{d_{max},i}}{1 + \left(\kappa_{r,s_v} + \frac{k_{e_p}}{k_{PF}}\right) v_{d_{\max},i}},$$

then there exist known control gains k_R, k_P, k_I, and $k_{PF,i} > 0$ for all $i \in \mathcal{I}$, such that for all initial conditions $(\boldsymbol{\zeta_0}, \boldsymbol{e_{p0}}) \in \Omega_{s_0}$, the speed command in (2.11), with the protocol for strict temporal constraints in (5.1) and (5.2) ensure that

$$\|\boldsymbol{v_{cmd,i}}(t)\| \leq v_{max,i}, \qquad \forall t \geq 0, \qquad \forall i \in \mathcal{I},$$

and the individual temporal, coordination, and rate errors satisfy

$$\begin{bmatrix} |\xi_i(t) - \xi_R(t)| \\ |\xi_i(t) - \xi_j(t)| \\ |\dot{\xi}_i(t) - \rho| \end{bmatrix} \leq \boldsymbol{K_s}(t) \begin{bmatrix} \|\boldsymbol{\zeta_0}\| \\ \|\boldsymbol{e_{p0}}\| \\ \|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \end{bmatrix}$$

where $\boldsymbol{K_s}(t) \in \mathbb{R}^{3\times 3}$ *is defined as*

$$\boldsymbol{K_s}(t) := \begin{bmatrix} \kappa_{t,s} e^{-\lambda_s t} & \frac{\kappa_{t,s_p}}{\lambda_s - \underline{k}_{PF}} \left(e^{-\underline{k}_{PF} t} - e^{-\lambda_s t}\right) & & \kappa_{t,s_v} & \\ \kappa_{c,s} e^{-\lambda_s t} & \frac{\kappa_{c,s_p}}{\lambda_s - \underline{k}_{PF}} \left(e^{-\underline{k}_{PF} t} - e^{-\lambda_s t}\right) & & \kappa_{c,s_v} & \\ \kappa_{r,s} e^{-\lambda_s t} & \frac{\kappa_{r,s_p}}{\lambda_s - \underline{k}_{PF}} \left(e^{-\underline{k}_{PF} t} - e^{-\lambda_s t}\right) & + \; k_{e_p} e^{-\underline{k}_{PF} t} & \kappa_{r,s_v} & + \; \frac{k_{e_p}}{\underline{k}_{PF}} \end{bmatrix}.$$

Ω_{s_0} *is a known non-empty set,* $\kappa_{t,s}$*,* κ_{t,s_p}*,* κ_{t,s_v}*,* $\kappa_{c,s}$*,* κ_{c,s_p}*,* κ_{c,s_v}*,* $\kappa_{r,s}$*,* κ_{r,s_p}*, and* κ_{r,s_v} *are known constants, and* λ_s *is the guaranteed rate of convergence.*

Proof. Constants $\kappa_{t,s}$, $\kappa_{c,s}$, and $\kappa_{r,s}$, the bounds for the control gains k_R, k_P, and k_I, and the rate of convergence λ_s are the same as in Theorem 3. The proof and expressions for κ_{t,s_p}, κ_{t,s_v}, κ_{c,s_p}, κ_{c,s_v}, κ_{r,s_p}, and κ_{r,s_v} are given in Appendix C.7. □

The elements in matrix $\boldsymbol{K_s}(t)$ have been color coded using the same criteria as in Section 6.2.1. Notice also that $\boldsymbol{K_s}(t)$ has the same structure as $\boldsymbol{K_r}(t)$, but with different constants and rate of convergence. The reason for this resemblance is that strict temporal constraints are a particular case of mode ☁, where all link peers listen to the reference agent at all times.

6.3 Safety Criticality of the Coordination Protocols

Consider now a pair of UASs with identification numbers i and j, equipped with the coordinated path-following algorithms described so far. The time-critical coordination algorithm has two feedback terms, both of importance for the safety of the cooperating peers: the coordination feedback $\boldsymbol{u_c}(t)$ attempts to synchronize the virtual targets to ensure safe separation among the UASs; and the target-tracking feedback $\boldsymbol{u}_{\mathcal{T}_\models}(t)$ slows or expedites the progress of the virtual targets along their trajectories if the actual vehicles are behind or ahead their targets. This establishes a negotiation process between the agents and their virtual targets that prevents vehicles from cutting corners, and thus avoids potential collisions with nearby obstacles, particularly in cluttered environments. These vehicles are also assigned time-deconflicted trajectories, as illustrated in Figure 6.4, that is

$$\|\boldsymbol{p_{d,i}}(t_d) - \boldsymbol{p_{d,j}}(t_d)\| \geq d_s, \qquad \forall t_d \in [t_{d_{init}}, t_{d_{end}}], \qquad i, j \in \mathcal{I}, \qquad i \neq j. \tag{6.6}$$

Generally, the initial and final times of a pair of trajectories may not coincide. As a result, trajectories represented as a sequence of Bézier curves can be clipped accordingly using the de Casteljau algorithm so that the initial and final times of the trajectory segments being compared coincide, as in Equation (6.6). The safety distance used to deconflict these trajectories in the trajectory generation phase

$$d_s = c_s \left(d_{p,i} + d_{p,j} \right),$$

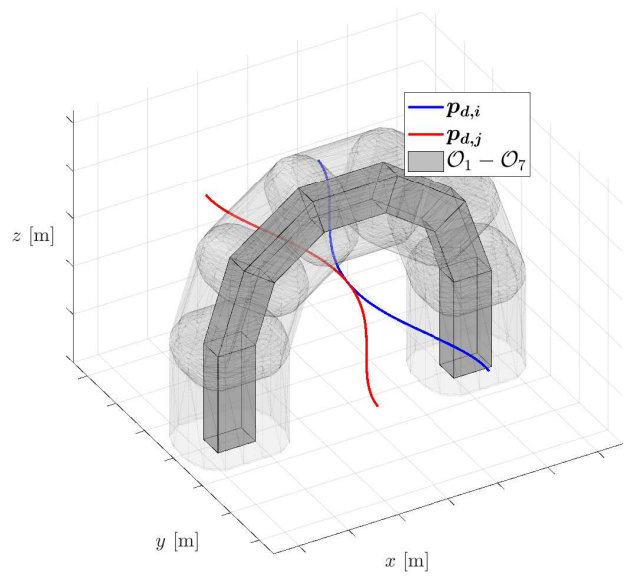


Figure 6.4: Time-deconflicted trajectories.

is informed with the uncertainty distances $d_{p,i}$ and $d_{p,j}$, which account for the dimensions of the vehicle frames, and expected target-tracking errors under nominal conditions. The safety factor $c_s > 1$ provides an additional safety buffer for unmodeled errors. The desired trajectories and the coordination states of the UASs at time t determine the position of the virtual targets

$$\boldsymbol{p_{\mathcal{T},i}}(t) := \boldsymbol{p_{d,i}}(\xi_i(t)), \qquad \boldsymbol{p_{\mathcal{T},j}}(t) := \boldsymbol{p_{d,j}}, (\xi_j(t)).$$

In addition, Lemma 1 provides a bound for the target-tracking errors as a function of time, which can be further simplified as follows:

$$\|\boldsymbol{e_{p,i}}(t)\| \leq \|\boldsymbol{e_{p,i_0}}\| e^{-k_{PF,i}(t-t_{d_{init}})} + \frac{\bar{e}_{v,i}}{k_{PF,i}}, \qquad \|\boldsymbol{e_{p,j}}(t)\| \leq \|\boldsymbol{e_{p,j_0}}\| e^{-k_{PF,j}(t-t_{d_{init}})} + \frac{\bar{e}_{v,j}}{k_{PF,j}}. \quad (6.7)$$

These expressions define two balls centered at each of the virtual targets with a monotonically decreasing radius over time, which serve as bounding regions for the center of mass of each agent. As the virtual targets move, these balls define a pair of shrinking tubes, as illustrated in Figure 6.5. Consequently, the distance between the centers of mass of vehicles i and j can be bounded by

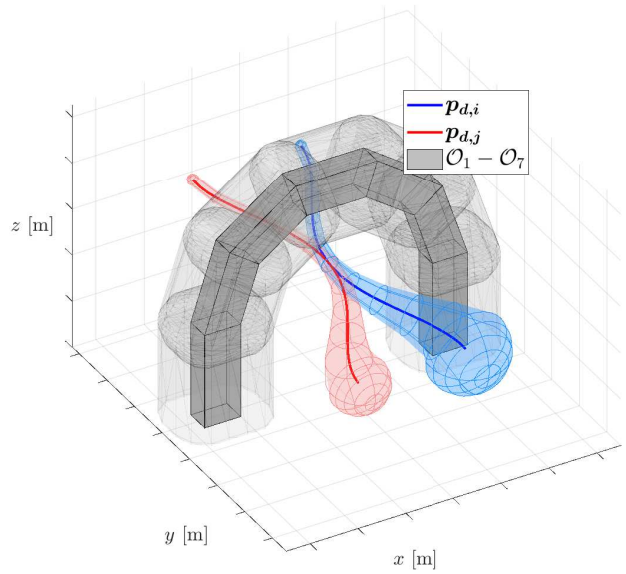


Figure 6.5: Bounding tubes.

$$\begin{aligned}\|\boldsymbol{p_i}(t) - \boldsymbol{p_j}(t)\| &= \|\boldsymbol{p_{\mathcal{T},i}}(t) + \boldsymbol{e_{p,i}}(t) - \boldsymbol{p_{\mathcal{T},j}}(t) - \boldsymbol{e_{p,j}}(t)\| \\ &\geq \|\boldsymbol{p_{d,i}}(\xi_i(t)) - \boldsymbol{p_{d,j}}(\xi_j(t))\| - \|\boldsymbol{e_{p,i}}(t)\| - \|\boldsymbol{e_{p,j}}(t)\|. \end{aligned} \tag{6.8}$$

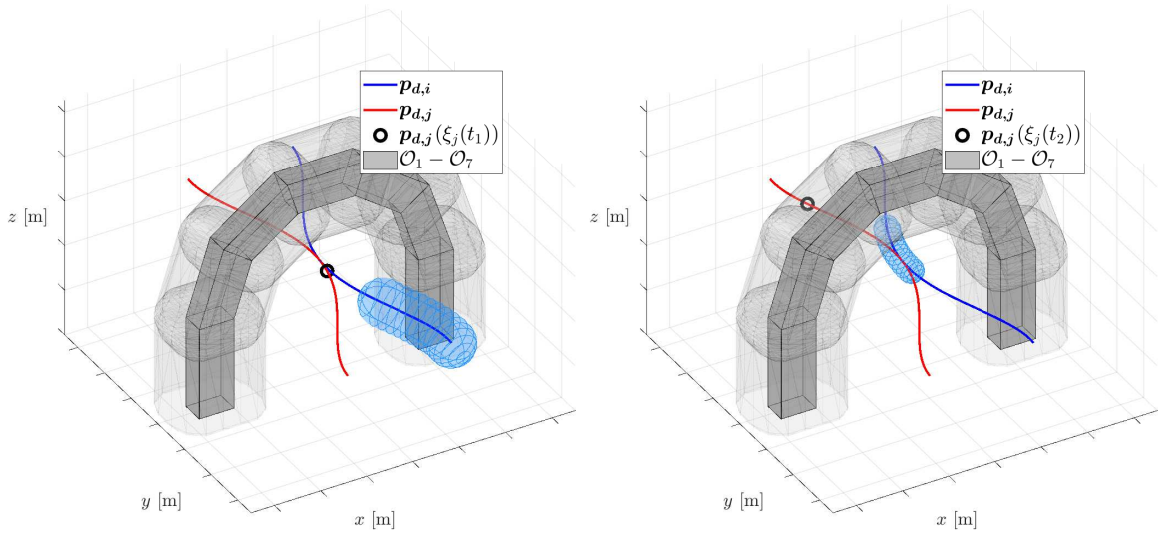
If the virtual targets were perfectly coordinated, that is $\xi_i(t) \equiv \xi_j(t)$, then the distance between the virtual targets would be $\|\boldsymbol{p_{d,i}}(\xi_i(t)) - \boldsymbol{p_{d,j}}(\xi_j(t))\| \geq d_s$ for all $\xi_i, \xi_j \in [t_{d_{init}}, t_{d_{end}}]$. However, Theorems 4, 5, and 6 provide bounds for the individual coordination errors. For simplicity, these bounds will be expressed within this section as

$$|\xi_i(t) - \xi_j(t)| < \delta\xi_{i,j}(\|\boldsymbol{\zeta_0}\|, \|\boldsymbol{e_{p_0}}\|, \|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|, t),$$

for all the coordination strategies presented within this chapter. Henceforth, the arguments of $\delta\xi_{i,j}$ will be dropped for brevity. Moreover, the trajectories used within this thesis are uniformly continuous by construction, and thus for any $\delta p_i > 0$ there exists a $\delta\xi_{i,j} > 0$ such that for all $\xi_i, \xi_j \in [t_{d_{init}}, t_{d_{end}}]$ satisfying

$$|\xi_i - \xi_j| < \delta\xi_{i,j} \implies \|\boldsymbol{p_{d,i}}(\xi_i) - \boldsymbol{p_{d,i}}(\xi_j)\| < \delta p_i, \qquad i, j \in \mathcal{I}, \qquad i \neq j. \tag{6.9}$$

Therefore, for every value of $\xi_j(t) \in [t_{d_{init}}, t_{d_{end}}]$ there exists a bounded time interval where the



(a) Snapshot at time t_1

(b) Snapshot at time $t_2 > t_1$

Figure 6.6: Trajectories, virtual target $\boldsymbol{p_{\mathcal{T},j}}$, and bounding tube for the ith UAS at times t_1 and t_2.

coordination state of the ith peer lies $\xi_i(t) \in [\xi_j(t) - \delta\xi_{i,j}(t), \xi_j(t) + \delta\xi_{i,j}(t)]$. This bounded interval, the trajectory $\boldsymbol{p_{d,i}}$, and Equation (6.7) define a moving tube segment, illustrated in Figure 6.6 for times t_1 and t_2. Hence, to ensure safe separation between the coordinating agents, for every value of $\xi_j(t) \in [t_{d_{init}}, t_{d_{end}}]$ the virtual target $\boldsymbol{p_{d,j}}(\xi_j(t))$ and the tube associated with the ith coordinating peer must be sufficiently separated, as depicted in Figure 6.6. To this end, Equation (6.9) yields

$$\|\boldsymbol{p_{d,i}}(\xi_i(t)) - \boldsymbol{p_{d,j}}(\xi_j(t))\| \geq \|\boldsymbol{p_{d,i}}(\xi_j(t)) - \boldsymbol{p_{d,j}}(\xi_j(t))\| - \|\boldsymbol{p_{d,i}}(\xi_i(t)) - \boldsymbol{p_{d,i}}(\xi_j(t))\| > d_s - \delta p_i(t).$$

Then, applying the same argument as in Equation (6.9) to $\boldsymbol{p_{d,j}}$, choosing the least conservative bound, and plugging the result in Equation (6.8) yields

$$\|\boldsymbol{p_i}(t) - \boldsymbol{p_j}(t)\| > d_s - \min\{\delta p_i(t), \delta p_j(t)\} - \|\boldsymbol{e_{p,i}}(t)\| - \|\boldsymbol{e_{p,j}}(t)\|.$$

Assume that the speed-tracking precisions under nominal conditions $\bar{e}_{v,i}$ and $\bar{e}_{v,j}$ are known in the trajectory generation phase, that the gains $k_{PF,i}$ and $k_{PF,j}$ used to compute the speed command in (2.11) are also known in the trajectory generation phase, and that the frames for vehicles i and j are bounded by a ball of radius $r_{f,i}$ and $r_{f,j}$ centered at the center of mass of each UAS. Then,

choosing

$$d_{p,i} = r_{f,i} + \frac{\bar{e}_{v,i}}{k_{PF,i}}, \qquad d_{p,j} = r_{f,j} + \frac{\bar{e}_{v,j}}{k_{PF,j}}$$

for the trajectory generation algorithm yields

$$\|\boldsymbol{p_i}(t) - \boldsymbol{p_j}(t)\| > r_{f,i} + r_{f,j} + (c_s - 1)\left(r_{f,i} + r_{f,j} + \frac{\bar{e}_{v,i}}{k_{PF,i}} + \frac{\bar{e}_{v,j}}{k_{PF,j}}\right) - \min\{\delta p_i(t), \delta p_j(t)\} - \|\boldsymbol{e_{p,i_0}}\| - \|\boldsymbol{e_{p,j_0}}\|.$$

Consequently, choosing a sufficiently large c_s ensures that vehicles i and j do not collide during the mission

$$c_s \geq 1 + \frac{\sup_t \left(\min\{\delta p_i(t), \delta p_j(t)\}\right) + \|\boldsymbol{e_{p,i_0}}\| + \|\boldsymbol{e_{p,j_0}}\|}{r_{f,i} + r_{f,j} + \frac{\bar{e}_{v,i}}{k_{PF,i}} + \frac{\bar{e}_{v,j}}{k_{PF,j}}} \implies \|\boldsymbol{p_i}(t) - \boldsymbol{p_j}(t)\| > r_{f,i} + r_{f,j}.$$

In the fraction above, the numerator contains all the errors that are not modeled in the trajectory generation phase, while the denominator contains all the errors accounted for in this phase.

6.4 Online Monitoring of Coordination and Temporal Errors

Consider now a human operator in charge of supervising a fleet of n UASs subject to coordination and temporal constraints. Estimating the coordination and temporal errors from the location of the vehicles, as well as the implications that these may have for the success of a mission, can be challenging for a single person. Hence, to aid human operators interpret and visualize abidance to the coordination and temporal constraints during the execution of a mission, two types of spider charts, shown in Figures 6.7 and 6.8, are proposed.

Figure 6.7 presents the coordination constraints plot, which compares the coordination state of each vehicle with all its cooperating peers. This is represented in $n(n-1)$ axes that are joined at the center of the chart. Each axis represents the coordination error $\xi_i - \xi_j$ with $i \neq j$, where i can be identified by the color of the filled circular marker, and j is

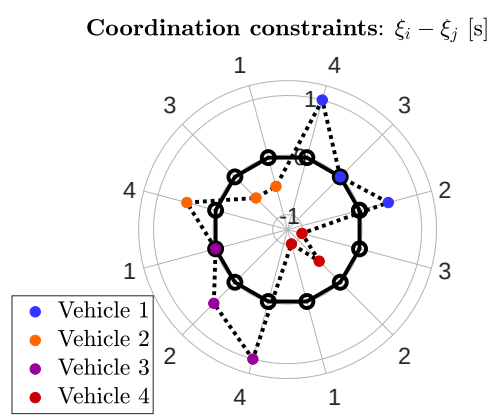


Figure 6.7: Online monitoring plot for the coordination errors.

defined by the number at the outermost end of the axis. For tight coordination constraints, the vehicles are tasked to drive the coordination errors to zero. This coordination goal is represented by the *zero polygon*, highlighted in black. As a result, if the marker associated with the coordination error $\xi_i - \xi_j$ lies outside the zero polygon, then the ith vehicle is ahead of the jth vehicle; whereas a marker inside the zero polygon indicates that the ith UAS is running behind when compared to the jth agent. For instance, in Figure 6.7 agent 1 is ahead agent 2, but perfectly coordinated with vehicle 3. Likewise, one can infer at a glance that agent 4 is behind all of its peers.

Figure 6.8 shows the temporal constraint plots for unenforced, relaxed, and strict temporal constraints. In these charts, the coordination state of each vehicle is compared with the reference state. This is represented in n axes that are joined at the center of the chart. Each axis represents the temporal error $\xi_i - \xi_R$, where i can be identified either by the color of the marker, or the number at the outermost end of the axis. Note that the temporal goals are represented differently depending on the type of temporal constraints imposed:

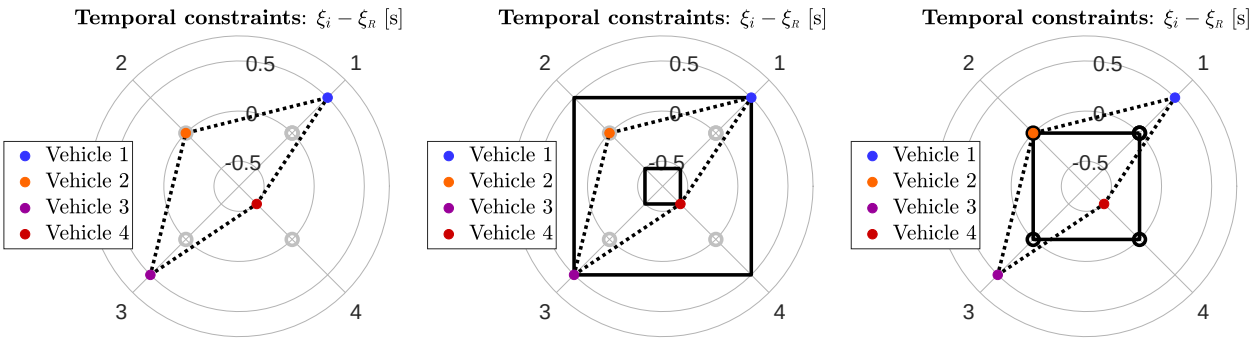
i) Unenforced: since the vehicles are not asked to observe ξ_R, the vertices of the zero polygon are represented by empty circular markers in light gray just for reference, see Figure 6.8a.

ii) Relaxed: since the temporal goal is to maintain $\xi_i - \xi_R$ within the interval $[-\Delta_t, +\Delta_t]$, an outer and an inner polygon are used to delimit the boundaries of the desired temporal window, see Figure 6.8b. The figure illustrates a mission with four agents and thus the inner and outer polygons are squares.

iii) Strict: the zero polygon is highlighted in black to denote that the temporal goal is to drive all $\xi_i - \xi_R$ to zero, see Figure 6.8c.

6.5 Simulation Results

This section presents simulation results for a time-critical cooperative mission through a cluttered urban-like environment with tight coordination constraints and three types of temporal constraints: unenforced, relaxed, and strict. As a result, this section analyzes and compares three simulation runs with the same mission design, initial conditions, control gains, and time-varying network



(a) Unenforced (b) Relaxed (c) Strict

Figure 6.8: Online monitoring plots for the temporal errors.

topology, but different temporal constraints. In these simulations, a group of eight UASs is tasked to converge to and follow their time-deconflicted trajectories while coordinating, to ensure that vehicles maintain a safe separation throughout the mission. The sequential arrival of all the UASs to their final destination in the center plaza, see Figure 6.18, marks the completion of the mission. In this scenario, coordination is safety critical since there are multiple narrow passages through which the UASs must pass sequentially. In fact, all vehicles must fly underneath a single arch to successfully complete the mission. Consequently, a lack of coordination in bottlenecks like this could lead to vehicle collisions. Figures 6.12 through 6.18 show snapshots of the mission, and online monitoring plots for the coordination and temporal errors at different times throughout the simulation. In these figures the top, middle, and bottom images always correspond to unenforced, relaxed, and strict temporal constraints, respectively.

The cooperating fleet is composed of eight heterogeneous multirotors: four quadcopters, two hexacopters, and two octocopters. The underlying dynamical model for each UAS is based on the work developed in [97, 98, 99]. It includes non-linear translational and rotational dynamics that are coupled as follows:

$$
\begin{aligned}
\dot{\boldsymbol{p}}_{\boldsymbol{i}}(t) &= \boldsymbol{v}_{\boldsymbol{i}}(t), \\
m_i \dot{\boldsymbol{v}}_{\boldsymbol{i}}(t) &= -f_{T,i}(t) \boldsymbol{R}^{\boldsymbol{\mathcal{I}}}_{\boldsymbol{\mathcal{B}},\boldsymbol{i}}(t) \boldsymbol{e_3} + m_i g \boldsymbol{e_3} + \boldsymbol{f}_{\boldsymbol{D},\boldsymbol{i}}(t), \\
\dot{\boldsymbol{R}}^{\boldsymbol{\mathcal{I}}}_{\boldsymbol{\mathcal{B}},\boldsymbol{i}}(t) &= \boldsymbol{R}^{\boldsymbol{\mathcal{I}}}_{\boldsymbol{\mathcal{B}},\boldsymbol{i}}(t) \left(\boldsymbol{\omega}^{\boldsymbol{\mathcal{B}}}_{\boldsymbol{i}}(t)\right)^{\wedge}, \\
\boldsymbol{J}_{\boldsymbol{i}} \dot{\boldsymbol{\omega}}^{\boldsymbol{\mathcal{B}}}_{\boldsymbol{i}}(t) &= -\boldsymbol{\omega}^{\boldsymbol{\mathcal{B}}}_{\boldsymbol{i}}(t) \times \boldsymbol{J}_{\boldsymbol{i}} \boldsymbol{\omega}^{\boldsymbol{\mathcal{B}}}_{\boldsymbol{i}}(t) + \boldsymbol{\tau}^{\boldsymbol{\mathcal{B}}}_{\boldsymbol{T},\boldsymbol{i}}(t) + \boldsymbol{\tau}^{\boldsymbol{\mathcal{B}}}_{\boldsymbol{D},\boldsymbol{i}}(t) + \boldsymbol{\tau}^{\boldsymbol{\mathcal{B}}}_{\boldsymbol{G},\boldsymbol{i}}(t),
\end{aligned}
$$

where the subindex $i \in \mathcal{I}$ denotes the agent identification number; $\boldsymbol{p_i} \in \mathbb{R}^3$ is the position of the center of mass of the UAS expressed in the inertial frame $\{\mathcal{I}\}$ (North-East-Down); $\boldsymbol{v_i} \in \mathbb{R}^3$ is the inertial velocity; $m_i \in \mathbb{R}$ is the mass; $f_{T,i} \in \mathbb{R}$ is the total thrust generated by the rotors; $\boldsymbol{R^{\mathcal{I}}_{\mathcal{B},i}} \in \mathbb{R}^{3\times 3}$ is the rotation matrix from the body frame $\{\mathcal{B}\}$ (Forward-Right-Down) to the inertial frame $\{\mathcal{I}\}$, see Figure 2.2; $\boldsymbol{e_3} := [0, 0, 1]^\top$; the gravitational acceleration is $g = 9.81\,\mathrm{m/s^2}$; $\boldsymbol{f_{D,i}} \in \mathbb{R}^3$ is the aerodynamic drag force expressed in $\{\mathcal{I}\}$; $\boldsymbol{\omega^{\mathcal{B}}_i} \in \mathbb{R}^3$ is the angular rate of the frame $\{\mathcal{B}\}$ with respect to $\{\mathcal{I}\}$ expressed in $\{\mathcal{B}\}$; $(\cdot)^\wedge$ denotes the hat map[1]; $\boldsymbol{J_i} \in \mathbb{R}^{3\times 3}$ is the moment of inertia of the ith UAS expressed in $\{\mathcal{B}\}$; $\boldsymbol{\tau^{\mathcal{B}}_{T,i}} \in \mathbb{R}^3$ is the total torque generated by the rotors expressed in $\{\mathcal{B}\}$; $\boldsymbol{\tau^{\mathcal{B}}_{D,i}} \in \mathbb{R}^3$ is the aerodynamic drag torque expressed in $\{\mathcal{B}\}$; and $\boldsymbol{\tau^{\mathcal{B}}_{G,i}} \in \mathbb{R}^3$ is the gyroscopic torque generated by the rotation of the rotors. The model used here includes rotational and translational drag, gyroscopic effects due to the rotation of the rotors, and coriolis effects. The blade flapping model proposed in [97] is not included for the sake of simplicity. Table 6.1 provides a summary of the characteristics of the UASs involved in the simulation, where $\boldsymbol{J_q}$, $\boldsymbol{J_h}$, and $\boldsymbol{J_o}$ denote the moment of inertia of the quadcopters, hexacopters, and octocopters, respectively

$$\boldsymbol{J_q} := \begin{bmatrix} 0.003960 & 0 & 0 \\ 0 & 0.003845 & 0 \\ 0 & 0 & 0.007350 \end{bmatrix}, \ \boldsymbol{J_h} := \begin{bmatrix} 0.004500 & 0 & 0 \\ 0 & 0.004320 & 0 \\ 0 & 0 & 0.008120 \end{bmatrix}, \ \boldsymbol{J_o} := \begin{bmatrix} 0.008000 & 0 & 0 \\ 0 & 0.007500 & 0 \\ 0 & 0 & 0.010000 \end{bmatrix}.$$

To track the speed command generated by the cooperative path-following algorithm in Equation (2.11), each UAS implements a cascaded non-linear speed-tracking controller whose design leverages control strategies from [100, 101, 102, 103, 104, 105, 106, 107].

Table 6.1: Vehicle configuration.

		Vehicle							
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Vehicle Configuration	**Peer type**	link	link	end	end	end	end	end	end
	Rotors	4	6	4	8	4	6	4	8
	$k_{PF,i}$ [1/s]	0.60	0.65	0.60	0.65	0.60	0.65	0.60	0.65
	m_i [kg]	0.55	0.70	0.55	0.85	0.55	0.70	0.55	0.85
	$\boldsymbol{J_i}$ [kg m^2]	$\boldsymbol{J_q}$	$\boldsymbol{J_h}$	$\boldsymbol{J_q}$	$\boldsymbol{J_o}$	$\boldsymbol{J_q}$	$\boldsymbol{J_h}$	$\boldsymbol{J_q}$	$\boldsymbol{J_o}$

The QoS of the communication network that supports the time-critical coordination algorithms

[1]Given a vector $\boldsymbol{\omega} := [\omega_x, \omega_y, \omega_z]^\top \in \mathbb{R}^3$, the hat map is defined as

$$(\boldsymbol{\omega})^\wedge = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}.$$

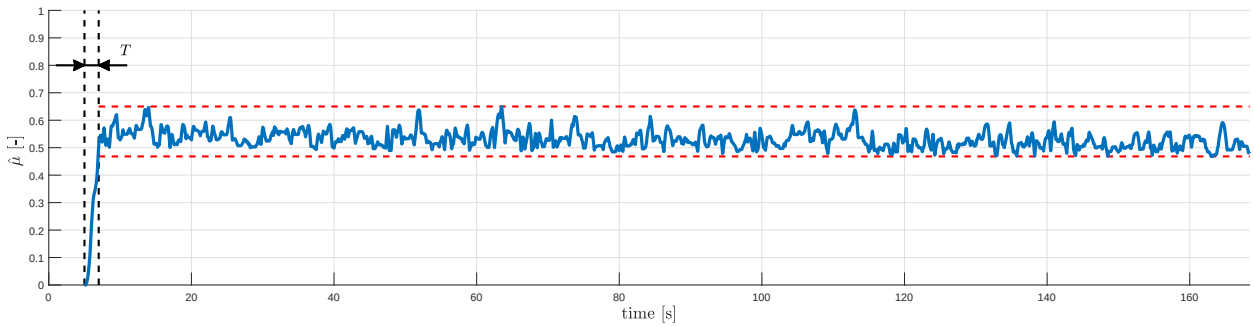


Figure 6.9: QoS of the network.

is estimated using Equation (5.13), with a fixed $T = 2.00\,\mathrm{s}$. To ensure the graph $\mathcal{G}(t)$ that represents the network satisfies Assumption 5, a pseudo-random sequence of Laplacian matrices that switches every $0.50\,\mathrm{s}$ was generated to maintain $\hat{\mu}$ within the range shown in Figure 6.9. The initial value for the coordination states is $\xi_i(0) = 0.00\,\mathrm{s}$ for all $i \in \mathcal{I}$, and the initial value for the integral states is $\chi_{i_0} = 1\,\mathrm{s/s}$ for all $i \in \mathcal{I}_e$. The control gains and dwell times are[2]

$$k_P = 0.20, \quad k_I = 0.08, \quad k_R = 0.40, \quad k_{e_p} = 2.50, \quad \tau_{R_0} = 0.00\,\mathrm{s}, \quad \text{and} \quad \tau_{R_1} = 0.50\,\mathrm{s}.$$

All the simulation runs include three events that significantly perturb the individual coordination and temporal errors. These events are introduced intentionally to evaluate how the time-critical coordination and path-following algorithms perform under realistic disturbances. The following list describes the nature and timing of these disturbances, as well as the effects on the cooperating fleet:

i) **Initial position error**: the cooperative path-following algorithm is engaged at $t = 5\,\mathrm{s}$. At that time, the actual positions of the UASs present an offset with respect to the desired positions, as indicated in Table 6.2. This can also be observed in the norm of the position error, shown in Figure 6.19. Figure 6.12 shows a snapshot of the mission shortly after the cooperative path-following algorithm is engaged. In this figure the vehicles are already tilted towards the desired position in an attempt to converge to the path. Note also that the initial position error has propagated through the coordination dynamics, leading to coordination and temporal errors in all simulation runs. Figure 6.13 illustrates how, after some time, the path-following algorithm makes the vehicles converge to their virtual targets. Simultaneously, the

[2]The tuning parameters given in Remarks 2, 4, 7 and Theorem 5 are conservative. For comparison and performance purposes some of the parameters chosen do not satisfy these bounds.

time-critical coordination algorithm synchronizes the virtual targets and enforces the desired temporal constraints. Three relevant symbols are visible in the 3D plot in Figure 6.13b:

- an empty circular marker ◯ denotes the position of the virtual target $\boldsymbol{p_{\mathcal{T},i}}(t) = \boldsymbol{p_{d,i}}(\xi_i(t))$;
- a filled star ★ represents the desired position given by the planned trajectory $\boldsymbol{p_{d,i}}(\xi_R(t))$;
- and two × symbols are used to delimit the trajectory segment that meets the relaxed temporal constraints at time t. Therefore, the trailing and leading × markers are located at $\boldsymbol{p_{d,i}}(\xi_R(t) - \Delta_t(t))$ and $\boldsymbol{p_{d,i}}(\xi_R(t) + \Delta_t(t))$, respectively.

Table 6.2: Actual and desired vehicle positions at $t = 5\,\mathrm{s}$.

		Vehicle							
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
$\boldsymbol{p_i}(t)$	x [m]	-13.50	-12.21	-5.93	-2.64	2.14	6.93	11.21	15.00
	y [m]	21.50	18.50	21.00	19.00	20.00	21.00	19.00	21.50
	z [m]	-1.50	-1.50	-1.50	-1.50	-1.50	-1.50	-1.50	-1.50
$\boldsymbol{p_{d,i}}(t)$	x [m]	-15.00	-10.71	-6.43	-2.14	2.14	6.43	10.71	15.00
	y [m]	20.00	20.00	20.00	20.00	20.00	20.00	20.00	20.00
	z [m]	-1.50	-1.50	-1.50	-1.50	-1.50	-1.50	-1.50	-1.50

ii) **Wind gust engaged**: a rectangular pulse with a magnitude of $8\,m/s$ is used to simulate a sudden wind gust. As depicted in Figure 6.10, the wind is engaged at $t = 20\,\mathrm{s}$. This disturbance is applied to all the UASs at the same time. However, the individual vehicle responses to the wind vary since the fleet is heterogeneous. Figure 6.14 shows a snapshot of the mission shortly after the wind gust is engaged. The arrows that appear next to each UAS icon denote the wind direction, which coincides with the direction of the drag force that pulls from each of the vehicles. This disturbance creates a position error that is clearly visible in Figure 6.19. As a result, Figure 6.14 shows an increase in the coordination and temporal errors. Notice that the overall effect of this wind gust is to push the fleet further behind the planned schedule. This can be inferred from the comparison of the temporal errors in Figures 6.13 and 6.14, or simply by looking at Figure 6.22. Note that while the wind is engaged the vehicles have difficulty tracking the virtual target and speed command with the same precision as with no wind, see Figures 6.19 and 6.20. However, as the speed-tracking controller of each UAS learns

to compensate for the wind, the position errors converge to a relatively small neighborhood of the origin. As a result, Figure 6.15 shows how the coordination errors return to a reasonably small neighborhood of the zero polygon, as expected for tight coordination constraints. Note also that in Figure 6.15b the temporal errors remain within the desired temporal window. In Figure 6.15c the temporal errors converge to a reasonably small neighborhood of the zero polygon, as expected for strict temporal constraints.

iii) **Wind gust disengaged**: the sudden removal of the wind gust at $t = 80\,\mathrm{s}$ causes the speed-tracking controller of every vehicle to overcompensate for a non-existent disturbance. As a result, the UASs overshoot their virtual targets, incurring in a position error shown in Figure 6.19. Figure 6.16 provides a snapshot of the mission shortly after the wind gust is removed. Again, the position error has propagated through the coordination dynamics, leading to an increase in the coordination errors. As for the temporal error, the overall effect of this disturbance is to advance the fleet forward along the mission, see Figure 6.22. After some time, the speed-tracking controllers learn that the wind has ceased. Then, the path-following algorithms are able to reduce the position errors. The time-critical coordination algorithm cancels out the effects of this disturbance on the coordination errors, which converge to a small neighborhood of the zero polygon, as shown in Figure 6.17. As expected, the temporal errors continue to drift linearly over time for unenforced temporal constraints, see Figure 6.22a or compare the temporal constraint plots in Figures 6.16a and 6.17a. For relaxed temporal constraints, the temporal link weights persistently switch between 0 and 1, to ensure the fleet remains within a small neighborhood of the temporal window, as shown in Figures 6.17b and 6.22b. Finally, if strict temporal constraints are enforced the temporal errors converge to a small neighborhood of zero, as illustrated in Figures 6.17c and 6.22c.

Figure 6.11 shows the collective control effort defined in Equation (5.14) for the three simulation runs. The three major events that lead to target-tracking errors can be easily identified at $t = 5\,\mathrm{s}$, $t = 20\,\mathrm{s}$, and $t = 80\,\mathrm{s}$. Three conclusions can be deduced from Figure 6.11. First, the coordination strategy with strict temporal constraints presents a more aggressive response to disturbances, since the increase in $\epsilon_u(t)$ at $t = 5\,\mathrm{s}$, $t = 20\,\mathrm{s}$, and $t = 80\,\mathrm{s}$ is larger for strict temporal constraints. Second, when the coordination strategy with relaxed temporal constraints is in mode Ø, see Figure 6.22b,

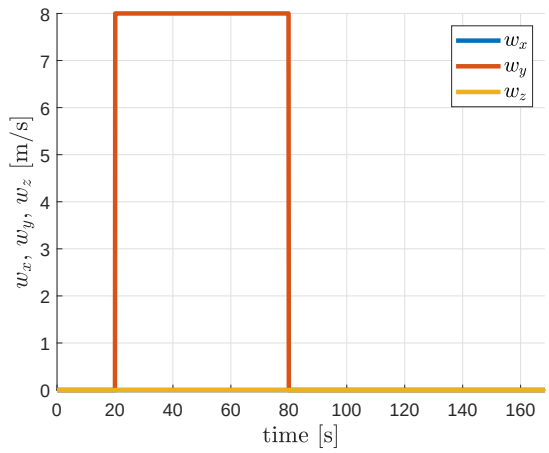


Figure 6.10: Wind profile.

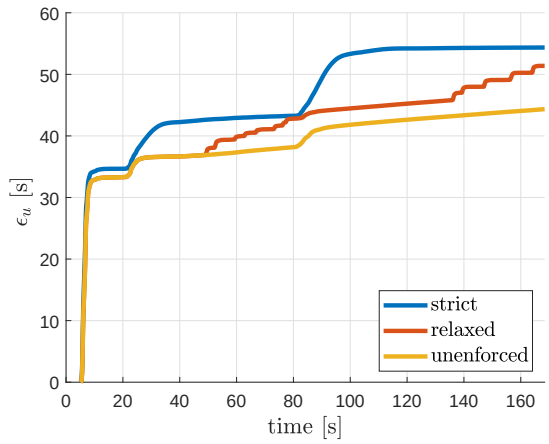


Figure 6.11: Coordination control effort.

the contribution to the control effort is the same as for unenforced temporal constraints, as expected. However, when system (6.3) switches to mode temporarily the control effort increases. Last, in the absence of disturbances the slope with which the control effort increases for unenforced temporal constraints is greater than that of strict temporal constraints. This is caused by the fact that the integral states under strict temporal constraints converge to a smaller neighborhood of the reference rate than under unenforced temporal constraints, see Figure 6.24.

Figure 6.21 shows the coordination error for unenforced, relaxed, and strict temporal constraints. In all simulation runs the coordination errors converge to a neighborhood of the origin as anticipated in Theorems 4, 5, and 6. Note that for relaxed temporal constraints the logic in Equation (6.4) results in the link peers switching their weights $\omega_{R_i}(t)$ persistently once the fleet arrives at the boundary of the desired temporal window, see Figure 6.22b. These switches are designed to pull the end peers inside the temporal window, but also lead to a temporary increase in the coordination errors in Figure 6.21b.

Figure 6.22 shows the temporal errors and the link weights for the three simulation runs. Note the effects of engaging and disengaging the wind gust at $t = 20\,\mathrm{s}$ and $t = 80\,\mathrm{s}$, respectively. As predicted in Theorem 4, the temporal error drifts linearly over time for unenforced temporal constraints, see Figure 6.22a. In the scenario with relaxed temporal constraints, the temporal error drifts linearly until the link peers reach the boundary of the temporal window, see Figure 6.22b between $t = 90\,\mathrm{s}$ and $t = 130\,\mathrm{s}$. During that time the switched system is in mode Ø. Then, the

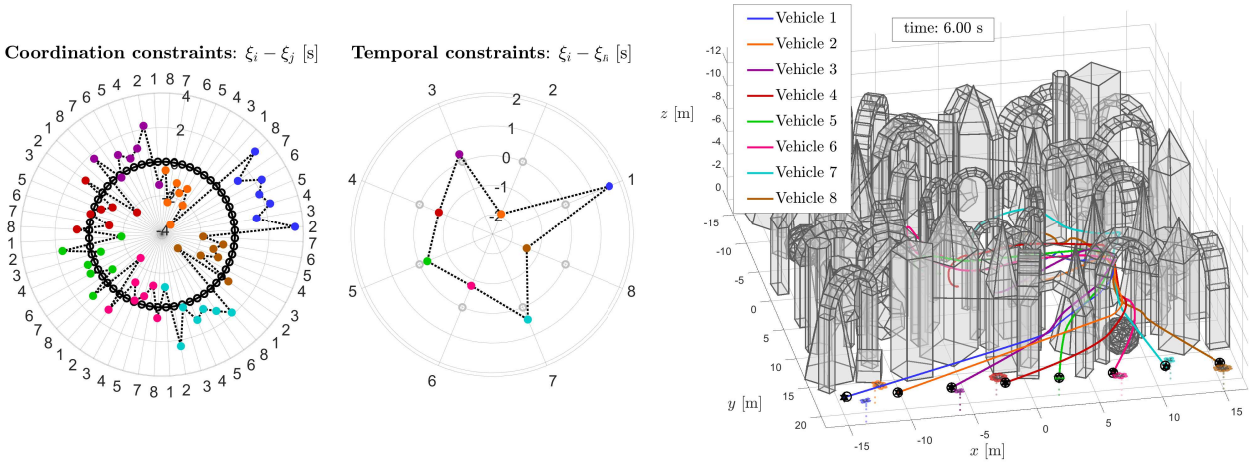
link weights $\omega_{R_i}(t)$ switch persistently between 0 and 1 to prevent the drift of the temporal errors. Consequently, the system alternates between modes Ø and ☁, ensuring that all temporal errors remain within a neighborhood of the temporal window, as anticipated in Corollary 2. Notice that the temporal window is set to a constant value $\Delta_t(t) \equiv 3\,\mathrm{s}$. Finally, Figure 6.22c confirms that the temporal errors under strict temporal constraints converge to a small neighborhood of the origin, as proven in Theorem 6. The effect of these behaviors can also be observed in the temporal evolution of the coordination states, shown in Figure 6.23.

Figure 6.24 shows the integral states of the end peers. Note that for unenforced temporal constraints the integral states do not exactly learn the mission rate, see Figure 6.24a. This is responsible for the predicted drift of the temporal errors. The same behavior can be observed for relaxed temporal constraints while the system is in mode Ø, see Figure 6.24b between $t = 90\,\mathrm{s}$ and $t = 130\,\mathrm{s}$. However, when the link peers reach the boundary of the temporal window the end peers are forced to persistently correct their integral states as the the weights $\omega_{R_i}(t)$ are set to 1. Finally, when enforcing strict temporal constraints the integral states converge to an even smaller neighborhood around the desired mission rate $\dot{\xi}_R$, as shown in Figure 6.24c.

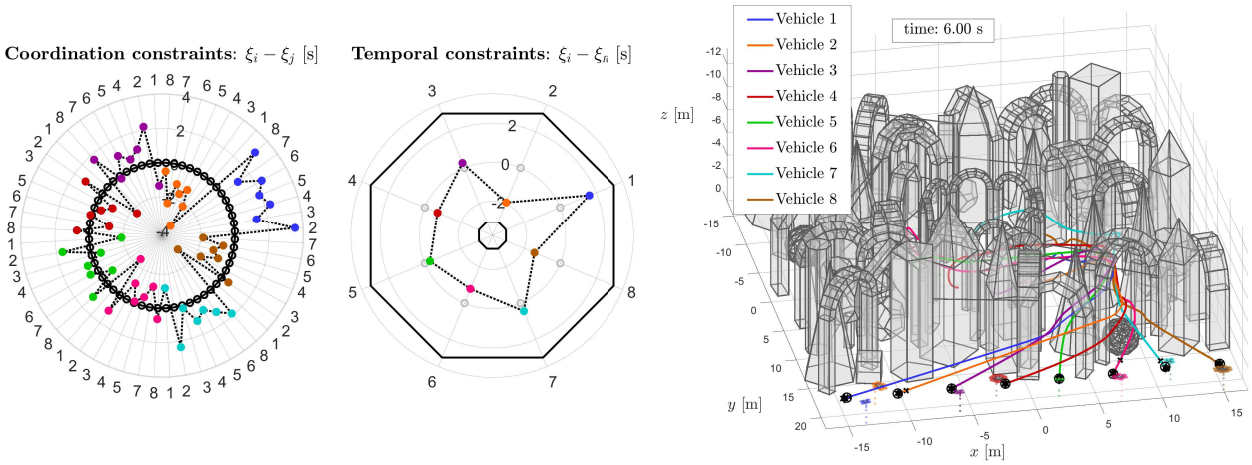
Figure 6.25 shows the coordination control signal. In all simulation runs the effects of the initial position errors and the wind gust are clearly visible at $t = 5\,\mathrm{s}$, $t = 20\,\mathrm{s}$, and $t = 80\,\mathrm{s}$. As expected with a switching control strategy, the changes in the link weights under relaxed temporal constraints create a sudden change in the control signal, see Figure 6.25b.

Figure 6.26 shows the norm of the desired velocity as computed during the trajectory generation phase, the norm of the velocity command, and the norm of the actual vehicle velocity. Note that for unenforced and relaxed temporal constraints there is a noticeable lag between the desired speed and the speed command. This is expected and caused by the delays in the progress of the mission, which grow linearly over time for unenforced temporal constraints, are bounded for relaxed temporal constraints, and are close to zero for strict temporal constraints.

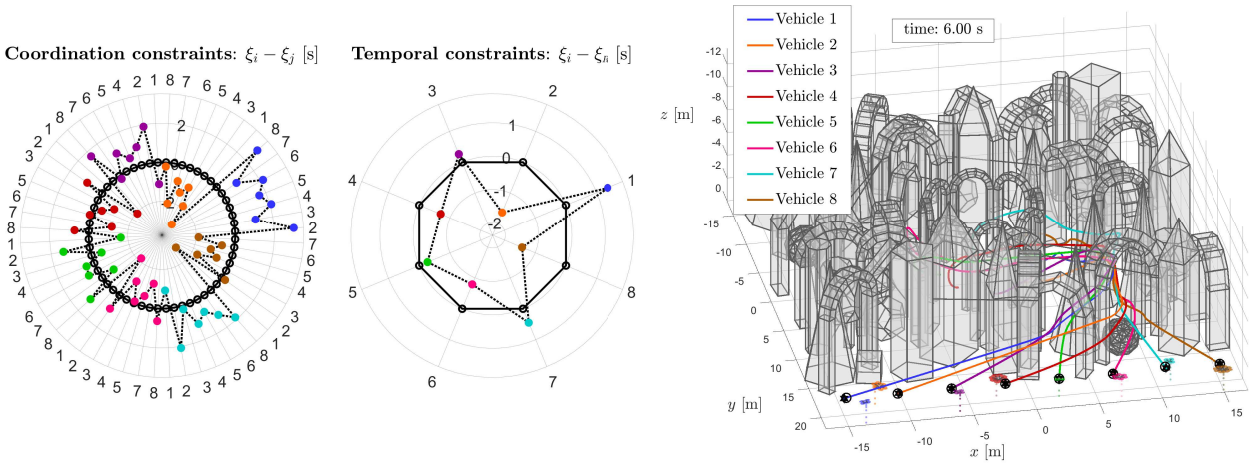
Figure 6.27 shows the temporal evolution of the distance among the centers of mass of the cooperating agents for the three coordination strategies discussed in this chapter. Note that the distance among vehicles is always greater than 0.50 m, which was established as the safe separation threshold and is marked by a dashed red line.



(a) Unenforced temporal constraints

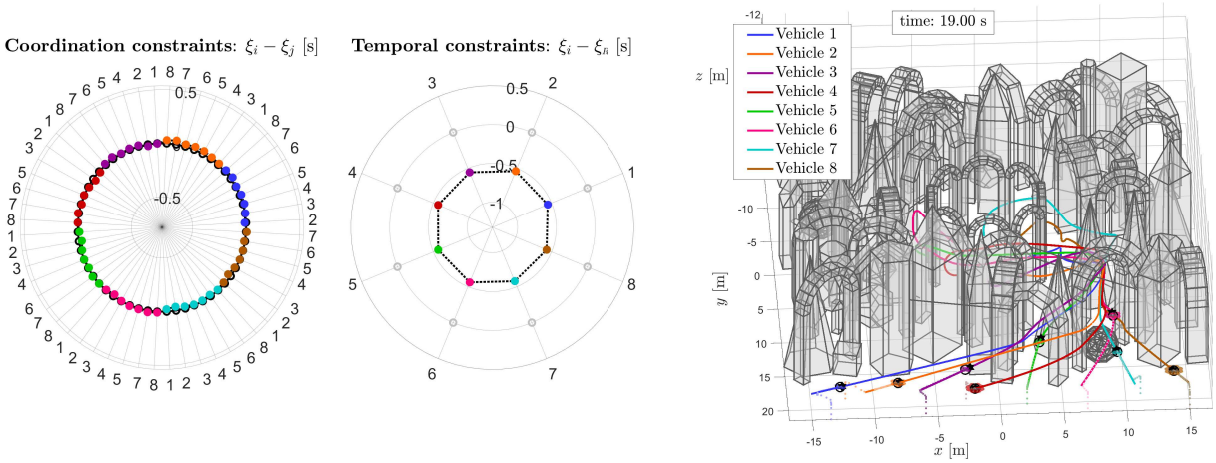


(b) Relaxed temporal constraints

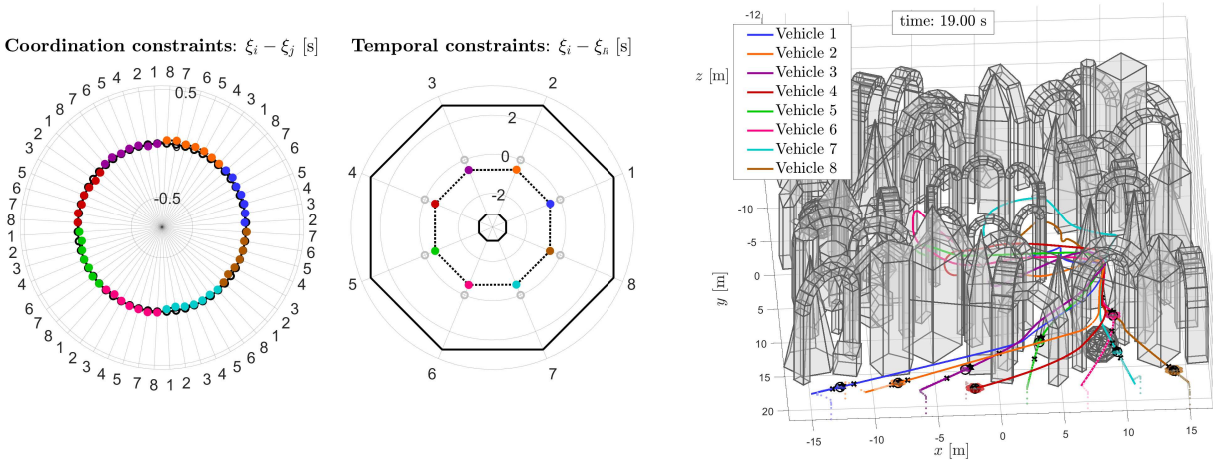


(c) Strict temporal constraints

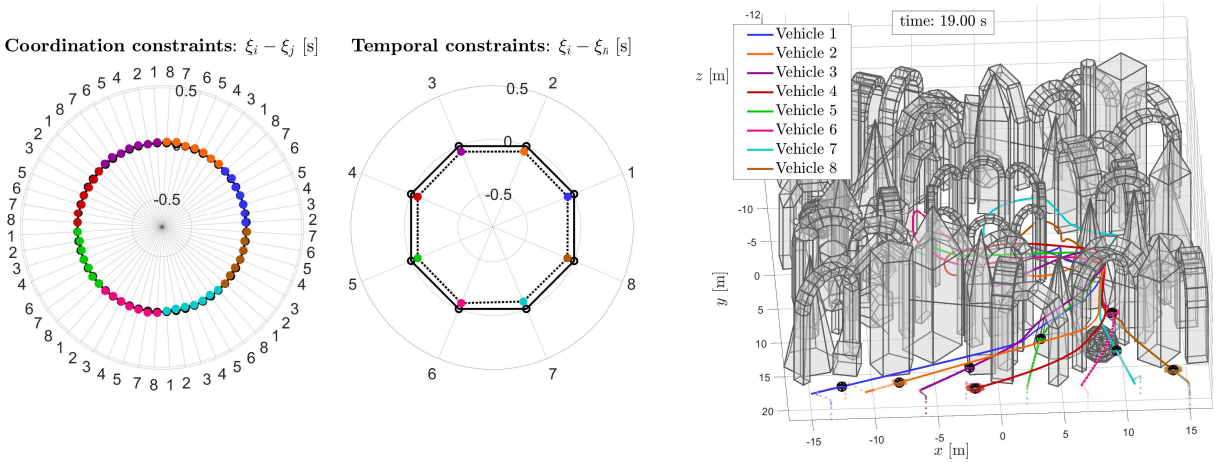
Figure 6.12: Simulation results 1 s after the time-critical coordination algorithm is engaged. The initial position errors lead to an increase in the coordination errors.



(a) Unenforced temporal constraints

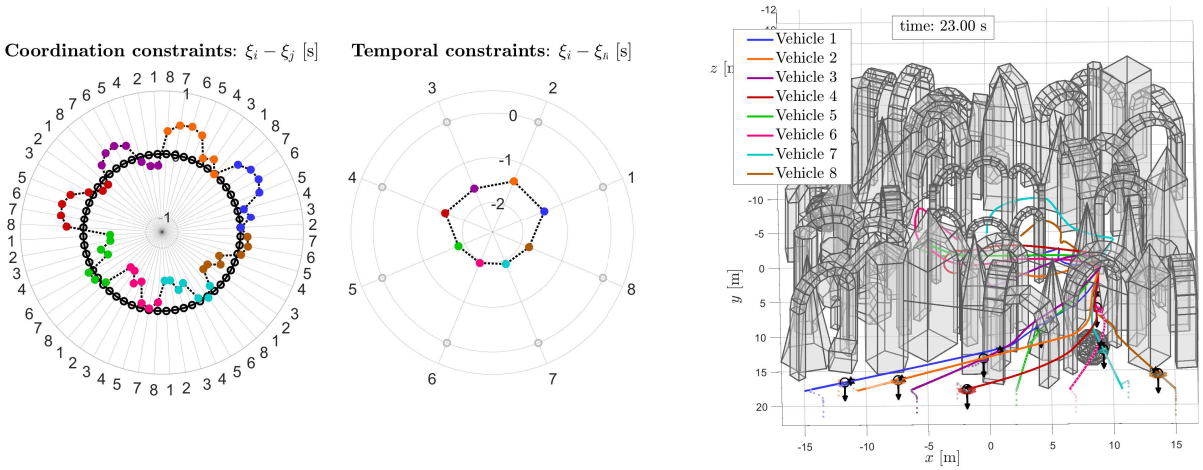


(b) Relaxed temporal constraints

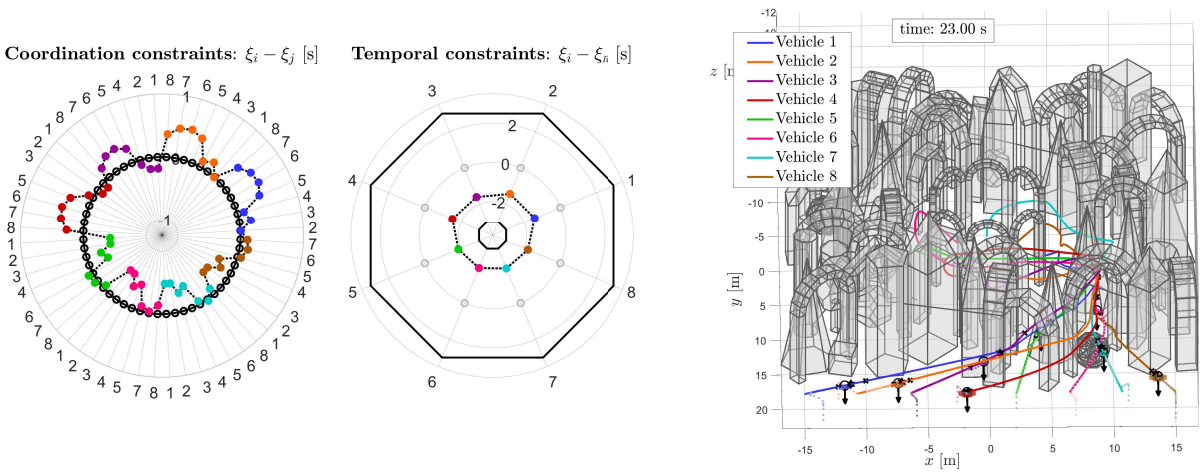


(c) Strict temporal constraints

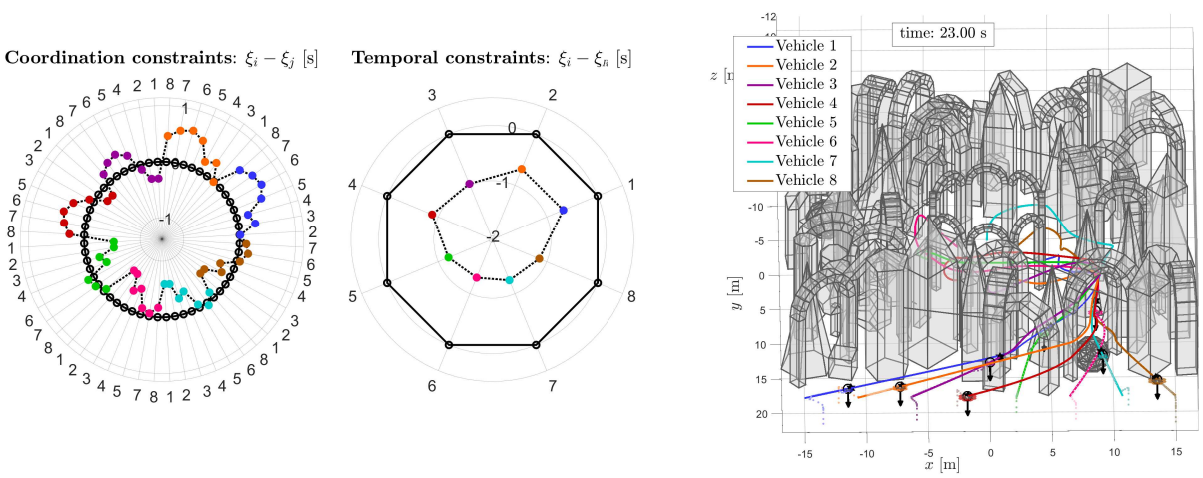
Figure 6.13: Simulation results 1 s before the wind gust is engaged. Vehicles adhere to their coordination and temporal constraints as expected.



(a) Unenforced temporal constraints

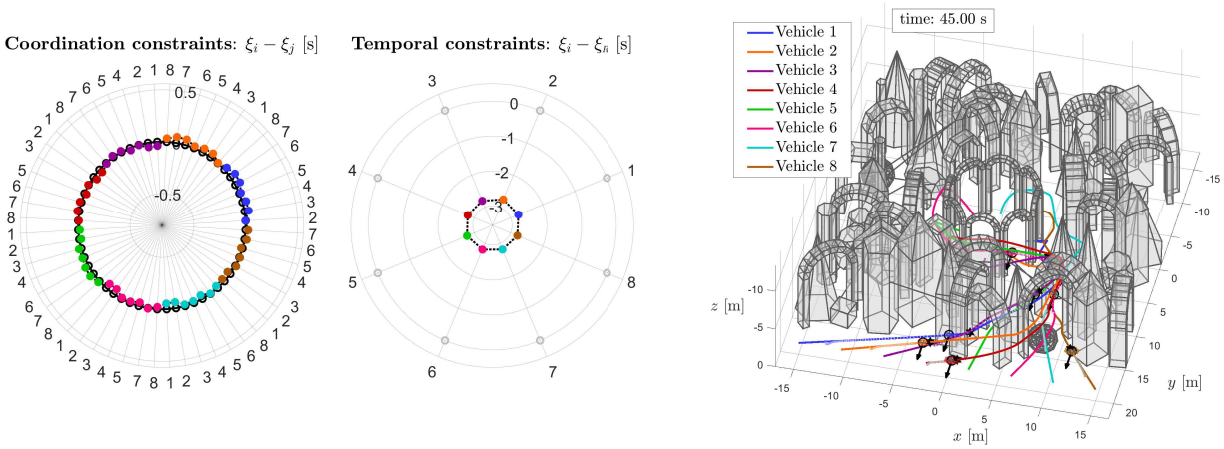


(b) Relaxed temporal constraints

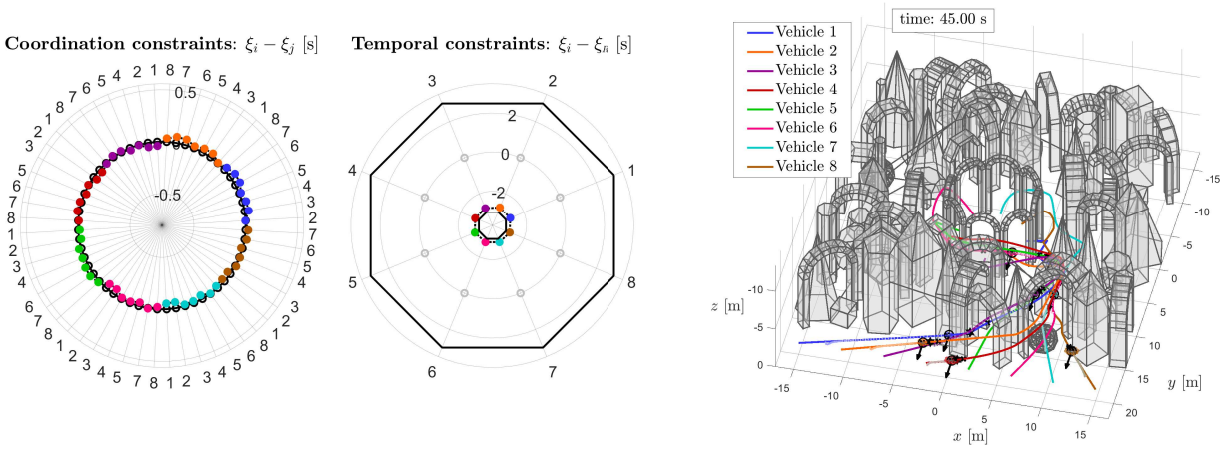


(c) Strict temporal constraints

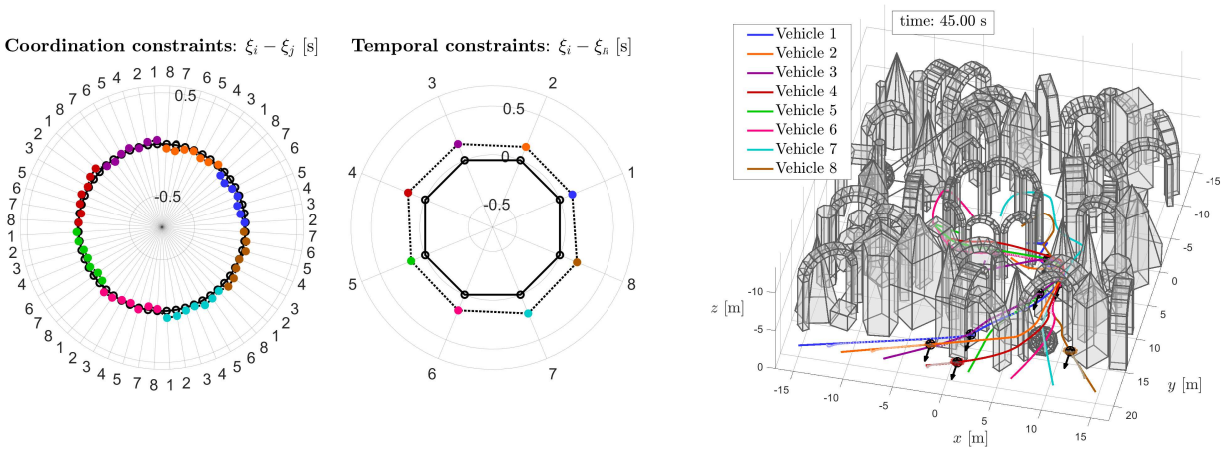
Figure 6.14: Simulation results 3 s after the wind gust is engaged. The wind leads to an increase in the coordination errors, and delays the fleet with respect to the planned schedule.



(a) Unenforced temporal constraints

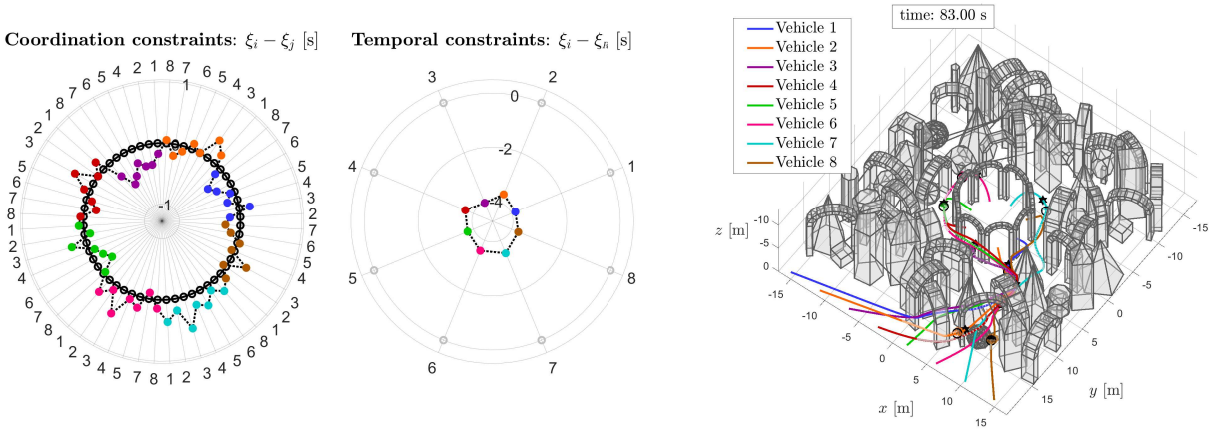


(b) Relaxed temporal constraints

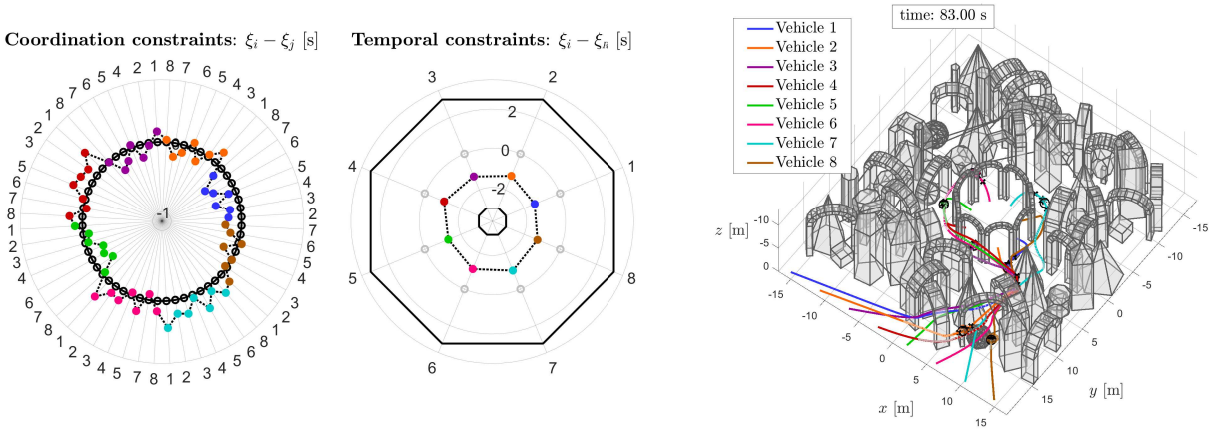


(c) Strict temporal constraints

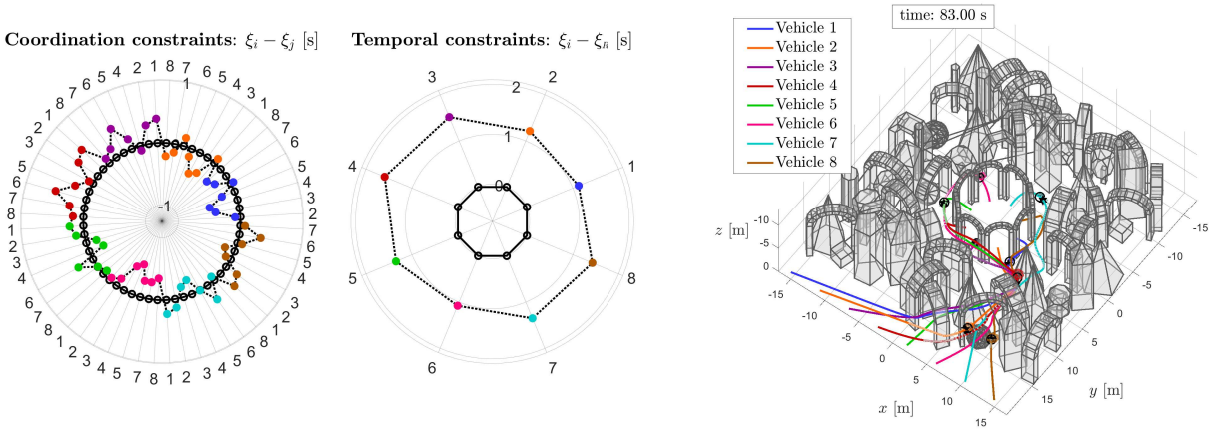
Figure 6.15: Simulation results 25 s after the wind gust is engaged. Vehicles adhere reasonably to their coordination and temporal constraints considering the challenge posed by the wind.



(a) Unenforced temporal constraints

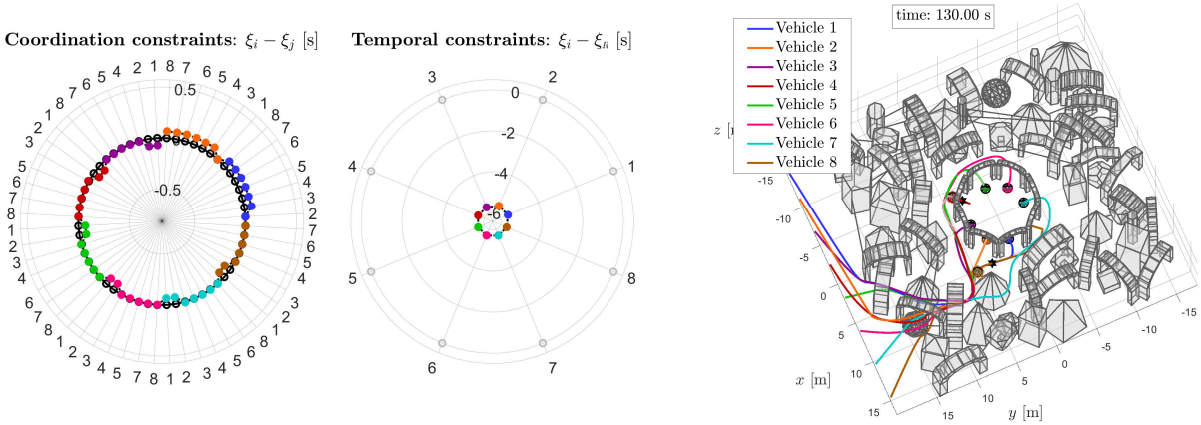


(b) Relaxed temporal constraints

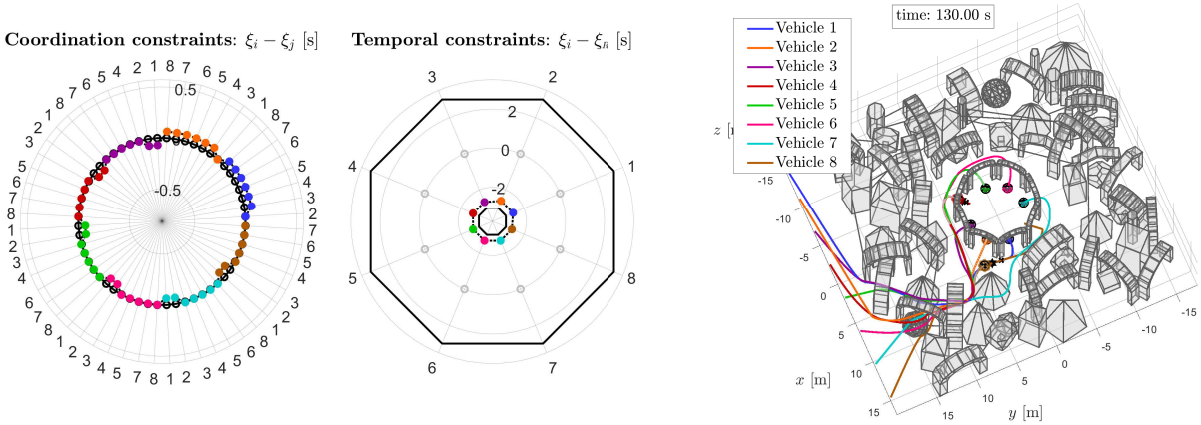


(c) Strict temporal constraints

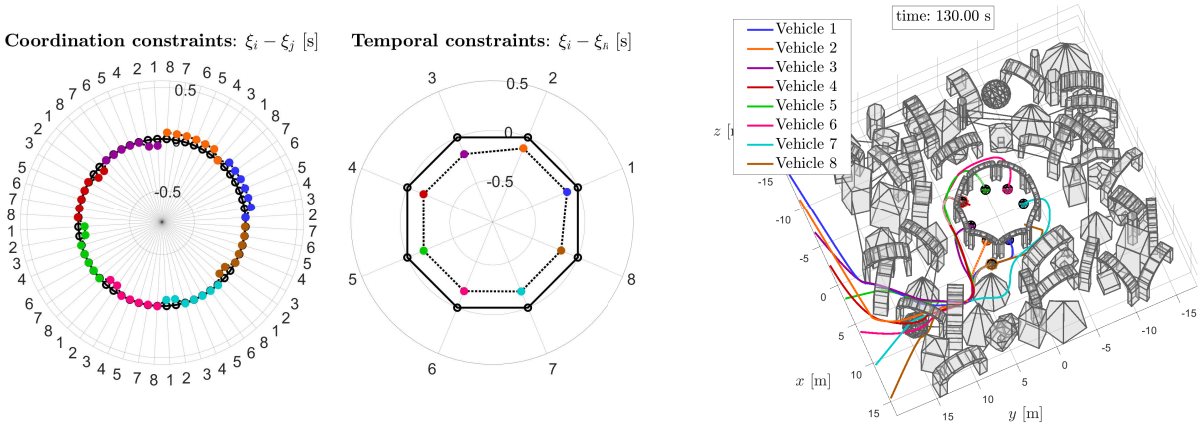
Figure 6.16: Simulation results 3 s after the wind gust is disengaged. The vehicles temporarily overshoot their virtual targets, which leads to an increase in the coordination errors.



(a) Unenforced temporal constraints

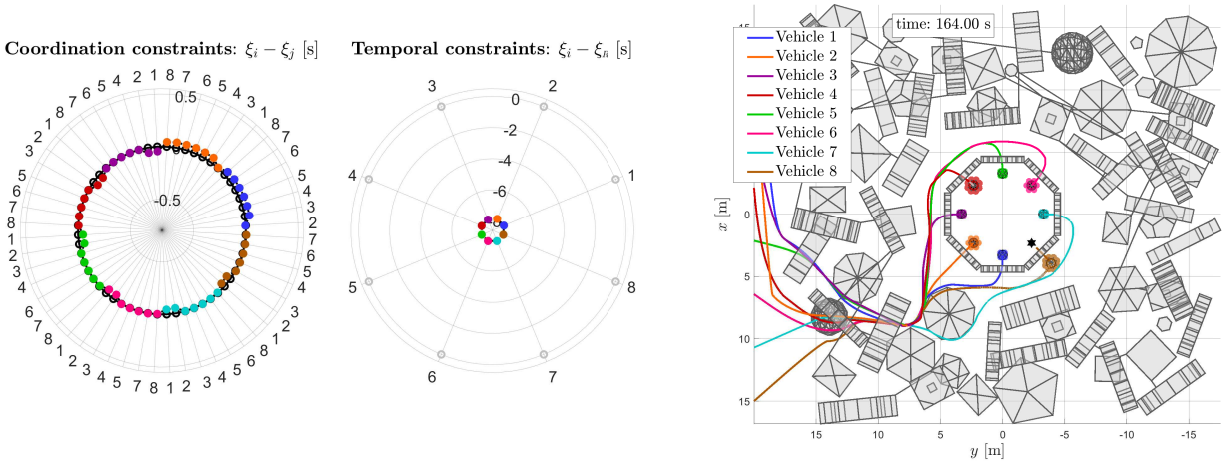


(b) Relaxed temporal constraints

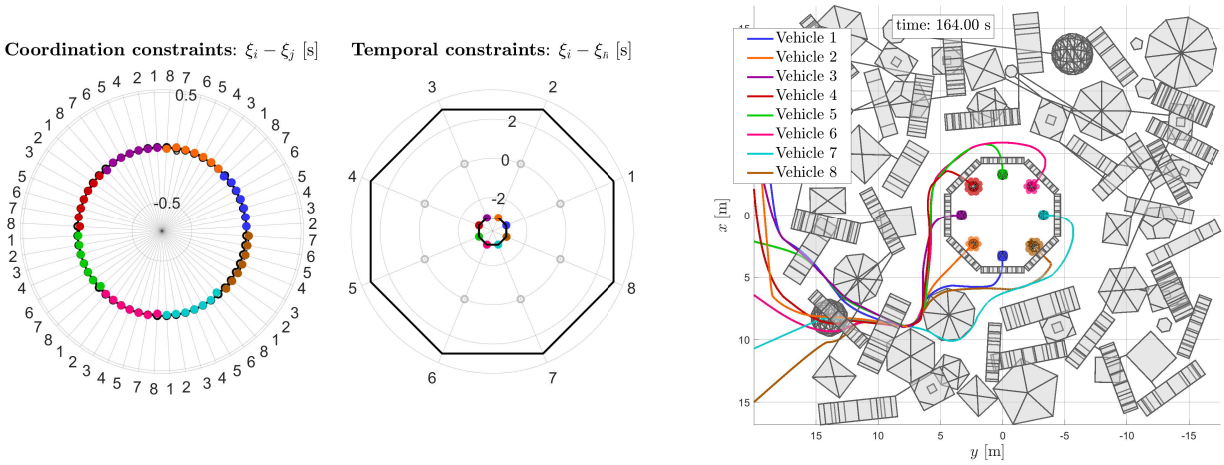


(c) Strict temporal constraints

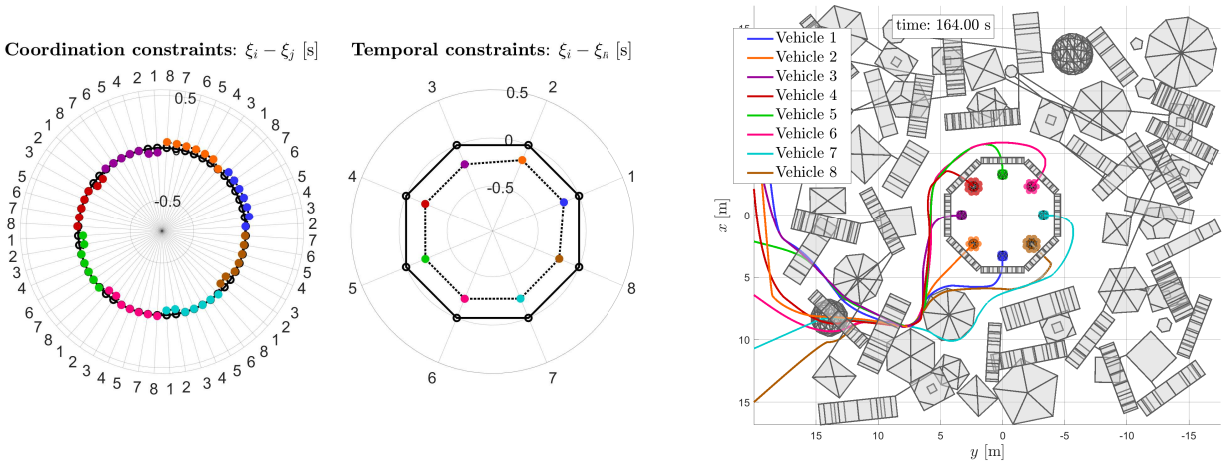
Figure 6.17: Simulation results 50 s after the wind gust is disengaged. Vehicles adhere to their coordination and temporal constraints as expected.



(a) Unenforced temporal constraints

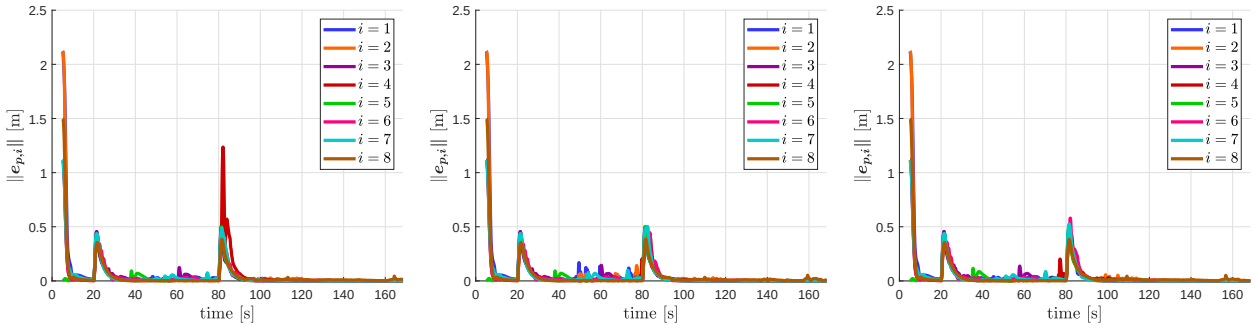


(b) Relaxed temporal constraints



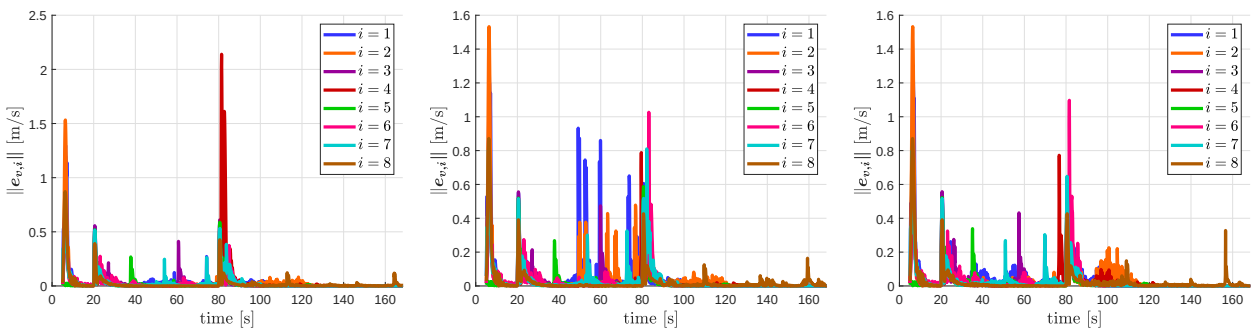
(c) Strict temporal constraints

Figure 6.18: Results at the conclusion of the simulation. Vehicles adhere to their coordination and temporal constraints as expected.



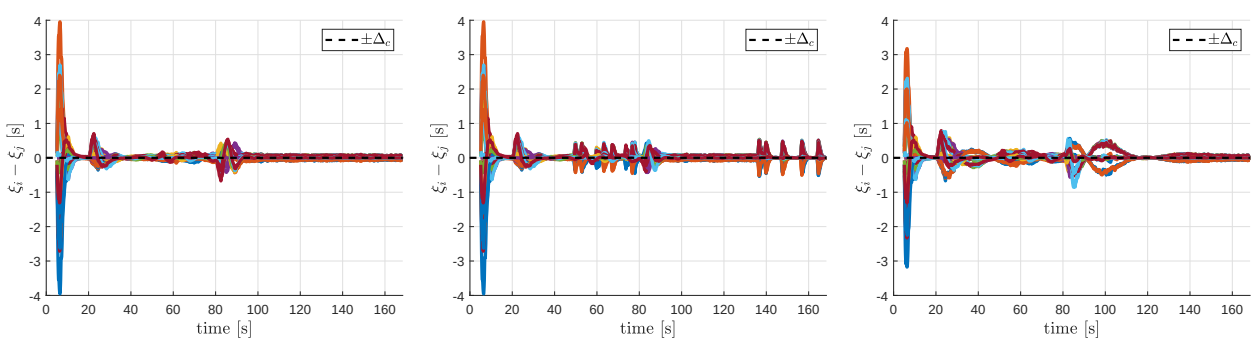
(a) Uneforced (b) Relaxed (c) Strict

Figure 6.19: Position-tracking errors.



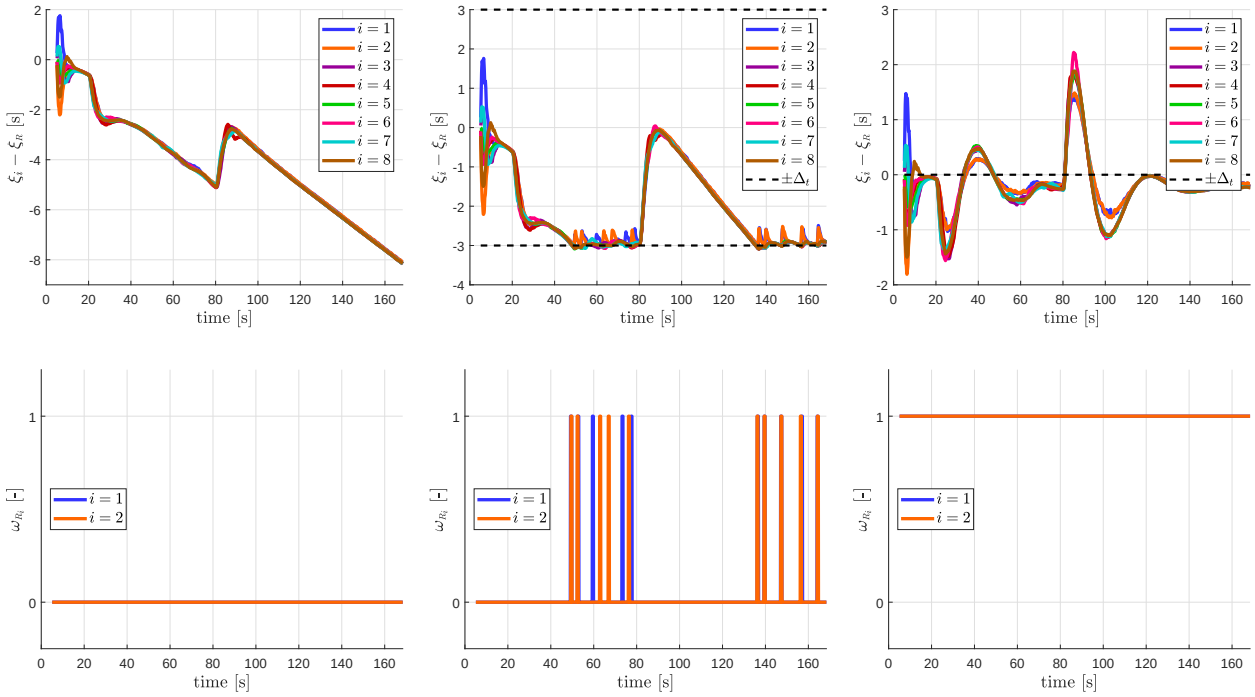
(a) Uneforced (b) Relaxed (c) Strict

Figure 6.20: Speed-tracking errors.



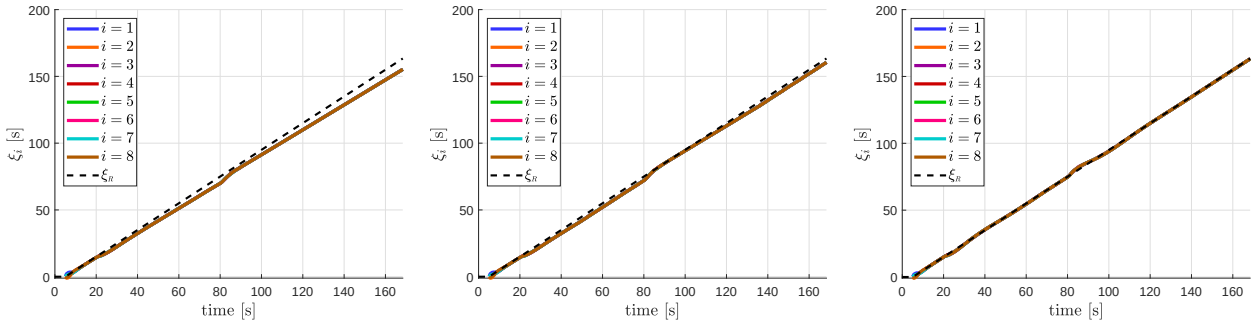
(a) Uneforced (b) Relaxed (c) Strict

Figure 6.21: Coordination errors.



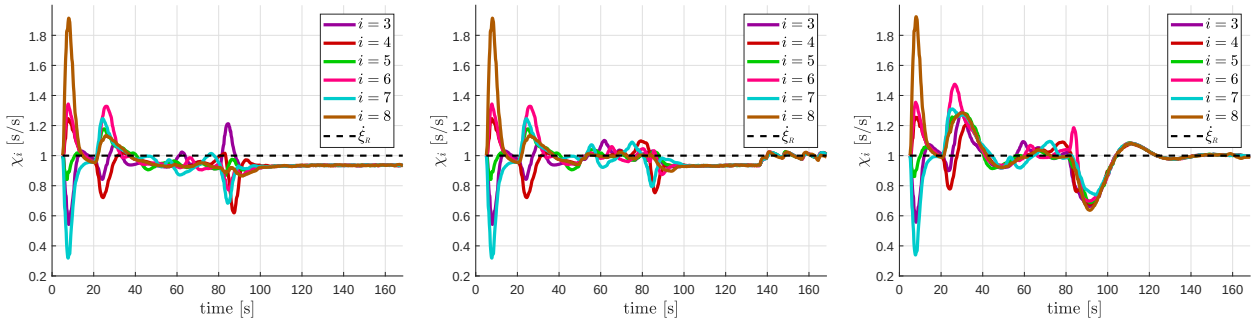
(a) Uneforced (b) Relaxed (c) Strict

Figure 6.22: Temporal errors (top) and temporal link weights (bottom).



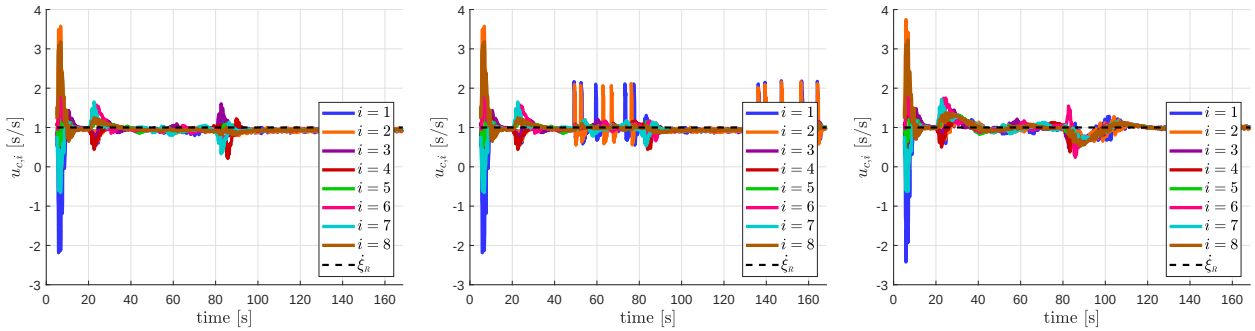
(a) Uneforced (b) Relaxed (c) Strict

Figure 6.23: Coordination states.



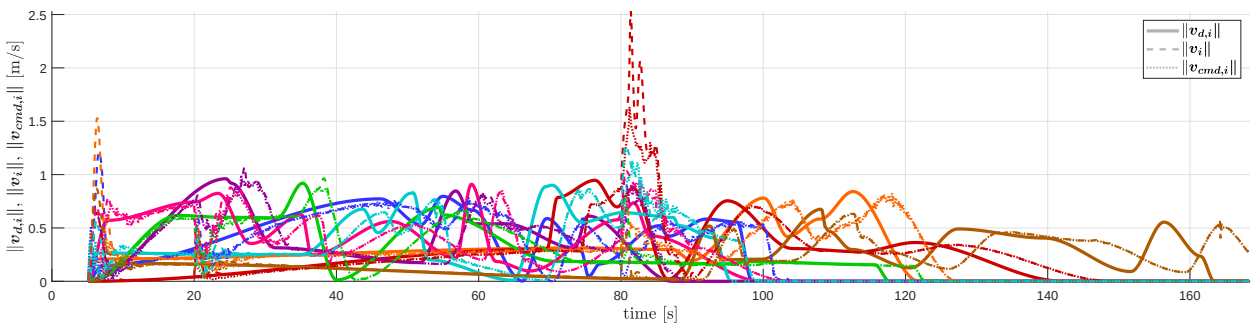
(a) Uneforced (b) Relaxed (c) Strict

Figure 6.24: Integral states.

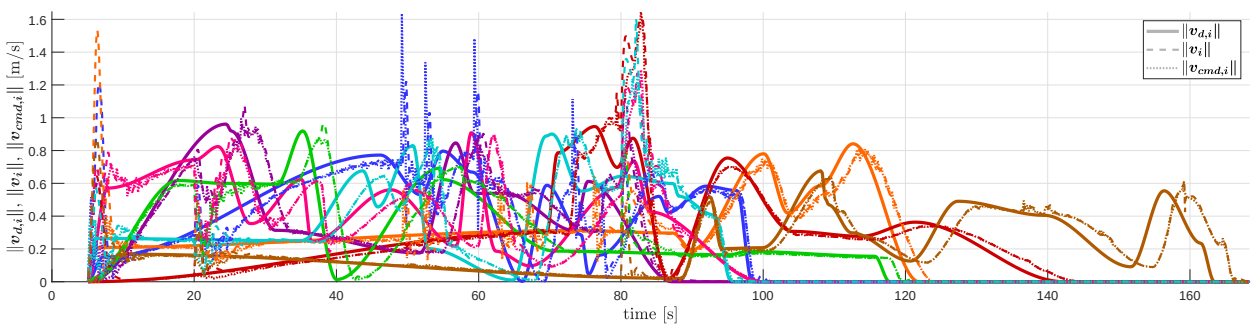


(a) Uneforced (b) Relaxed (c) Strict

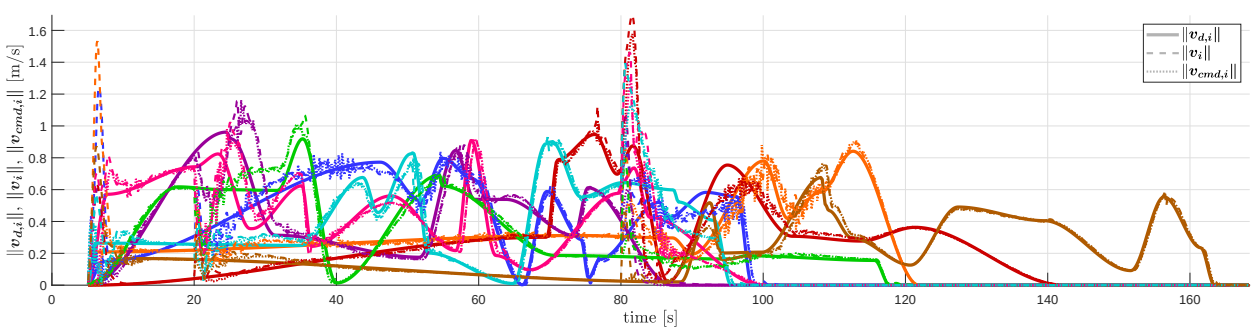
Figure 6.25: Coordination control signal.



(a) Uneforced

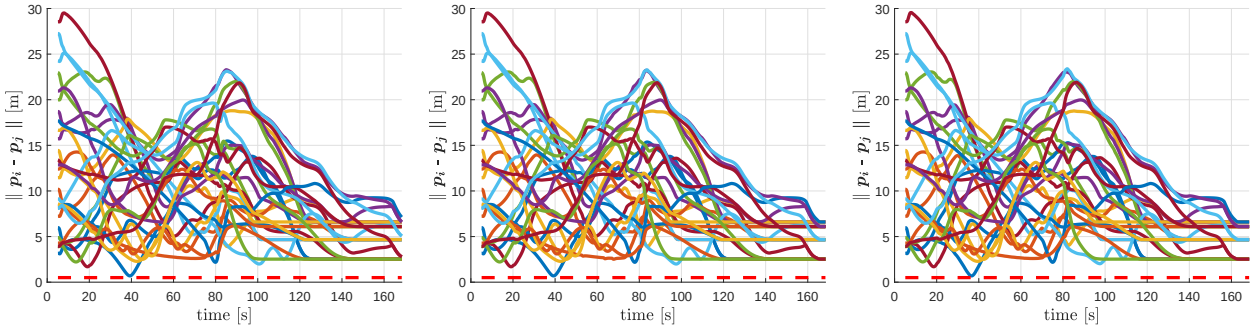


(b) Relaxed



(c) Strict

Figure 6.26: Speed profiles.



(a) Uneforced (b) Relaxed (c) Strict

Figure 6.27: Separation distance among vehicles.

Chapter 7

Loose Cooperative Path Following

This chapter proposes a distributed time-critical control law for loose coordination under a variety of temporal constraints. The algorithm requires each agent to exchange an additional variable over the network when compared to the tight coordination protocol. The design of a distributed estimator leads to this increase in the network traffic, and allows end peers to learn the reference state and rate without reaching consensus on their coordination states. To illustrate the performance of this protocol with non-ideal target-tracking conditions, simulation results for a group of eight cooperating peers are provided for unenforced, relaxed, and strict temporal specifications.

7.1 Learning the Reference Rate and State

As shown in Chapter 5, tight coordination relies on the consensus of the coordination states to make the end peers learn the reference rate. Recall that this is the purpose of the integral state $\chi_i(t)$. In the case of loose coordination, agreement in the coordination states will likely not be achieved, as specified in the control objective in Equation (2.15a). To overcome this difficulty, the fleet implements a distributed estimator for the reference state. To this end, agents exchange over the network a local *estimate of the reference state* $\hat{\xi}_{R,i}(t)$ with their cooperating peers. Since the link peers have access to the reference information, the following estimator is proposed

$$\dot{\hat{\xi}}_{R,i} = \dot{\xi}_R, \qquad \hat{\xi}_{R,i}(0) = \xi_R(0), \qquad i \in \mathcal{I}_\ell,$$

while the estimator for the end peers is

$$\dot{\hat{\xi}}_{R,i}(t) = -\hat{k}_P \sum_{j \in \mathcal{N}_i} \Big(\hat{\xi}_{R,i}(t) - \hat{\xi}_{R,j}(t) \Big) + \chi_i(t), \qquad \hat{\xi}_{R,i}(0) = \hat{\xi}_{R_0,i}, \qquad i \in \mathcal{I}_e,$$

$$\dot{\chi}_i(t) = -\hat{k}_I \sum_{j \in \mathcal{N}_i} \left(\hat{\xi}_{R,i}(t) - \hat{\xi}_{R,j}(t) \right), \qquad \chi_i(0) = \chi_{i_0}, \qquad i \in \mathcal{I}_e, \tag{7.1}$$

where $\hat{k}_P$ and $\hat{k}_I$ are control gains. This protocol still utilizes $\chi_i(t)$ to learn the reference rate $\dot{\xi}_R$. However, contrary to the protocol in (5.2), the evolution of $\chi_i(t)$ in Equation (7.1) does not depend on the type of temporal constraints imposed on the system. This resolves a slight inconvenience about the tight coordination protocol observed in Figure 5.8, where the rate of convergence of $\chi_i(t)$ to $\dot{\xi}_R$ slows down as the temporal constraints become more stringent. In addition, the protocol for tight coordination subjects the dynamics of $\chi_i(t)$ to the perturbations introduced by the target-tracking feedback $u_{\mathcal{T}_{\models},i}(t)$. Therefore, under non-ideal tracking conditions $\chi_i(t)$ does not fully converge to $\dot{\xi}_R$. This is proved in Theorems 4, 5, and 6 for unenforced, relaxed and strict temporal constraints, and confirmed through the simulation results presented in Chapter 6. Recall also, that the inability of $\chi_i(t)$ to accurately converge to $\dot{\xi}_R$ is fully responsible for the drift in the temporal error described in Theorem 4. Now, Equation (7.1) does not propagate the perturbation introduced by $u_{\mathcal{T}_{\models},i}(t)$ through the dynamics of the integral state. Consequently, it is expected that the proposed estimator defines a more accurate mechanism to teach $\xi_R(t)$ and $\dot{\xi}_R$ to the end peers, and does not lead to drift in the temporal error for unenforced temporal constraints under non-ideal target-tracking conditions, and the network Assumptions 3 through 5.

7.2 Distributed Coordination Control Law

To solve the multi-objective problem for loose coordination defined in (2.15), the coordination states $\xi_i(t)$ are no longer shared over the network. Instead, the agents exchange a local estimate of the *fleet-average coordination state* $\xi_{a,i}(t)$, together with the estimate of the reference state $\hat{\xi}_{R,i}(t)$. As a result, this protocol requires each cooperating peer to share one more variable over the network than the control law for tight coordination. However, the fact that $\xi_i(t)$ has become an internal state—only known by the ith agent—has cybersecurity implications. Recall that $\xi_i(t)$ governs the evolution of the ith virtual target, see Equation (2.8). Therefore, now not even a neighboring UAS that stores in memory the trajectory assigned to the ith vehicle can predict with accuracy the position of such agent at time t, unless additional information is shared over the network.

Accordingly, the following distributed protocol is proposed for the link peers:

$$\begin{aligned} u_{c,i}(t) &= -k_P\,\omega_{c_i}\,(\xi_i(t)-\xi_{a,i}(t)) - k_R\,\omega_{R_i}\,(\xi_i(t)-\xi_R(t)) + \dot{\xi}_R, \\ \dot{\xi}_{a,i}(t) &= -k_P\sum_{j\in\mathcal{N}_i}(\xi_{a,i}(t)-\xi_{a,j}(t)) - k_R\,\omega_{Ra_i}\,(\xi_{a,i}(t)-\xi_R(t)) + \dot{\xi}_R + u_{\mathcal{T}_{\models},i}(t), \end{aligned} \qquad i\in\mathcal{I}_\ell,$$

while the control law for the end peers is

$$\begin{aligned} u_{c,i}(t) &= -k_P\,\omega_{c_i}\,(\xi_i(t)-\xi_{a,i}(t)) - k_R\,\omega_{R_i}\left(\xi_i(t)-\hat{\xi}_{R,i}(t)\right) + \chi_i(t), \\ \dot{\xi}_{a,i}(t) &= -k_P\sum_{j\in\mathcal{N}_i}(\xi_{a,i}(t)-\xi_{a,j}(t)) - k_R\,\omega_{Ra_i}\left(\xi_{a,i}(t)-\hat{\xi}_{R,i}(t)\right) + \chi_i(t) + u_{\mathcal{T}_{\models},i}(t), \end{aligned} \qquad i\in\mathcal{I}_e,$$

where k_R, k_P, and k_I are control gains, ω_{R_i} and ω_{Ra_i} are *reference link weight*s that enforce the different types of temporal constraints; whereas ω_{c_i} is the *coordination link weight*, and is responsible for the loose coordination behavior. The long-track target-tracking error feedback $u_{\mathcal{T}_{\models},i}(t)$ is defined in Equation (2.13). Table 7.1 summarizes the logic design for ω_{R_i} and ω_{Ra_i} under unenforced, relaxed, and strict temporal constraints. For unenforced or strict temporal constraints ω_{R_i} and ω_{Ra_i} are identically equal to 0 or 1, respectively. However, for relaxed temporal constrains Table 7.1 defines a state-dependent switching logic subject to slow switching constraints. Here, $t^i_{R_s}$ and $t^i_{Ra_s}$ identify the last time the ith peer switched the value of ω_{R_i} and ω_{Ra_i}, respectively; $\omega^+_{R_i}(t^i_{R_s})$ and $\omega^+_{Ra_i}(t^i_{Ra_s})$ define the limit from the right of the corresponding weights at times $t^i_{R_s}$ and $t^i_{Ra_s}$. The dwell times τ_{R_0} and τ_{R_1} define the minimum times ω_{R_i} and ω_{Ra_i} will be set to 0 or 1, respectively. Finally, the switching law for the coordination link weights is

$$\omega_{c_i}(t) = \begin{cases} 1, & \text{if } |\xi_i-\xi_{a,i}| \geq \frac{\Delta_c(t)}{2} \;\wedge\; t-t^i_{c_s} > \tau_{c_0}, \\ 0, & \text{if } |\xi_i-\xi_{a,i}| < \frac{\Delta_c(t)}{2} \;\wedge\; t-t^i_{c_s} > \tau_{c_1}, \\ \omega^+_{c_i}(t^i_{c_s}), & \text{otherwise}, \end{cases} \tag{7.2}$$

where $t^i_{c_s}$ denotes the last time the ith agent switched the value of ω_{c_i}, and $\omega^+_{c_i}(t^i_{c_s})$ is the limit from the right at $t^i_{c_s}$. Again, to avoid Zeno behavior the dwell times τ_{c_0} and τ_{c_1} define the minimum times ω_{c_i} will be set to 0 or 1, respectively. Note that the logic for the coordination link weight depends on the half-width of the coordination window $\Delta_c(t)$, as opposed to the full width of the temporal window $\Delta_t(t)$ used in Table 7.1.

Table 7.1: Reference link-weight logic under different types of temporal constraints.

Unenforced	**Relaxed**	**Strict**
$\omega_{R_i}(t) \equiv 0$	$\omega_{R_i}(t) = \begin{cases} 1, & \text{if } \lvert\xi_i - \hat{\xi}_{R,i}\rvert \geq \Delta_t(t) \ \wedge \ t - t^i_{R_s} > \tau_{R0}, \\ 0, & \text{if } \lvert\xi_i - \hat{\xi}_{R,i}\rvert < \Delta_t(t) \ \wedge \ t - t^i_{R_s} > \tau_{R1}, \\ \omega^+_{R_i}(t^i_{R_s}), & \text{otherwise}, \end{cases}$	$\omega_{R_i}(t) \equiv 1$
$\omega_{Ra_i}(t) \equiv 0$	$\omega_{Ra_i}(t) = \begin{cases} 1, & \text{if } \lvert\xi_{a,i} - \hat{\xi}_{R,i}\rvert \geq \Delta_t(t) \ \wedge \ t - t^i_{Ra_s} > \tau_{R0}, \\ 0, & \text{if } \lvert\xi_{a,i} - \hat{\xi}_{R,i}\rvert < \Delta_t(t) \ \wedge \ t - t^i_{Ra_s} > \tau_{R1}, \\ \omega^+_{Ra_i}(t^i_{Ra_s}), & \text{otherwise}, \end{cases}$	$\omega_{Ra_i}(t) \equiv 1$

7.3 Online Monitoring of Coordination Errors

Section 6.4 proposed a set of plots to enhance operator awareness of the coordination and temporal state of the mission. The same spider plots presented in Section 6.4 are used to monitor the coordination and temporal errors in the simulation section of this chapter. Figure 7.1 introduces a few modifications to the coordination chart to represent loose coordination constraints. Again, given a fleet of n UASs participating in a cooperative mission, a chart with $n(n-1)$ axes that are joined at the center is used to represent the coordination constraints. Each axis represents the coordination error $\xi_i - \xi_j$ with $i \neq j$, where i is identified by the color of the filled circular marker, and j is defined by the number at the outermost end of each axis. In Figure 7.1, the vertices of the zero polygon are represented by empty circular markers in light gray, just for reference. If the marker associated with the error $\xi_i - \xi_j$ lies outside the zero polygon, then the ith peer is ahead of the jth peer; whereas a marker inside the zero polygon indicates that the ith UAS is running behind when compared to the jth agent. Since the coordination goal is to maintain $\xi_i - \xi_j$ within the interval $[-\Delta_c(t), +\Delta_c(t)]$, an outer and inner polygon are used to delimit the boundaries of the temporal window, highlighted in black in Figure 7.1. In this example, vehicle 1 is ahead of vehicles 2 and 4, and perfectly coordinated with vehicle 3. However, vehicle 1 meets its coordination

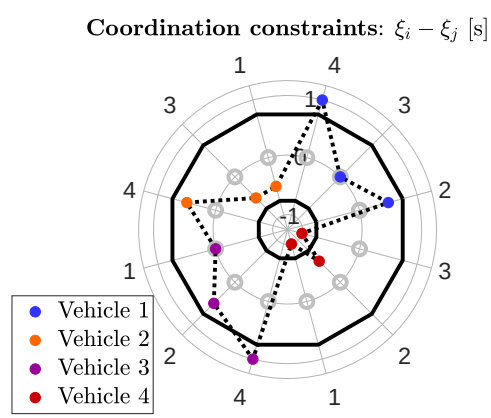


Figure 7.1: Online monitoring plot for the coordination errors.

constraints with vehicles 2 and 3, but violates the coordination constraints with vehicle 4, since the corresponding marker does not lie between the inner and outer polygons. The following section leverages this plot to present simulation results for loose coordination with unenforced, relaxed, and tight temporal constraints.

7.4 Simulation Results

This section presents simulation results for the same cooperative mission as in Section 6.5, but with loose coordination and three types of temporal constraints: unenforced, relaxed, and strict. As a result, this section compares three simulation runs with the same mission design, initial conditions, time-varying network topology, wind disturbances, link and end peers, vehicle frames, underlying vehicle dynamics, speed-tracking controllers, and control gains as in Section 6.5. The additional control knobs introduced by the loose coordination protocol are

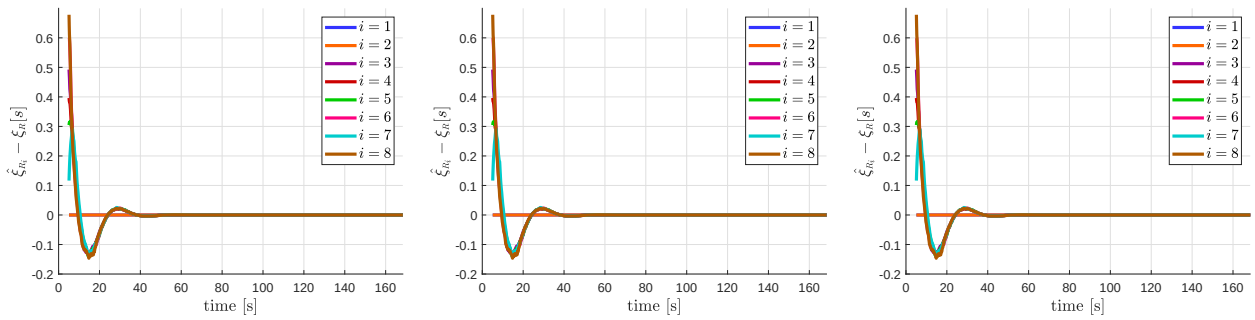
$$\hat{k}_P = 0.20, \quad \hat{k}_I = 0.05, \quad \tau_{c_0} = 0.00\,\text{s}, \quad \text{and} \quad \tau_{c_1} = 2.00\,\text{s}.$$

The initial conditions for the additional states introduced by the loose coordination protocol are

$$\hat{\boldsymbol{\xi}}_{\boldsymbol{R}}(0) = [0.00,\ 0.00,\ 0.50,\ 0.40,\ 0.30,\ 0.60,\ 0.10,\ 0.70]^\top\ \text{s},$$

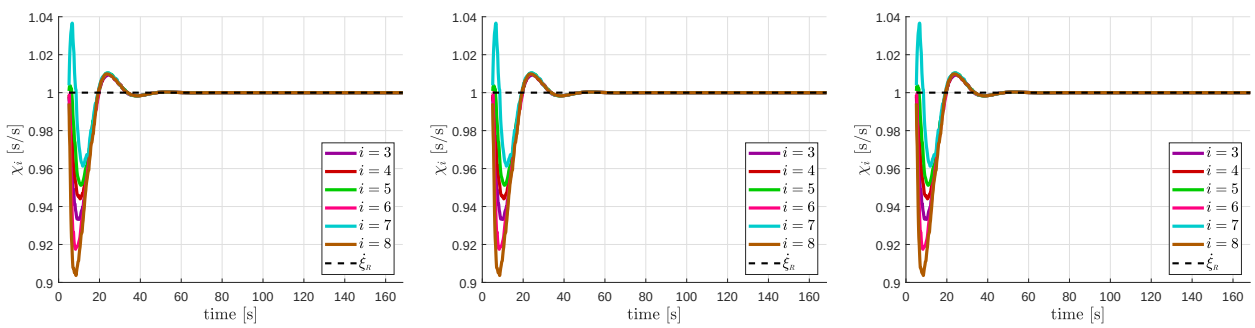
where $\hat{\boldsymbol{\xi}}_{\boldsymbol{R}}(t) := \left[\hat{\xi}_{R,1}(t), \ldots, \hat{\xi}_{R,n}(t)\right]^\top$, and $\xi_{a,i} = 0.00\,\text{s}$ for all $i \in \mathcal{I}$. Recall that agents 1 and 2 in Section 6.5 are defined as link peers. Note that the initial conditions above indicate that $\hat{\xi}_{R,i}(t)$ is not initialized at $\xi_R(0)$ for all end peers. This discrepancy is introduced intentionally to observe the behavior of the new reference state estimator. Figure 7.2 shows the estimation error. As expected, the estimate of the reference state $\hat{\xi}_{R,i}(t)$ converges to $\xi_R(t)$ for all the UASs. On the other hand, Figure 7.3 shows the temporal evolution of the integral states. As anticipated in Section 7.1, the integral states accurately learn the reference rate $\dot{\xi}_R$, and their responses are invariant to the different types of temporal constraints, unlike the tight coordination protocol.

As in Section 6.5, all the simulation runs include three events that significantly perturb the individual coordination and temporal errors. These events are also introduced intentionally to



(a) Uneforced (b) Relaxed (c) Strict

Figure 7.2: Reference state estimation errors.



(a) Uneforced (b) Relaxed (c) Strict

Figure 7.3: Integral states.

observe how the time-critical coordination and path-following algorithms perform under realistic disturbances, and serve to organize the chronology of the mission in three main sections:

i) **Initial position error**: the cooperative path-following algorithm is engaged at $t = 5\,\mathrm{s}$. At this time, the positions of the UASs present an offset with respect to the desired positions in the trajectory, as illustrated in Figure 7.12. Figure 7.5 shows a snapshot of the mission shortly after the cooperative path-following algorithm is engaged. As expected, the multirotors are tilted towards their virtual targets in an attempt to reduce the position error and converge to the path. Recall that the location of the virtual targets is depicted by an empty circular marker ◯, whereas the planned location as computed during the trajectory generation phase is marked by a filled star ★. In Figure 7.5, the target-tracking feedback $u_{\mathcal{T}_{\models},i}(t)$ has propagated the initial position error through the coordination dynamics, leading to coordination and temporal errors that in some cases surpass the desired values. Figure 7.6 illustrates how, after some time, the path-following controller is able to reduce the position errors, and the

distributed coordination protocol drives the coordination states to the coordination window. Note also that the vehicles adhere reasonably to their temporal constraints. An additional symbol is visible in Figure 7.6b. The two $\times$ markers delimit the trajectory segment that meets the desired temporal constraints. At this time, all virtual targets are located between their $\times$ markers, denoting that all vehicles meet their temporal constraints.

ii) **Wind gust engaged**: the same wind gust as in Section 6.5 is engaged at $t = 20\,\mathrm{s}$. Figure 7.7 shows a snapshot of the mission shortly after the wind gust is engaged. The arrows that appear next to each UAS icon indicate the wind direction. As expected, this disturbance induces a position error, as illustrated in Figure 7.12. The overall effect of this wind gust is to delay the progress of the fleet when compared to the planned schedule. This can be inferred from the plunge in the temporal errors at $t = 20\,\mathrm{s}$, shown in Figure 7.16. Note also that while the wind is engaged the underlying speed-tracking controller has difficulty tracking the speed command with the same precision as without wind, see Figure 7.13 between $t = 20\,\mathrm{s}$ and $t = 80\,\mathrm{s}$. However, as the speed-tracking controller learns to partially compensate for the wind the position errors shrink to a smaller neighborhood of the origin, see Figure 7.12 between $t = 20\,\mathrm{s}$ and $t = 80\,\mathrm{s}$. Figure 7.8 shows that after some time the coordination and temporal errors converge back to nominal values.

iii) **Wind gust disengaged**: the sudden removal of the wind gust at $t = 80\,\mathrm{s}$ causes the underlying speed-tracking controller to overcompensate for a non-existent disturbance. As a result, the UASs overshoot their virtual targets, incurring in the position error illustrated in Figure 7.12. The overall effect of this disturbance is to push the vehicles ahead along the mission. This can be observed in the sudden increase in the temporal errors depicted in Figure 7.16 at $t = 80\,\mathrm{s}$. Figure 7.9 shows a snapshot of the mission shortly after the wind gust is disengaged. The effects of this disturbance are more evident in Figure 7.9c where some of the coordination errors exceed the desired coordination window, and all the temporal errors lie outside the zero polygon. Figure 7.10 shows that eventually the fleet overcomes this disturbance and negotiates over the network to meet their coordination and temporal constraints with precision. Finally, Figure 7.11 shows a snapshot of the cooperating effort at the end of the mission, where the coordination and temporal assignments are met successfully.

Figure 7.4 shows the collective control effort defined in Equation (5.14) for the three simulation runs in this chapter. The three major events that lead to coordination and temporal errors can be clearly identified at $t = 5\,\mathrm{s}$, $t = 20\,\mathrm{s}$, and $t = 80\,\mathrm{s}$ by a steep increase in $\epsilon_u(t)$. This is also clearly visible in the coordination control signals in Figure 7.14. Three conclusions follow from the comparison of the control effort for tight and loose coordination in Figures 6.11 and 7.4. First, as with tight coordination, the strategies with more stringent temporal constraints in Figure 7.4 present a more aggressive response to disturbances, which yields a greater $\epsilon_u(t)$. Second, $\epsilon_u(t)$ is much smaller for the loose coordination strategies. These results indicate that to reduce the collective coordination control effort operators should choose the least stringent set of coordination and temporal constraints that guarantee the successful completion of the mission. Third, while $\epsilon_u(t)$ increases linearly in the absence of disturbances for tight coordination, see Figure 6.11, $\epsilon_u(t)$ plateaus for loose coordination constraints, see Figure 7.4. This is one of the intended effects of the reference state estimator introduced in Section 7.1. Recall that under non-ideal target-tracking conditions the integral state $\chi_i(t)$ converges to a small neighborhood of the origin for tight coordination. This yields the linear increase in $\epsilon_u(t)$, shown in Figure 6.11. However, with the reference state estimator the integral state $\chi_i(t)$ converges to $\dot{\xi}_R$. As a result, $\epsilon_u(t)$ remains constant after the external disturbances cease.

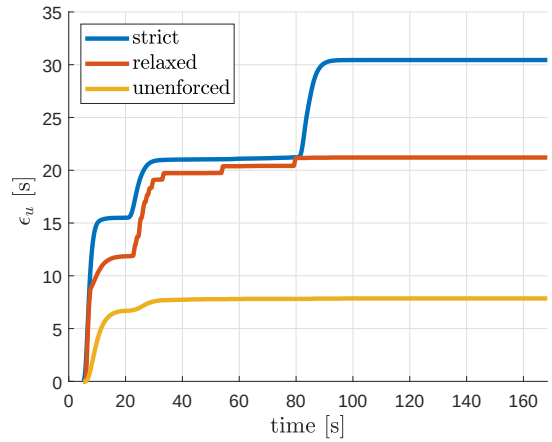


Figure 7.4: Coordination control effort.

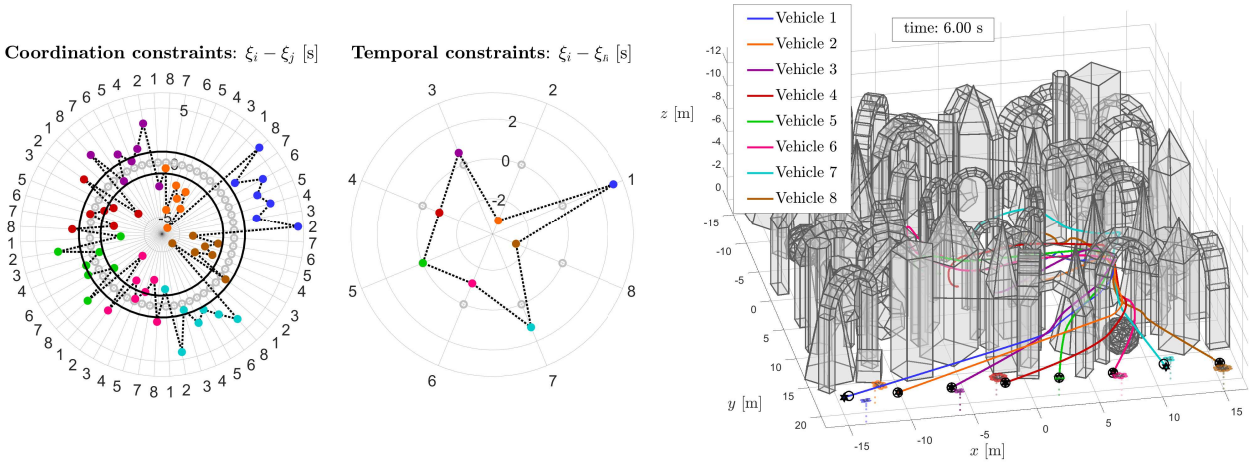
Figure 7.15 presents the temporal evolution of the coordination errors $\xi_i(t) - \xi_j(t)$, the distance to the local estimate of the fleet-average coordination state $\xi_i(t) - \xi_{a,i}(t)$, and the coordination link weights ω_{c_i}. In all simulation runs the coordination errors converge to the desired coordination window. However, in the presence of strict temporal constraints the coordination errors also converge to a small neighborhood of the origin. In this case, the coordination states are tasked to track the reference state, and thus all agents reach consensus on their coordination states too. This result begs the question: What is the difference between tight and loose coordination with

strict temporal constraints? The answer is two fold. First, the end result is similar. All agents reach consensus on the coordination states and track their reference state with some precision. Second, the difference is tactical. Under tight coordination the system is instilled a sense of urgency, and puts in an additional control effort to drive the coordination errors to zero. This is useful when vehicles fly in very close proximity, and tight coordination is safety critical. On the other hand, loose coordination implies that a precise agreement on the coordination states is not safety critical, as long as the coordination errors are maintained within some bounds. This is common in far and mid-proximity operations. However, the strict temporal constraints indicate that the time of arrival at certain locations is of importance for the success of the mission. As intended by the logic in Equation (7.2), the coordination weights in Figure 7.15 switch their value to 1 when $|\xi_i(t) - \xi_{a,i}(t)| \geq \Delta_c/2$ since, $\tau_{c_0} = 0.00\,\mathrm{s}$.

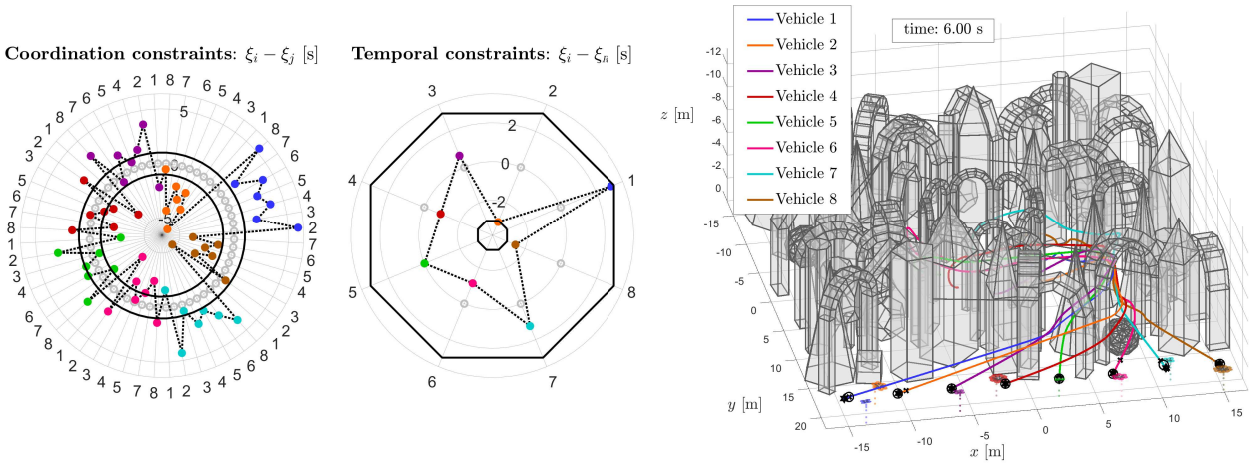
Figure 7.16 shows the time evolution of the temporal errors $\xi_i(t) - \xi_R(t)$, the estimate of the temporal errors $\xi_i(t) - \hat{\xi}_{R,i}(t)$, and the reference link weights ω_{R_i}. In all simulation runs $\xi_i(t) - \xi_R(t)$ converge to a small neighborhood of the desired set, marked by the black dashed lines. Notice the effects of engaging and disengaging the wind gust at $t = 20\,\mathrm{s}$ and $t = 80\,\mathrm{s}$, respectively. As intended by the logic in Table 7.1, for unenforced and strict temporal constraints the reference link weights are identically 0 or 1, respectively. However, for relaxed temporal constraints the reference weights change their value to 1 as soon as $|\xi_i(t) - \hat{\xi}_{R,i}(t)| \geq \Delta_t$, since $\tau_{R_0} = 0.00\,\mathrm{s}$.

Figure 7.17 shows the norm of the desired velocity as computed during the trajectory generation phase, the norm of the velocity command, and the norm of the actual vehicle velocity. Note that for unenforced and relaxed temporal constraints there is an offset between the desired speed and the speed command. This is expected and caused by the discrepancies between the coordination and reference states, illustrated in Figure 7.16.

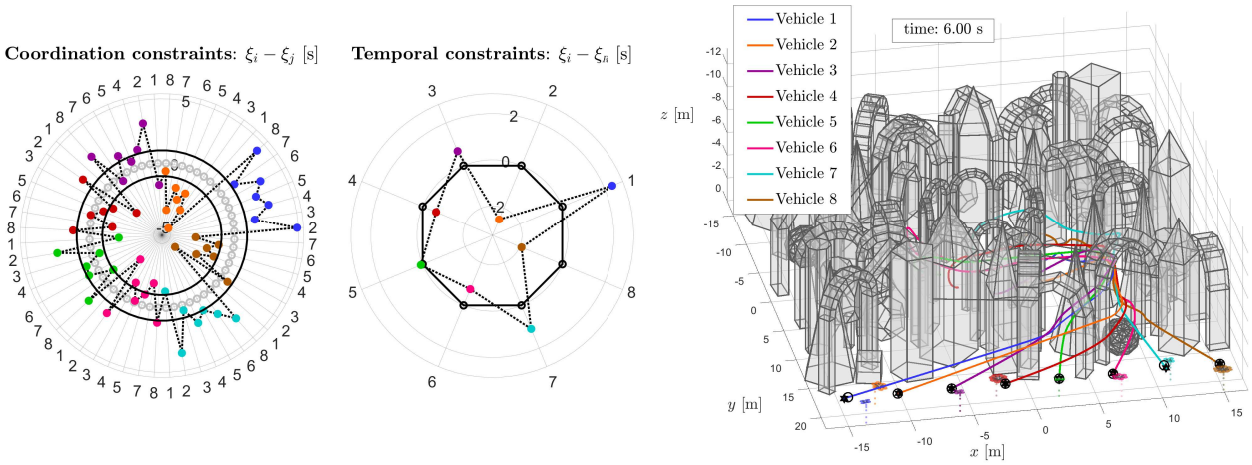
Finally, Figure 7.18 shows the temporal evolution of the distance among the centers of mass of the cooperating agents for the three coordination strategies discussed in this chapter. Note that the distance among vehicles is always greater than 0.50 m, which was established as the safe separation threshold and is marked by a dashed red line.



(a) Unenforced temporal constraints

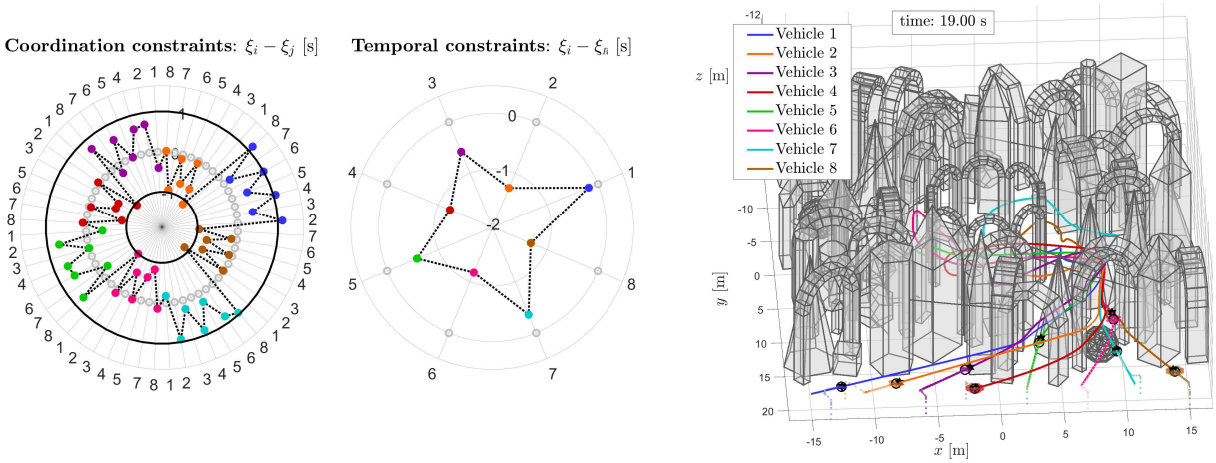


(b) Relaxed temporal constraints

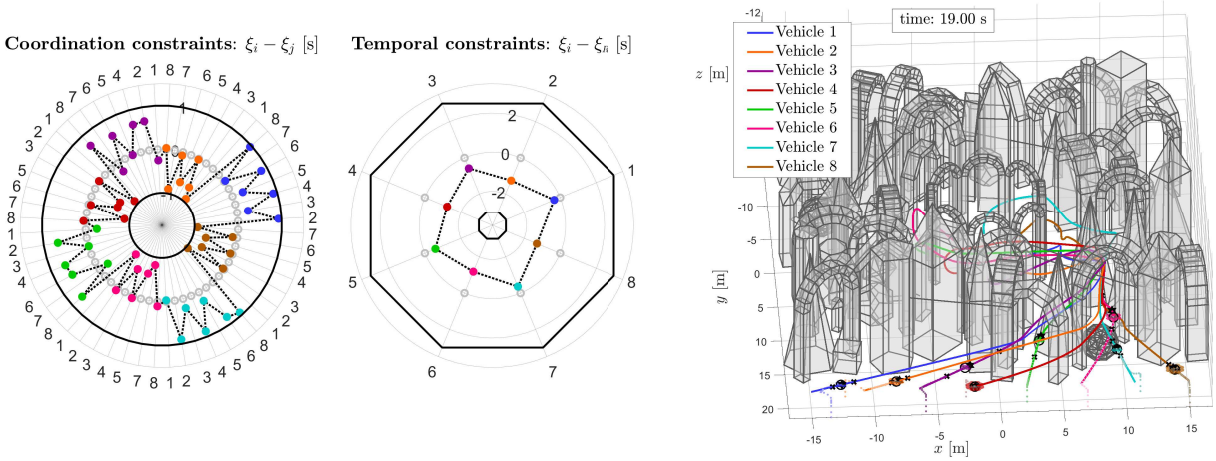


(c) Strict temporal constraints

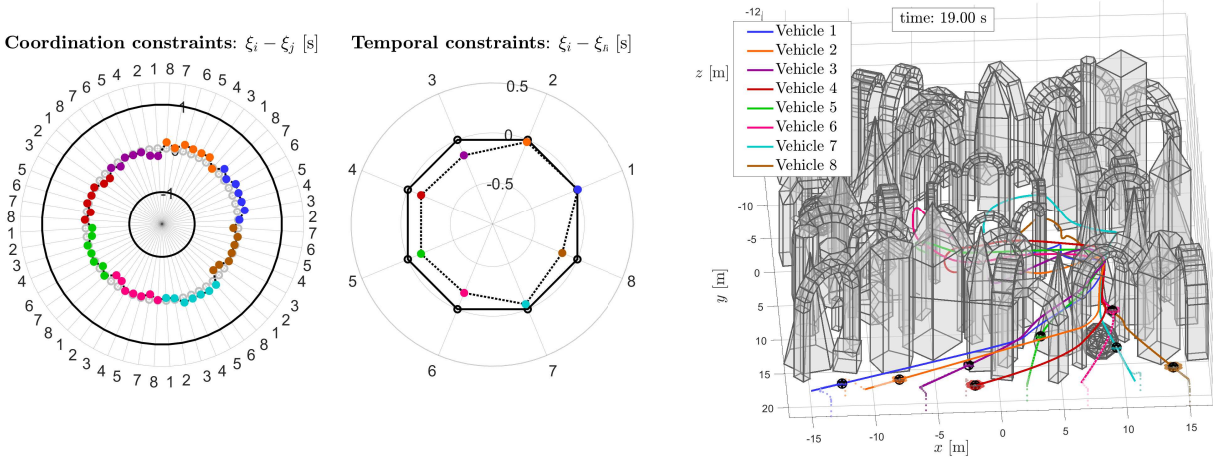
Figure 7.5: Simulation results 1 s after the time-critical coordination algorithm is engaged. The initial position errors lead to an increase in the coordination errors.



(a) Unenforced temporal constraints

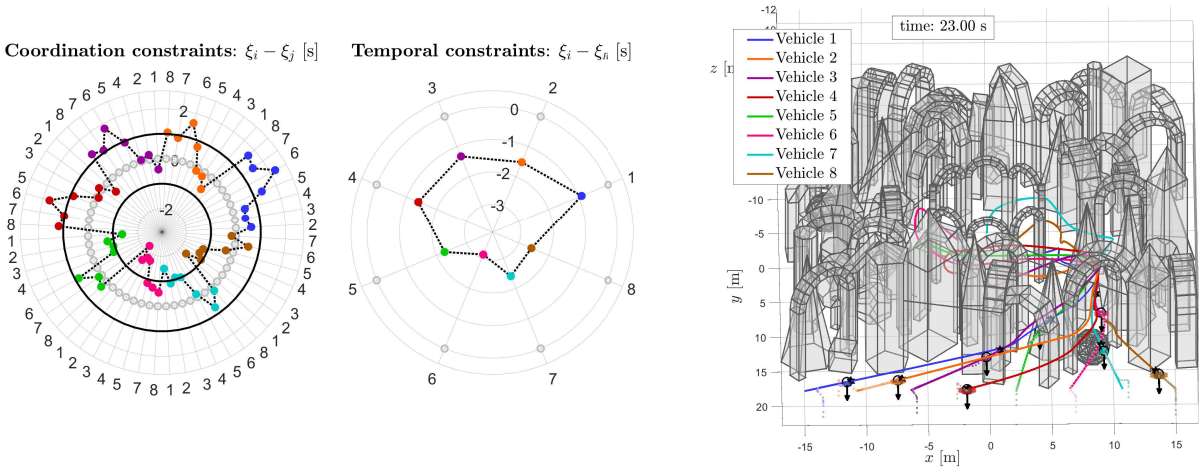


(b) Relaxed temporal constraints

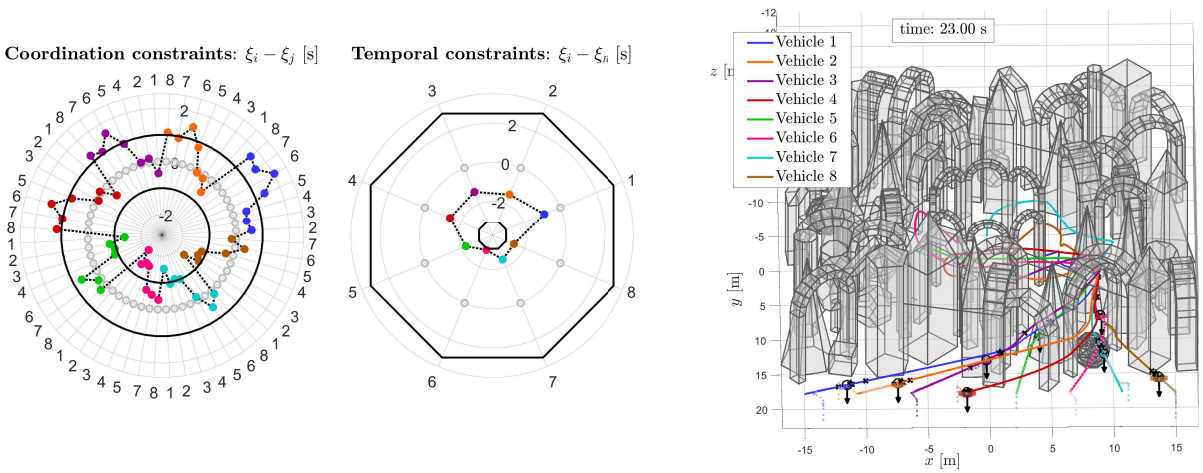


(c) Strict temporal constraints

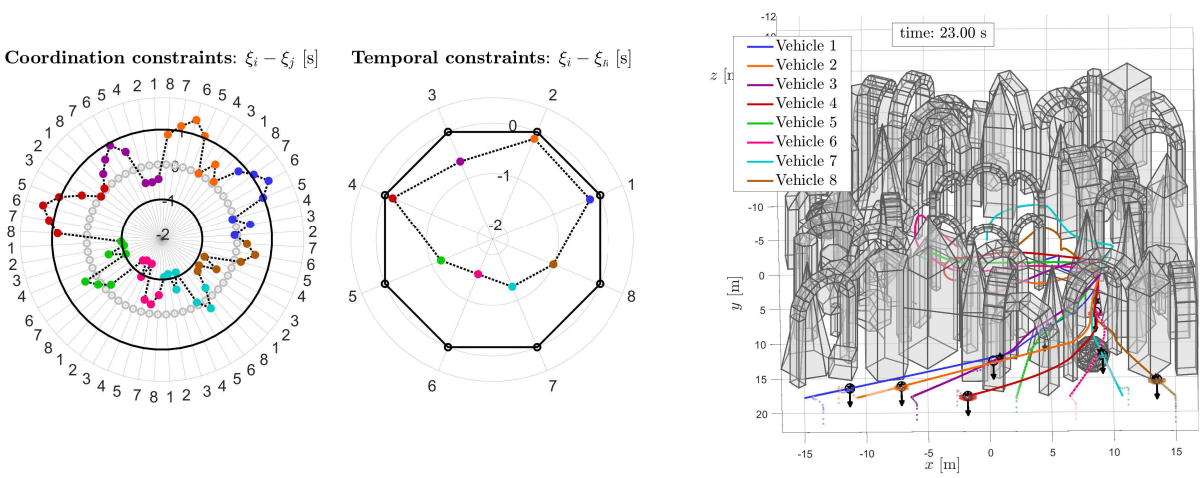
Figure 7.6: Simulation results 1 s before the wind gust is engaged. Vehicles adhere reasonably to their coordination and temporal constraints as expected.



(a) Unenforced temporal constraints

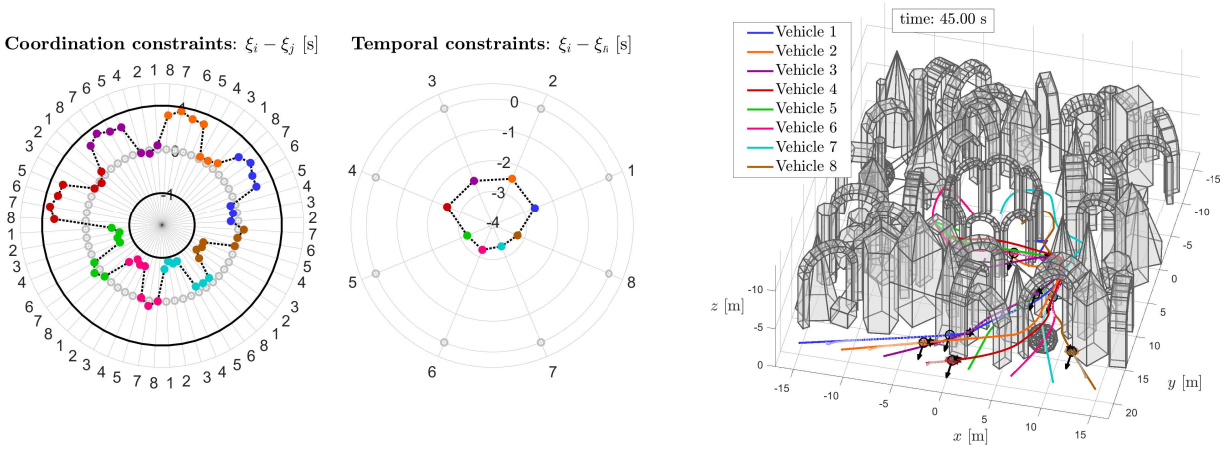


(b) Relaxed temporal constraints

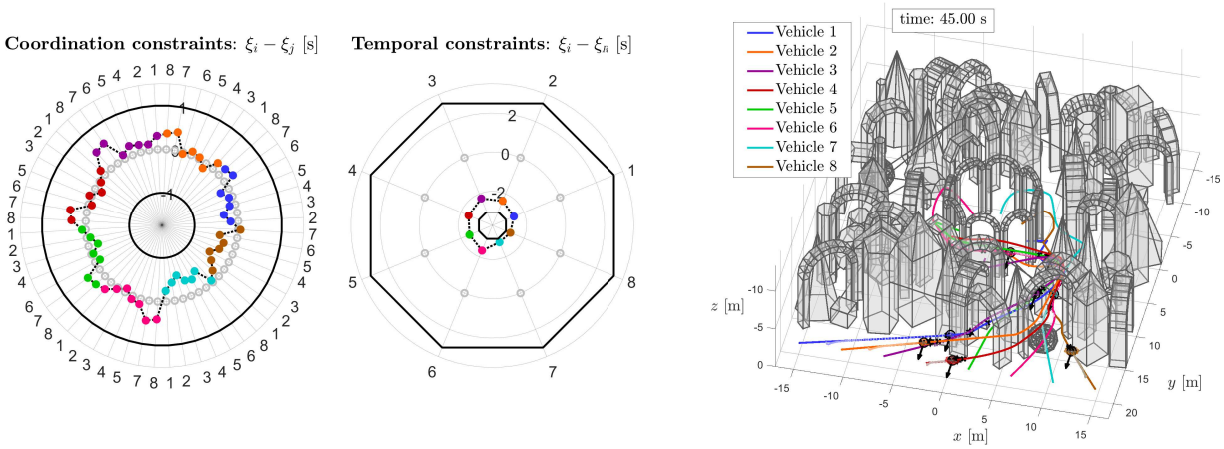


(c) Strict temporal constraints

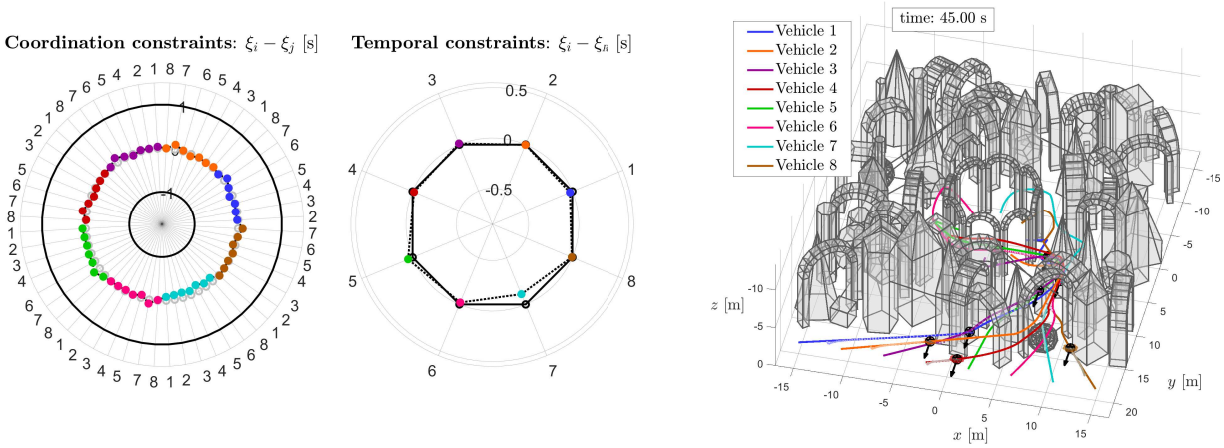
Figure 7.7: Simulation results 3 s after the wind gust is engaged. The wind leads to an increase in the coordination errors, and delays the fleet with respect to the planned schedule.



(a) Unenforced temporal constraints

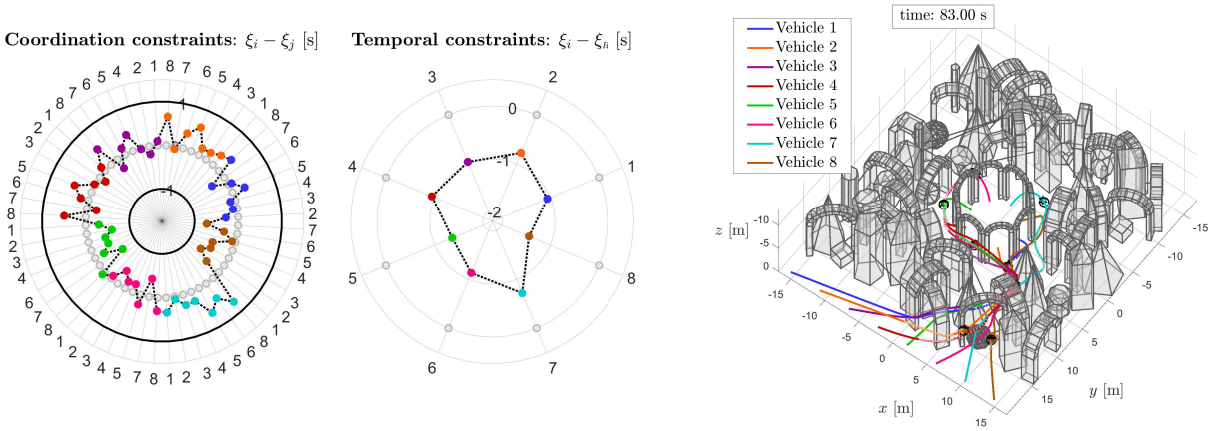


(b) Relaxed temporal constraints

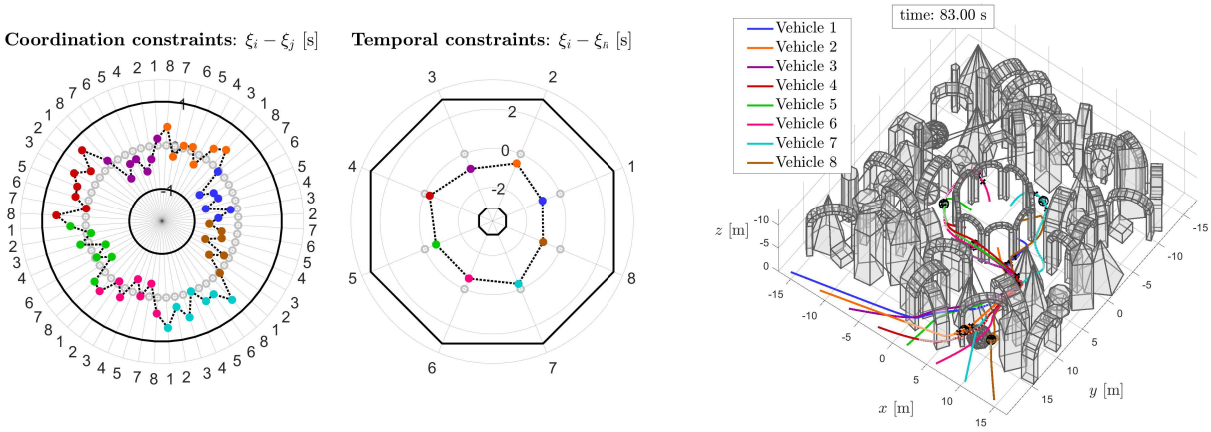


(c) Strict temporal constraints

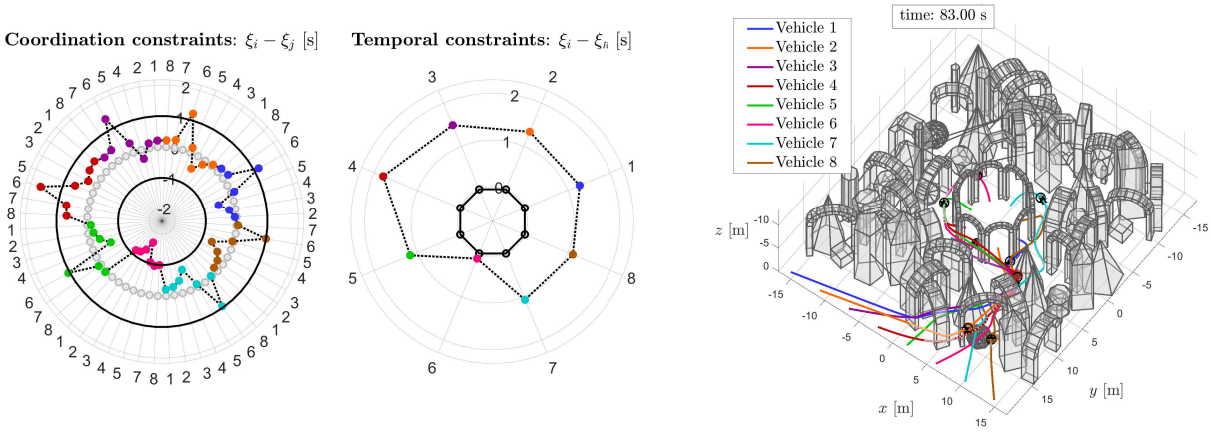
Figure 7.8: Simulation results 25 s after the wind gust is engaged. Vehicles adhere reasonably to their coordination and temporal constraints considering the challenge posed by the wind.



(a) Unenforced temporal constraints

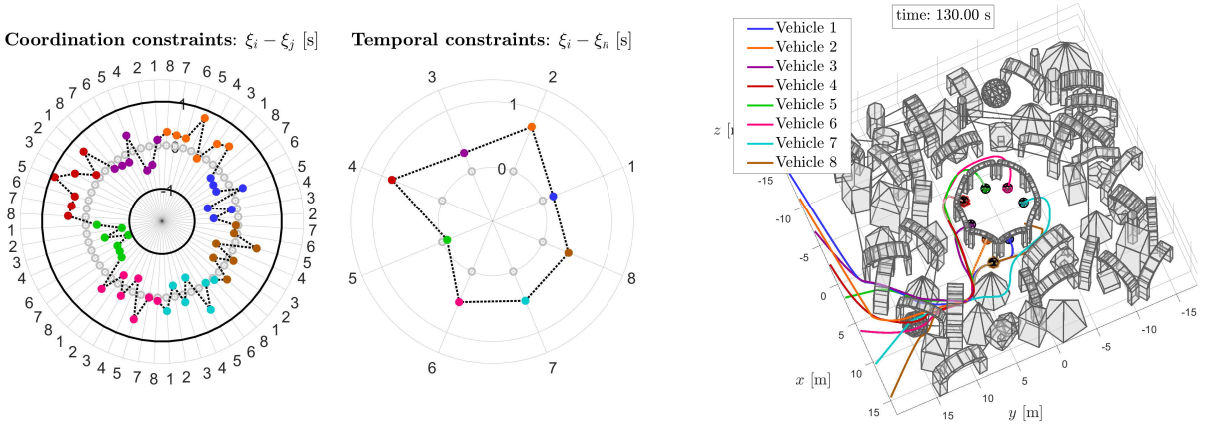


(b) Relaxed temporal constraints

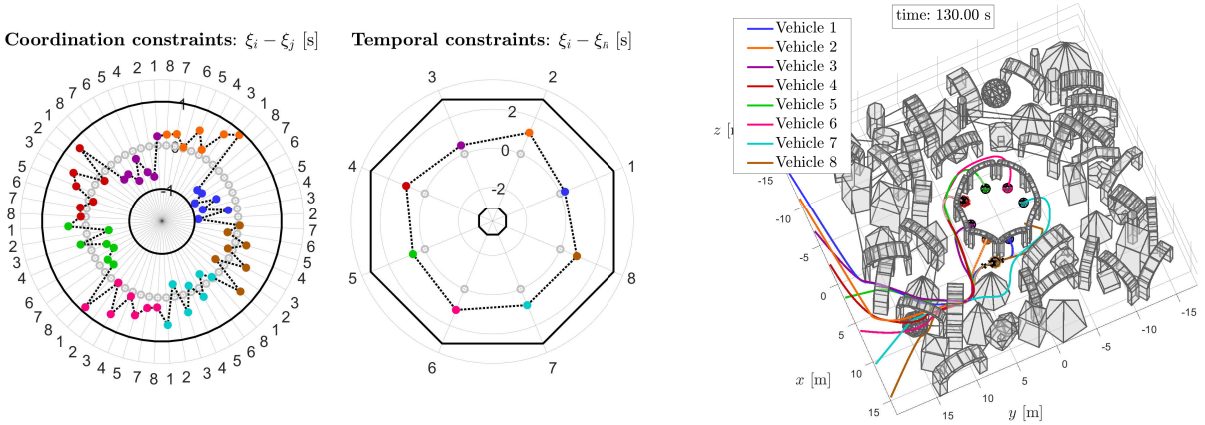


(c) Strict temporal constraints

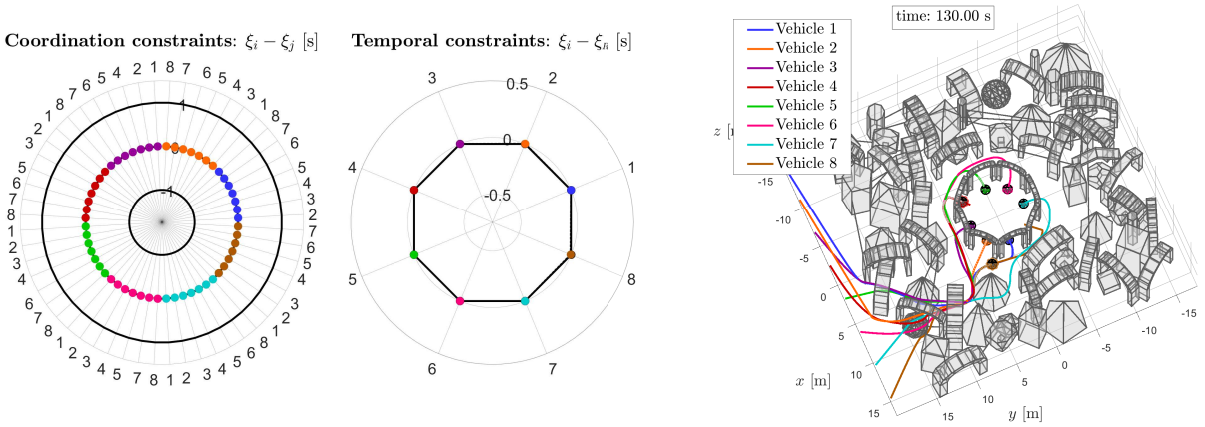
Figure 7.9: Simulation results 3 s after the wind gust is disengaged. The vehicles temporarily overshoot their virtual targets, which leads to an increase in the coordination errors.



(a) Unenforced temporal constraints

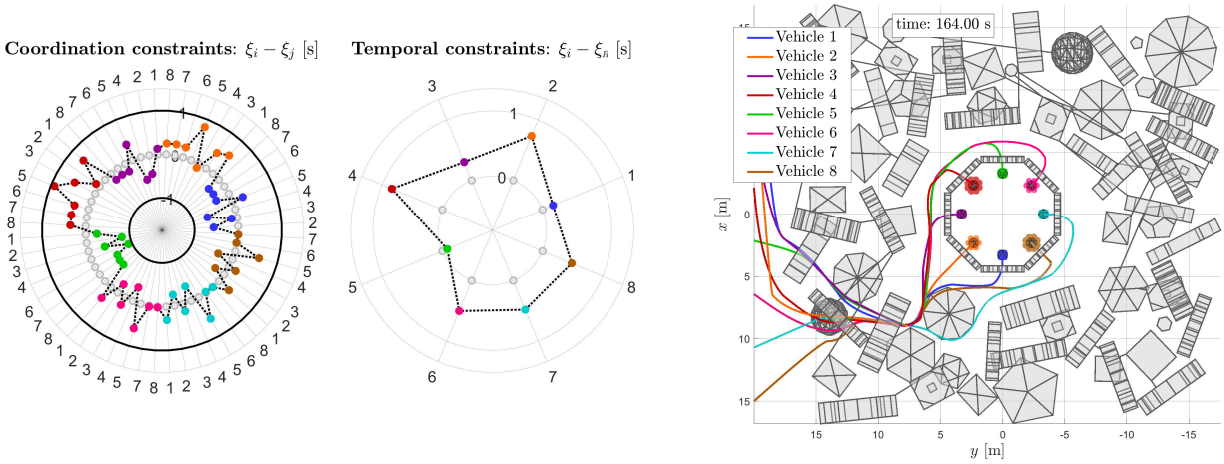


(b) Relaxed temporal constraints

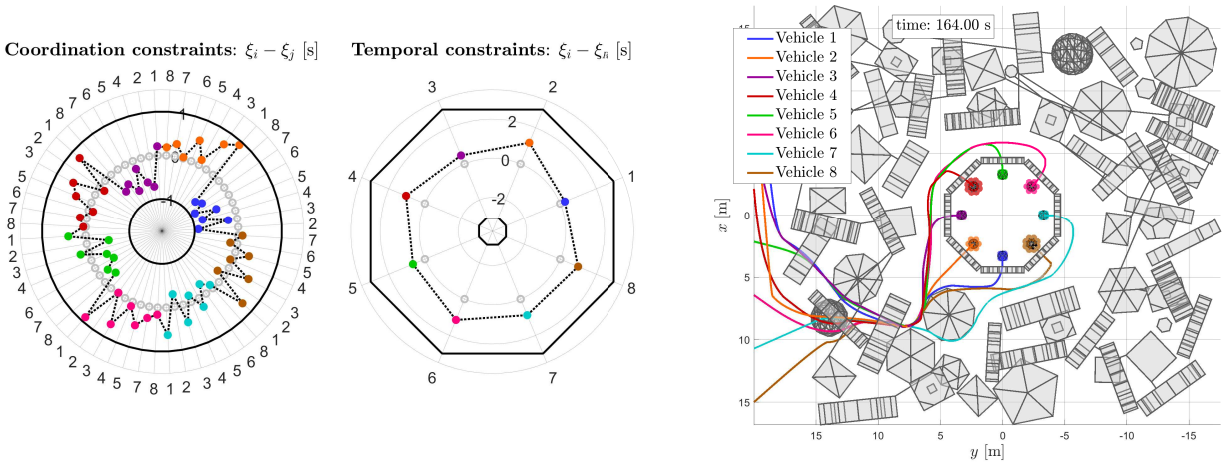


(c) Strict temporal constraints

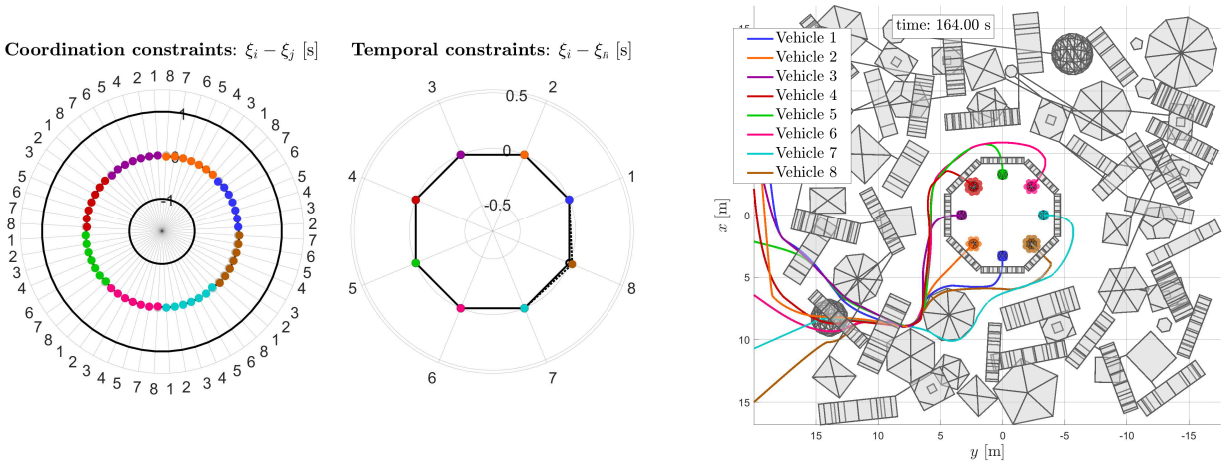
Figure 7.10: Simulation results 50 s after the wind gust is disengaged. Vehicles adhere to their coordination and temporal constraints as expected.



(a) Unenforced temporal constraints

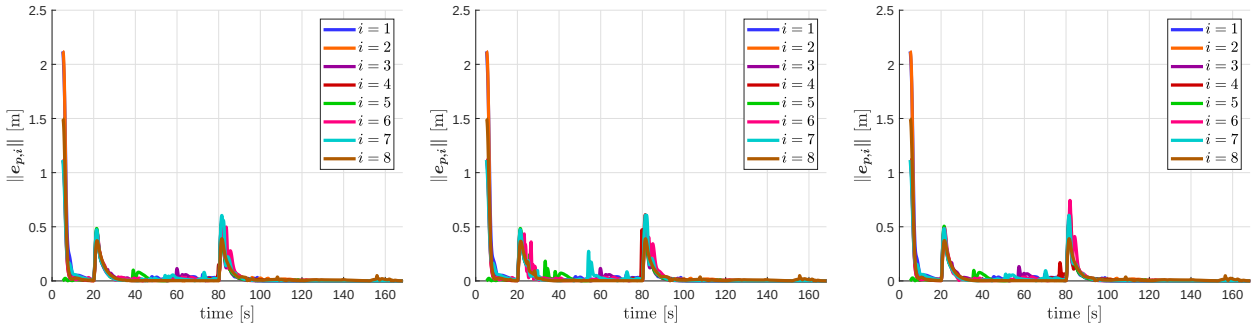


(b) Relaxed temporal constraints



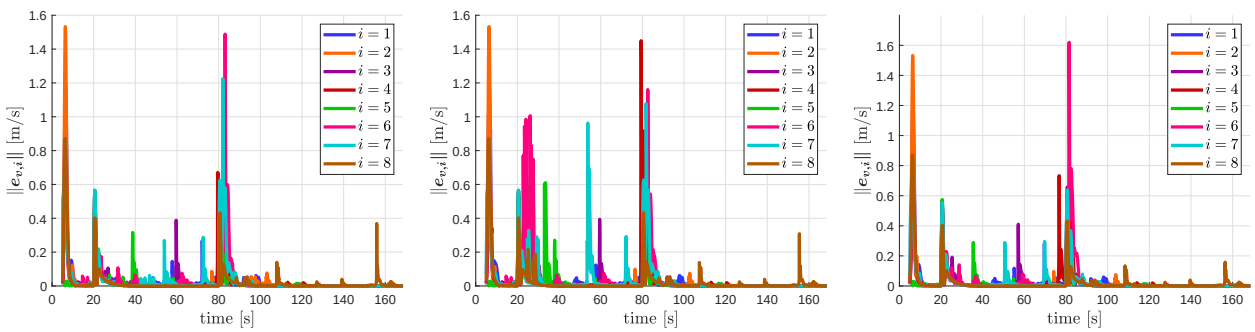
(c) Strict temporal constraints

Figure 7.11: Results at the conclusion of the simulation. Vehicles adhere to their coordination and temporal constraints as expected.



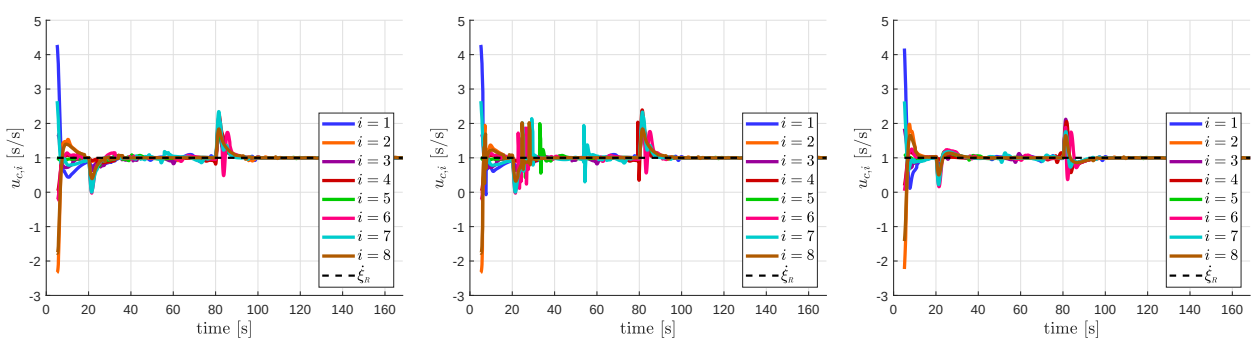
(a) Uneforced (b) Relaxed (c) Strict

Figure 7.12: Position-tracking errors.



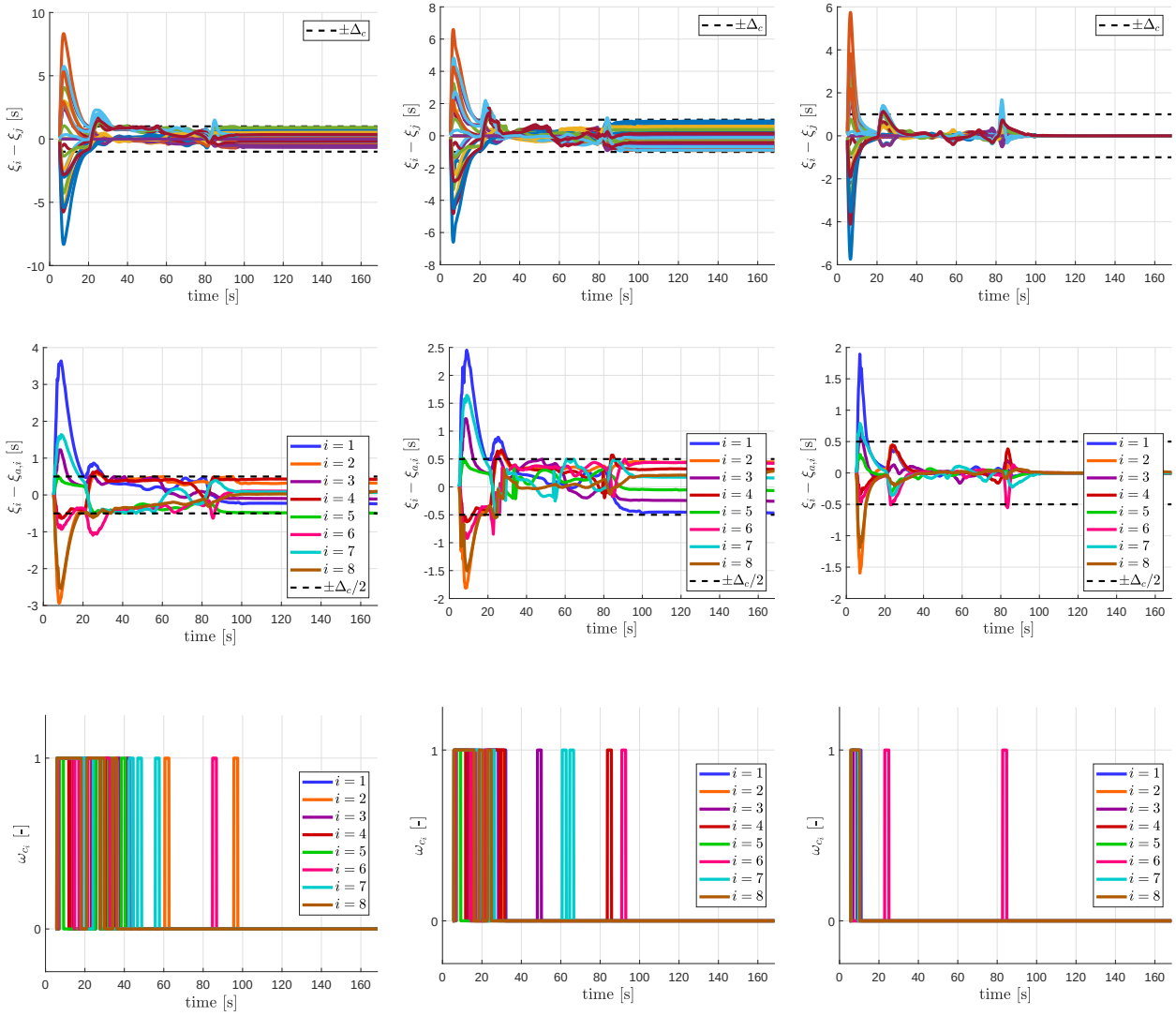
(a) Uneforced (b) Relaxed (c) Strict

Figure 7.13: Speed-tracking errors.



(a) Uneforced (b) Relaxed (c) Strict

Figure 7.14: Coordination control signal.

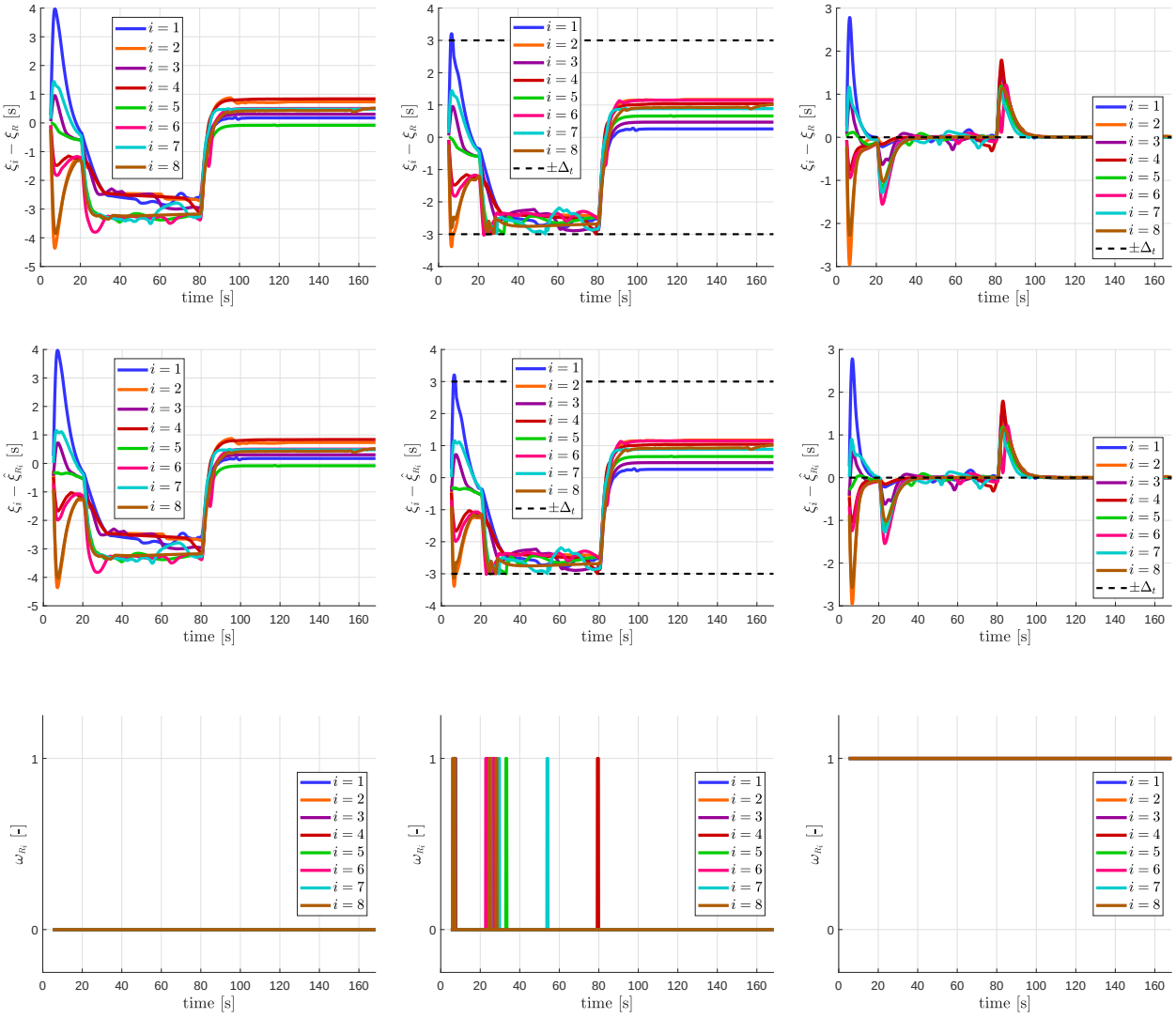


(a) Uneforced

(b) Relaxed

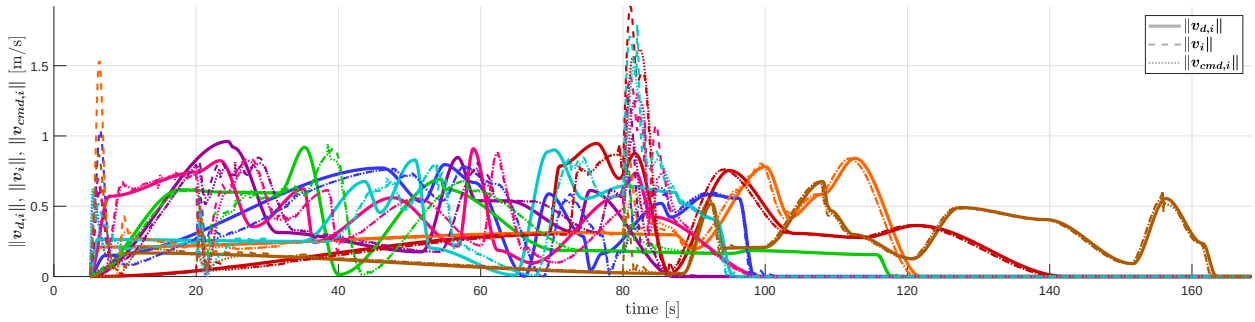
(c) Strict

Figure 7.15: Coordination errors (top), difference between the coordination state and the estimate of the fleet-average coordination state (middle), and coordination link weights (bottom).

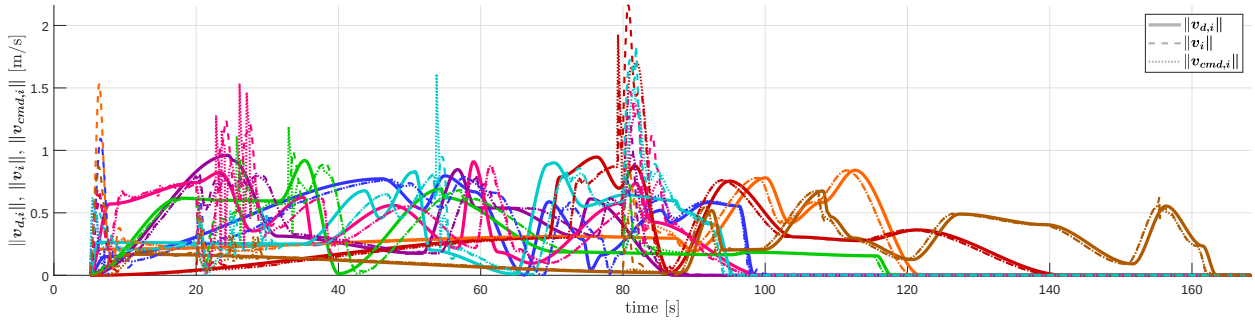


(a) Uneforced (b) Relaxed (c) Strict

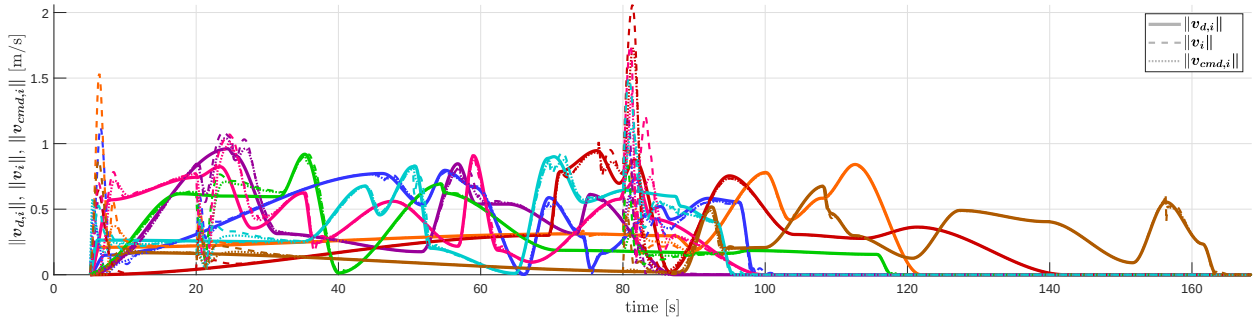
Figure 7.16: Temporal errors (top), estimated temporal error by each cooperating peer (middle), and reference link weights (bottom).



(a) Uneforced

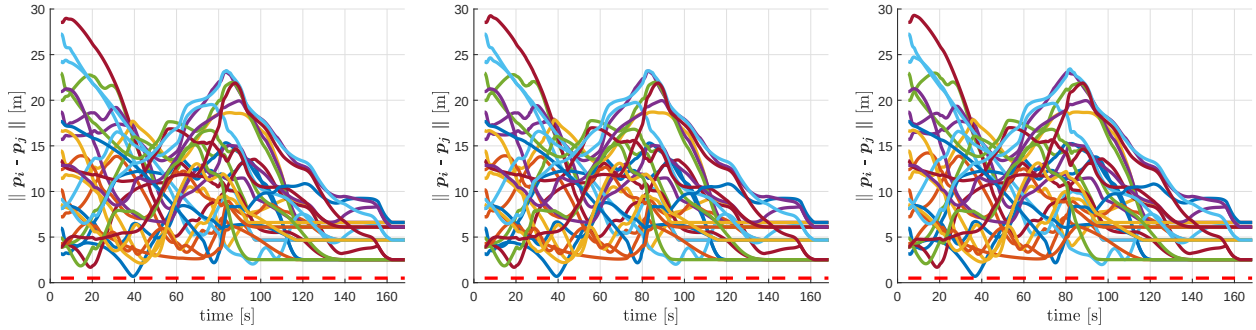


(b) Relaxed



(c) Strict

Figure 7.17: Speed profiles.



(a) Uneforced (b) Relaxed (c) Strict

Figure 7.18: Separation distance among vehicles.

Chapter 8

Conclusions and Future Work

This chapter closes the thesis with a summary of the contributions, the conclusions of this work, and future lines of research.

8.1 Conclusions

This thesis undertakes the problem of cooperative motion planning and control through cluttered environments. The objective is to steer a fleet of UASs through complex scenarios, while enforcing a variety of coordination and temporal constraints. The thesis is divided in two focus areas. The first part develops a centralized cooperative trajectory-generation framework for cluttered environments. It yields smooth piecewise parameterized trajectories that maintain safe separation with obstacles and cooperating peers. The second part develops a collection of distributed coordination protocols that capture a broader spectrum of coordination and temporal constraints than existing algorithms. These two parts are designed to integrate seamlessly with each other. Their consolidation into a unified architecture expands the range of cooperative missions that can be automated, and narrows the gap in versatility between human operators and cooperative motion control algorithms. Three conclusions follow from the work in cooperative trajectory generation for cluttered environments:

i) A geometric model that explicitly considers the uncertainty in obstacles and vehicle motion is proposed. Obstacles are represented by a polytope and an uncertainty distance. Similarly, vehicle motions are represented by piecewise Bézier curves and an uncertainty distance. These uncertainty distances capture the expected errors in the geometric description and location of the polytopes, as well as the vehicle dimensions and expected path-following errors. If a method to characterize these uncertainties is available, this yields a systematic approach for motion planning in uncertain scenarios.

ii) Novel geometric queries are proposed to aid motion-planning algorithms in cluttered and uncertain scenarios. The silhouette can be a useful geometric query to increase the probability of finding a path through a narrow passage for sample-based motion-planning algorithms. In addition, tolerance verification queries are an efficient mechanism to deconflict trajectories. They avoid the expensive step of explicitely computing the uncertainty buffers around the objects in the geometric model. To this effect, this thesis proposes tolerance verification queries for a polytope and a Bézier curve, and a pair of Bézier curves.

iii) A silhouette-informed sample-based motion-planning method is put forward in this thesis. The algorithm balances the narrow passage search—provided by the silhouette—with the random exploration of the configuration space. It successfully informs the generation of a tree through the narrow corridors in a cluttered urban-like environment. The output is processed by an edge-reduction technique posed as a linear programming problem, and a CNC-inspired smoothing algorithm that uses PH Bézier curves. The result is a piecewise $\mathcal{G}^2$ continuous path for each of the cooperating peers. Then, a centralized cooperative speed-assignment algorithm designs the speed profiles for all cooperating agents, and enforces temporal separation among vehicle trajectories. The resulting trajectories are piecewise $\mathcal{C}^2$ continuous curves that meet desired boundary conditions, safe separation, and dynamic constraints. The algorithm differs from existing approaches in the use of silhouette information, the tolerance verification queries developed explicitly for this application, and a different formulation of the curve reparameterization problem that decouples the design of the spatial and temporal components of the trajectory.

On the other hand, these conclusions follow from the time-critical coordination work:

i) This thesis defines six standard time-critical coordination strategies that result naturally from the combination of two types of coordination constraints—tight and loose—and three types of temporal constraints—unenforced, relaxed, and strict. Prior to this work, coordinated path-following algorithms were available for two of these six strategies—tight coordination with unenforced and strict temporal constraints.

ii) Distributed protocols are proposed for the six time-critical coordination strategies. The tight

coordination protocols require the vehicles to exchange a single variable with their peers over the network, whereas loose coordination protocols require the exchange of two variables. While this increases the network traffic, it also presents several structural advantages.

iii) Algebraic graph, Lyapunov, switched systems, and finite state machine theory are combined to prove that the tight coordination protocols solve the coordination problem under ideal virtual-target-tracking conditions. Transient and steady state performance bounds are derived as a function of the QoS of the communication network. These results are extended using perturbation theory for realistic virtual-target-tracking conditions. The resulting theorems indicate that the coordination and temporal errors converge to a neighborhood of the control objective. All the proofs are constructive, and provide a range for the tuning parameters that assures the system is stable. Both the transient performance bounds and range of suitable tuning parameters are conservative due to the complexity of the system dynamics and the concatenation of different Lyapunov tools, particularly for relaxed temporal constraints.

iv) To illustrate the efficacy of these algorithms, a simulated cooperative mission through a cluttered urban-like environment is presented for each of the time-critical coordination strategies. The results show that the protocols successfully impose the desired spectrum of coordination and temporal constraints, even in the presence of external disturbances such as wind gusts. To help interpret the simulation results, and aid human operators monitor adherence to the coordination and temporal constraints, this thesis proposes two spider plots.

v) Less stringent coordination and temporal constraints often lead to a smaller collective coordination control efforts. Thus, human operators should select the least stringent set of constraints that guarantee the successful completion of the mission.

Finally, two design principles were fundamental in breaking down a non-linear networked problem with switched dynamics into multiple tractable problems. First, a cascaded structure that relies on the inner-loop controllers to stabilize the vehicle dynamics is used to track the commands produced at the kinematic level by the cooperative path-following algorithm. Second, decoupling space and time both within the coordinated path-following and motion-planning algorithms facilitates the design of more complex behaviors, and the derivation of performance bounds.

8.2 Future Work

The work in this thesis could be expanded if several fundamental results were available. First, the design of continuous proximity queries between polyhedral objects and parametric curves with associated uncertainties can be used to include dynamic obstacles in the trajectory generation phase, as long as these obstacles can be parameterized as a function of the mission time. Second, a passivity condition for directed networked systems that does not assume the underlying graph is connected at some point during the mission would serve to relax the network assumptions within this thesis.

The cooperative trajectory-generation framework presented in this thesis includes a simplified dynamic model, and ignores communication constraints among vehicles. Future work should consider the thrust limits of the rotors within a UAS to accurately determine if a trajectory satisfies its dynamic constraints. Communication constraints such as transceiver range, line-of-sight communications, and obstacle obstructions must be taken into account to ensure vehicles do not lose contact with their cooperating peers for prolonged periods of time. In addition, decentralization of the proposed motion-planning algorithms can be used to leverage the computational power of all the agents participating in the cooperative effort.

In complex time-critical scenarios, the definition of feasible boundary constraints by human operators, such as the time of arrival, may pose a challenging problem. Automated reachability analyses can provide human operators with a reasonable expectation of the workload and demands they can request of the cooperating fleet. In line with these aids to mission management, further integration between motion planning and time-critical coordination can be helpful in a real scenario. To this effect, an analysis tool that consumes the assigned trajectories and identifies soft and hard bounds for the coordination and temporal windows can assist mission management and safety monitoring. Contrary to the work in this thesis, time-varying coordination and temporal windows would be designed automatically. The soft bounds would specify the coordination control objective; whereas hard bounds would trigger a trajectory recomputation, so the fleet can adapt to off-nominal mission states.

Regarding the time-critical coordination protocols, this thesis does not present stability analyses for loose coordination. The development of proofs that provide transient and steady-state

performance bounds for these algorithms would improve the trustworthiness of the loose coordination protocols. This thesis identifies that the switching logic for the the link weights in loose coordination and relaxed temporal constraints leads to smaller bounds on the allowable position error and velocity-tracking precision. These bounds ensure the velocity command remains within the flight envelope. Other switching strategies can be explored to mitigate this. Moreover, some of the structural advantages identified in the use of a distributed reference state estimator in loose coordination can be used to redesign the tight coordination protocols. Future research should also consider communication delays, malicious attacks on the communication network, channel noise, and directed communication links. Finally, the derivation of agent-specific time-critical coordination protocols is still an unresolved problem. However, it holds potential to further extend the flexibility of cooperative motion control algorithms, and expand the class of missions that can be automated with the proposed framework.

Appendix A

Algorithms

Algorithm 1: Tolerance verification algorithm for a polytope and a Bézier curve.

```
function toleranceVerificationPolytopeBezier(𝒪, p_d, d_s);
Input  : 𝒪 polytope
         p_d(ζ) Bézier curve of degree n with control points p̄_{d_i} and i = 0, ..., n
         d_s safe separation distance
Output : v_t tolerance verification result
             v_t = true,   if d̂(𝒪, p_d) > d_s + ε̄
             v_t = false,  otherwise
𝒫 ← controlPolytope(p_d) ;                ▷ 𝒱-representation with no boundary information
v ← GJKTolerance(𝒪, 𝒫, d_s);              ▷ Tolerance verification between 𝒪 and 𝒫
if v then
    v_t ← true;                           ▷ 𝒪 and 𝒫 are at least d_s apart ⟹ d(𝒪, p_d) > d_s
else
    v ← GJKTolerance(𝒪, p̄_{d_0}, d_s);    ▷ Tolerance verification between 𝒪 and p̄_{d_0}
    if v then
        v ← GJKTolerance(𝒪, p̄_{d_n}, d_s);  ▷ Tolerance verification between 𝒪 and p̄_{d_n}
        if v then
            (q_d, r_d) ← deCasteljau(p_d, 0.5);   ▷ Curve subdivision at ζ = 0.5
            v_t ← toleranceVerificationBezierPolytope(𝒪, q_d, d_s);   ▷ Recursive call
            if v_t then
                v_t ← toleranceVerificationBezierPolytope(𝒪, r_d, d_s);   ▷ Recursive call
            end
        else
            v_t ← false;                  ▷ Cannot guarantee 𝒪 and p̄_{d_n} are d_s apart
        end
    else
        v_t ← false;                      ▷ Cannot guarantee 𝒪 and p̄_{d_0} are d_s apart
    end
end
return v_t
```

Algorithm 2: Tolerance verification algorithm for a pair of Bézier curves.

```
function toleranceVerificationBezierBezier(p^1_d, p^2_d, d_s);
Input  : p^1_d(ζ) Bézier curve of degree n_1 with control points p̄^1_{d_i} and i = 0, ..., n_1
         p^2_d(ζ) Bézier curve of degree n_2 with control points p̄^2_{d_i} and i = 0, ..., n_2
         d_s safe separation distance
Output : v_t tolerance verification result
             v_t = true,   if d̂(p^1_d, p^2_d) > d_s + ε̄
             v_t = false,  otherwise
P_1 ← controlPolytope(p^1_d);          ▷ V-representation with no boundary information
P_2 ← controlPolytope(p^2_d);
v ← GJKTolerance(P_1, P_2, d_s);       ▷ Tolerance verification between P_1 and P_2
if v then
    v_t ← true;                        ▷ P_1 and P_2 are at least d_s apart ⟹ d(p^1_d, p^2_d) > d_s
else
    if min_{i∈{0,n_1}, j∈{0,n_2}}(‖p̄^1_{d_i} − p̄^2_{d_j}‖) > d_s + ε̄ then
        (q^1_d, r^1_d) ← deCasteljau(p^1_d, 0.5);   ▷ Curve subdivision at ζ = 0.5
        (q^2_d, r^2_d) ← deCasteljau(p^2_d, 0.5);
        v_t ← toleranceVerificationBezierBezier(q^1_d, q^2_d, d_s);   ▷ Recursive call
        if v_t then
            v_t ← toleranceVerificationBezierBezier(q^1_d, r^2_d, d_s);   ▷ Recursive call
            if v_t then
                v_t ← toleranceVerificationBezierBezier(r^1_d, q^2_d, d_s);   ▷ Recursive call
                if v_t then
                    v_t ← toleranceVerificationBezierBezier(r^1_d, r^2_d, d_s);   ▷ Recursive call
                end
            end
        end
    else
        v_t ← false;                   ▷ Cannot guarantee p̄^1_{d_{i*}} and p̄^1_{d_{j*}} are d_s apart
    end
end
return v_t
```

Algorithm 3: Silhouette of a polytope with an incomplete boundary representation.

function silhouetteIncompleteBoundary($\mathcal{O}, \boldsymbol{p_v}, \epsilon_v$);

Input : $\mathcal{O}$ polytope with incomplete boundary representation
$\boldsymbol{p_v}$ point of view
ϵ_v visibility threshold

Output : sil closed sequence of vertices that define the silhouette

$\mathcal{P} \leftarrow$ incompleteBoundaryPolytope($\mathcal{O}$) ; ▷ polytope with vertices on the boundary of $\mathcal{O}$

$(\mathcal{P}, f_k) \leftarrow$ reachVisibleBoundary($\mathcal{O}, \mathcal{P}, \boldsymbol{p_v}$) ; ▷ expand $\mathcal{P}$ until a facet $f_k \subset \mathcal{P}$ on the boundary of $\mathcal{O}$ is found

$(\mathcal{P}, f_v, f_o) \leftarrow$ reachSilhouette($\mathcal{O}, \mathcal{P}, f_k, \boldsymbol{p_v}$) ; ▷ expand $\mathcal{P}$ until two incident facets on the boundary of $\mathcal{O}$ are found, $f_v \subset \mathcal{P}$ is visible and $f_o \subset \mathcal{P}$ is not visible

$sil \leftarrow$ closeSilhouette($\mathcal{O}, \mathcal{P}, f_v, f_o, \boldsymbol{p_v}$) ; ▷ expand $\mathcal{P}$ and retrieve silhouette information until it defines a closed sequence of edges

return sil

Algorithm 4: Silhouette of a polytope with no boundary representation.

function silhouetteNoBoundary($\mathcal{O}, \boldsymbol{p_v}, \epsilon_v$);

Input : $\mathcal{O}$ polytope with no boundary representation
$\boldsymbol{p_v}$ point of view
ϵ_v visibility threshold

Output : sil closed sequence of vertices that define the silhouette

$\mathcal{P} \leftarrow$ nonDegenerateTetrahedon($\mathcal{O}$) ; ▷ tetrahedron with vertices on the boundary of $\mathcal{O}$

$(\mathcal{P}, f_k) \leftarrow$ reachVisibleBoundary($\mathcal{O}, \mathcal{P}, \boldsymbol{p_v}$) ; ▷ expand $\mathcal{P}$ until a facet $f_k \subset \mathcal{P}$ on the boundary of $\mathcal{O}$ is found

$(\mathcal{P}, f_v, f_o) \leftarrow$ reachSilhouette($\mathcal{O}, \mathcal{P}, f_k, \boldsymbol{p_v}$) ; ▷ expand $\mathcal{P}$ until two incident facets on the boundary of $\mathcal{O}$ are found, $f_v \subset \mathcal{P}$ is visible and $f_o \subset \mathcal{P}$ is not visible

$sil \leftarrow$ closeSilhouette($\mathcal{O}, \mathcal{P}, f_v, f_o, \boldsymbol{p_v}$) ; ▷ expand $\mathcal{P}$ and retrieve silhouette information until it defines a closed sequence of edges

return sil

Algorithm 5: Silhouette Informed Trees.

```
function silhouetteInformedTree(p_init, p_end, n_max, n_sil, d_s, d_e, X_obs);
Input  : p_init coordinates of the initial point
         p_end coordinates of the goal point
         n_max total number of samples, computational budget
         n_sil number of sampling attempts per silhouette-sampling batch
         d_s safe separation distance
         d_e expanding distance
         X_obs set of all obstacles
Output : T tree
V ← {p_init} ; E ← ∅; T = (V, E);                                   ▷ Initialize tree
s_m ← false;                                                        ▷ Initialize sampling-mode flag
while number of samples < n_max do
    // Sampling logic
    if s_m then
        sil ← silhouetteWire(p_nearest, id_c, d_s, d_e) ;           ▷ Compute silhouette
        p_rand ← sampleSilhouette(sil, n_sil) ;                     ▷ Sampling batch from the extended silhouette
    else
        p_rand ← sample(p_end) ;                                    ▷ Sample from the configuration space
    end
    p_nearest ← nearest(T, p_rand);                                 ▷ Find vertex on the tree that is closest to p_rand
    p_new ← steer(p_nearest, p_rand);                               ▷ Candidate to new vertex
    // Tolerance verification
    (v_t, id_c) ← toleranceVerification(p_nearest, p_new, X_obs);   ▷ Check safe distance separation
    if v_t then
        // Connections along a minimum-cost path
        r ← radius(V);                                              ▷ Compute maximum radius to consider other edge connections
        P_near ← near(T, p_new, r) ;                                ▷ Compute set of near vertices
        V ← V ∪ p_new ;                                             ▷ Add vertex
        p_min ← p_nearest; c_min ← cost(p_nearest) + costLine(p_nearest, p_new);
        foreach p_near ∈ P_near do
            (v_tn, ∼) ← toleranceVerification(p_near, p_new, X_obs);  ▷ Check safe distance separation
            c_n ← cost(p_near) + costLine(p_near, p_new);           ▷ Compute cost
            if v_tn ∧ c_n < c_min then
                p_min ← p_near; c_min ← c_n ;                       ▷ Update minimum cost and corresponding vertex
            end
            E ← E ∪ {p_min, p_new} ;                                ▷ Add edge with minimum cost
        end
        // Tree rewiring
        foreach p_near ∈ P_near do
            (v_tn, ∼) ← toleranceVerification(p_new, p_near, X_obs);  ▷ Check safe distance separation
            c_n ← cost(p_new) + costLine(p_new, p_near);            ▷ Compute cost
            if v_tn ∧ c_n < cost(p_near) then
                p_parent ← parent(p_near) ;                         ▷ Find parent node
            end
            E ← (E \ {p_parent, p_near}) ∪ {p_new, p_near} ;        ▷ Remove initial edge, and add new edge
        end
        s_m ← false
    else
        s_m ← ¬ s_m
    end
end
```

Appendix B

Coordination Dynamics

B.1 Tight Coordination

First, express the coordination control law in (5.3) in terms of the collective temporal, coordination, and rate errors. From the definition of $\boldsymbol{\zeta_c}(t)$ and $\boldsymbol{\zeta_r}(t)$ in (5.6), and $\mathbf{L}\,(t)\,\boldsymbol{\Pi} = \mathbf{L}\,(t)$ it follows that

$$\begin{aligned}\boldsymbol{u_c}(t) &= -k_P\mathbf{L}\,(t)\,\boldsymbol{\Pi}\boldsymbol{\xi}(t) - k_R\boldsymbol{\Omega}(t)\,(\boldsymbol{\xi}(t) - \xi_R(t)\mathbf{1_n}) + \begin{bmatrix}\mathbf{0}\\ \boldsymbol{\chi}(t)-\dot{\xi}_R\mathbf{1_{n-n_\ell}}\end{bmatrix} + \dot{\xi}_R\mathbf{1_n}\\ &= -k_P\mathbf{L}\,(t)\,\mathbf{Q}^\top\boldsymbol{\zeta_c}(t) - k_R\boldsymbol{\Omega}(t)\,(\boldsymbol{\xi}(t) - \xi_R(t)\mathbf{1_n}) + \begin{bmatrix}\mathbf{0}\\ \boldsymbol{\zeta_r}(t)\end{bmatrix} + \dot{\xi}_R\mathbf{1_n}.\end{aligned}$$

Consider the augmented coordination state $\bar{\boldsymbol{\xi}}(t)$, and matrix $\overline{\mathbf{Q}}$:

$$\bar{\boldsymbol{\xi}}(t) := \left[\xi_R(t)\,,\,\boldsymbol{\xi}^\top(t)\right]^\top, \qquad \overline{\mathbf{Q}} := \begin{bmatrix}\sqrt{\frac{n}{n+1}} & -\frac{1}{\sqrt{n(n+1)}}\mathbf{1_n^\top}\\ \mathbf{0} & \mathbf{Q}\end{bmatrix},$$

where $\overline{\mathbf{Q}}\,\overline{\mathbf{Q}}^\top = \mathbb{I}_\mathbf{n}$, with the augmented projection matrix being $\overline{\boldsymbol{\Pi}} := \overline{\mathbf{Q}}^\top\,\overline{\mathbf{Q}} = \mathbb{I}_{\mathbf{n+1}} - \frac{1}{n+1}\mathbf{1_{n+1}}\mathbf{1_{n+1}^\top}$. Expressing $\boldsymbol{\xi}(t) - \xi_R(t)\mathbf{1_n}$ in terms of the augmented coordination state as

$$\boldsymbol{\xi}(t) - \xi_R(t)\mathbf{1_n} = [\,-\mathbf{1_n}\;\;\mathbb{I}_\mathbf{n}\,]\,\bar{\boldsymbol{\xi}}(t),$$

noting that $[\,-\mathbf{1_n}\;\;\mathbb{I}_\mathbf{n}\,]\,\overline{\boldsymbol{\Pi}} = [\,-\mathbf{1_n}\;\;\mathbb{I}_\mathbf{n}\,]$, the definition of $\boldsymbol{\zeta_c}(t)$, and algebraic manipulation yields

$$\boldsymbol{u_c}(t) = -k_P\mathbf{L}\,(t)\,\mathbf{Q}^\top\boldsymbol{\zeta_c}(t) - k_R\boldsymbol{\Omega}(t)\left(-\zeta_t(t)\mathbf{1_n} + \mathbf{Q}^\top\boldsymbol{\zeta_c}(t)\right) + \begin{bmatrix}\mathbf{0}\\ \boldsymbol{\zeta_r}(t)\end{bmatrix} + \dot{\xi}_R\mathbf{1_n}. \tag{B.1}$$

Now, the definition of $\zeta_t(t)$, property $\mathbf{1}_{\mathbf{n}}^{\top}\mathbf{L}\,(t) = \mathbf{0}$, which can be proven using Assumption 4, and Equation (B.1) imply that

$$\begin{aligned}\dot{\zeta}_t(t) &= \dot{\xi}_R - \frac{1}{n}\mathbf{1}_{\mathbf{n}}^{\top}\left(-k_R\boldsymbol{\Omega}(t)\left(-\zeta_t(t)\mathbf{1}_{\mathbf{n}} + \mathbf{Q}^{\top}\boldsymbol{\zeta_c}(t)\right) + \begin{bmatrix}\mathbf{0}\\ \boldsymbol{\zeta_r}(t)\end{bmatrix} + \dot{\xi}_R\mathbf{1}_{\mathbf{n}} + \boldsymbol{e_{\mathcal{T}}}(t)\right)\\ &= -\frac{k_R}{n}\tilde{\omega}(t)\zeta_t(t) + \frac{k_R}{n}\boldsymbol{\omega}^{\top}(t)\,\mathbf{Q}^{\top}\boldsymbol{\zeta_c}(t) - \frac{1}{n}\mathbf{1}_{\mathbf{n}-\mathbf{n}_\ell}^{\top}\boldsymbol{\zeta_r}(t) - \frac{1}{n}\mathbf{1}_{\mathbf{n}}^{\top}\boldsymbol{e_{\mathcal{T}}}(t),\end{aligned} \tag{B.2}$$

where the definitions for $\tilde{\omega}(t)$ and $\boldsymbol{\omega}(t)$ are given in Equation (5.8). The definition of $\boldsymbol{\zeta_c}(t)$, property $\mathbf{Q}\mathbf{1}_{\mathbf{n}} = \mathbf{0}$, and Equation (B.1) lead to the expression:

$$\begin{aligned}\dot{\boldsymbol{\zeta_c}}(t) &= k_R\mathbf{Q}\boldsymbol{\omega}(t)\zeta_t(t) - k_P\mathbf{Q}\mathbf{L}\,(t)\,\mathbf{Q}^{\top}\boldsymbol{\zeta_c}(t) - k_R\mathbf{Q}\boldsymbol{\Omega}(t)\mathbf{Q}^{\top}\boldsymbol{\zeta_c}(t) + \mathbf{Q}\mathbf{C_e}\boldsymbol{\zeta_r}(t) + \mathbf{Q}\boldsymbol{e_{\mathcal{T}}}(t)\\ &= k_R\mathbf{Q}\,\boldsymbol{\omega}(t)\zeta_t(t) - k_P\widehat{\mathbf{L}}\,(t)\,\boldsymbol{\zeta_c}(t) + \mathbf{Q}\mathbf{C_e}\boldsymbol{\zeta_r}(t) + \mathbf{Q}\boldsymbol{e_{\mathcal{T}}}(t).\end{aligned} \tag{B.3}$$

Similarly, from the definition of $\boldsymbol{\zeta_r}(t)$ it follows that

$$\dot{\boldsymbol{\zeta_r}}(t) = -k_I\mathbf{C_e}^{\top}\boldsymbol{\Pi}\,\mathbf{L}\,(t)\,\mathbf{Q}^{\top}\boldsymbol{\zeta_c}(t). \tag{B.4}$$

Equations (B.2), (B.3), and (B.4) rewritten in matrix form yield the collective error dynamics in Equation (5.7).

Appendix C

Proofs

C.1 Proof of Lemma 1

The position error dynamics in Equation (2.9) and the speed command in Equation (2.11) yield

$$\dot{\boldsymbol{e}}_{\boldsymbol{p},\boldsymbol{i}}(t) = -k_{PF,i}\boldsymbol{e}_{\boldsymbol{p},\boldsymbol{i}}(t) + \boldsymbol{e}_{\boldsymbol{v},\boldsymbol{i}}(t). \tag{C.1}$$

Consider the Lyapunov function candidate

$$V_p = \frac{1}{2}\boldsymbol{e}_{\boldsymbol{p},\boldsymbol{i}}^{\top}(t)\boldsymbol{e}_{\boldsymbol{p},\boldsymbol{i}}(t).$$

Then, the time derivative of V_p along the trajectories of (C.1) can be bounded as follows:

$$\begin{aligned}
\dot{V}_p(\boldsymbol{e}_{\boldsymbol{p},\boldsymbol{i}}) &= -k_{PF,i}\|\boldsymbol{e}_{\boldsymbol{p},\boldsymbol{i}}\|^2 + \boldsymbol{e}_{\boldsymbol{v},\boldsymbol{i}}^{\top}(t)\boldsymbol{e}_{\boldsymbol{p},\boldsymbol{i}}(t) \\
&\leq -2k_{PF,i}V_p(\boldsymbol{e}_{\boldsymbol{p},\boldsymbol{i}}) + \|\boldsymbol{e}_{\boldsymbol{v},\boldsymbol{i}}(t)\|\sqrt{2V_p(\boldsymbol{e}_{\boldsymbol{p},\boldsymbol{i}})} \\
&\leq -2k_{PF,i}V_p(\boldsymbol{e}_{\boldsymbol{p},\boldsymbol{i}}) + \bar{e}_{v,i}\sqrt{2V_p(\boldsymbol{e}_{\boldsymbol{p},\boldsymbol{i}})},
\end{aligned}$$

where $\bar{e}_{v,i}$ is defined in Assumption 1. Next, the change of variables

$$W_p(\boldsymbol{e}_{\boldsymbol{p},\boldsymbol{i}}) := \sqrt{V_p(\boldsymbol{e}_{\boldsymbol{p},\boldsymbol{i}})}, \tag{C.2}$$

and its time derivative $\dot{W}_p(\boldsymbol{e}_{\boldsymbol{p},\boldsymbol{i}}) = \frac{\dot{V}_p(\boldsymbol{e}_{\boldsymbol{p},\boldsymbol{i}})}{2\sqrt{V_p(\boldsymbol{e}_{\boldsymbol{p},\boldsymbol{i}})}}$ yield a linear differential inequality for all $V_p(\boldsymbol{e}_{\boldsymbol{p},\boldsymbol{i}}) \neq 0$

$$\dot{W}_p(\boldsymbol{e}_{\boldsymbol{p},\boldsymbol{i}}) \leq -k_{PF,i}W_p(\boldsymbol{e}_{\boldsymbol{p},\boldsymbol{i}}) + \frac{\bar{e}_{v,i}}{\sqrt{2}}. \tag{C.3}$$

On the other hand, when $V_p(\boldsymbol{e_{p,i}}) = 0$

$$\lim_{\|\boldsymbol{e_{p,i}}\|\to 0} \dot{W}_p(\boldsymbol{e_{p,i}}) = \lim_{\|\boldsymbol{e_{p,i}}\|\to 0} \frac{-k_{PF,i}\|\boldsymbol{e_{p,i}}\|^2 + \|\boldsymbol{e_{p,i}}\|\|\boldsymbol{e_{v,i}}\|\cos\theta}{\sqrt{2}\|\boldsymbol{e_{p,i}}\|} = \frac{\|\boldsymbol{e_{v,i}}\|\cos\theta}{\sqrt{2}} \leq \frac{\bar{e}_{v,i}}{\sqrt{2}},$$

where $\theta(t)$ is the angle between vectors $\boldsymbol{e_{p,i}}(t)$ and $\boldsymbol{e_{v,i}}(t)$ at time t. Hence, the linear inequality in Equation (C.3) can be used with the comparison lemma, Lemma 3.4 in [108], to conclude that

$$W_p(t) \leq W_p(0)e^{-k_{PF,i}t} + \frac{\bar{e}_{v,i}}{\sqrt{2}k_{PF,i}}\left(1 - e^{-k_{PF,i}\,t}\right),$$

for both cases, $V_p(\boldsymbol{e_{p,i}}) \neq 0$ and $V_p(\boldsymbol{e_{p,i}}) = 0$. Finally, the change of variables in (C.2) implies

$$\|\boldsymbol{e_{p,i}}(t)\| \leq \|\boldsymbol{e_{p,i}}(0)\|e^{-k_{PF,i}\,t} + \frac{\bar{e}_{v,i}}{k_{PF,i}}\left(1 - e^{-k_{PF,i}\,t}\right).$$

□

C.2 Proof of Theorem 1

The following change of variables, proposed in [4], removes the Laplacian in $\boldsymbol{A_u}(t)$ from the off-diagonal element:

$$\boldsymbol{z_u}(t) = \boldsymbol{S_u}\boldsymbol{\zeta_u}(t), \qquad \boldsymbol{S_u} = \begin{bmatrix} \mathbb{I}_{\mathbf{n-1}} & \mathbf{0} \\ -\frac{k_I}{k_P}\mathbf{C_e}^\top\mathbf{Q} & \mathbb{I}_{\mathbf{n-n_\ell}} \end{bmatrix}. \tag{C.4}$$

Then, the dynamics in (5.9) and the change of variables above yield

$$\dot{\boldsymbol{z_u}}(t) = \boldsymbol{A_{z_u}}(t)\boldsymbol{z_u}(t), \qquad \boldsymbol{A_{z_u}}(t) := \boldsymbol{S_u}\boldsymbol{A_u}(t)\boldsymbol{S_u}^{-1}, \tag{C.5}$$

$$\boldsymbol{A_{z_u}}(t) = \begin{bmatrix} -k_P\bar{\boldsymbol{L}}(t) + \frac{k_I}{k_P}\boldsymbol{W_e} & \mathbf{Q}\mathbf{C_e} \\ -\left(\frac{k_I}{k_P}\right)^2\boldsymbol{\Pi_e}\mathbf{C_e}^\top\mathbf{Q}^\top & -\frac{k_I}{k_P}\boldsymbol{\Pi_e} \end{bmatrix},$$

where $\boldsymbol{\Pi_e} := \mathbf{C_e}^\top\boldsymbol{\Pi}\mathbf{C_e}$, and $\boldsymbol{W_e} := \mathbf{Q}\mathbf{C_e}\mathbf{C_e}^\top\mathbf{Q}^\top$. As suggested in [4], to aid in the analysis of the collective system, consider the dynamics

$$\dot{\boldsymbol{\phi}}(t) = -k_P\bar{\boldsymbol{L}}(t)\,\boldsymbol{\phi}(t), \qquad \boldsymbol{\phi}(t) \in \mathbb{R}^{n-1}. \tag{C.6}$$

Since $\bar{\boldsymbol{L}}(t)$ is symmetric, positive semidefinite, and satisfies Assumption 5, then similar to the proof of Lemma 5 in [109] or Lemma 3 in [110] one can show that

$$\|\boldsymbol{\phi}(t)\| \leq \kappa_\phi \|\boldsymbol{\phi}(0)\| e^{-\lambda_\phi t}, \quad \text{with} \quad \kappa_\phi := 1, \quad \text{and} \quad \lambda_\phi \geq \gamma_u := \frac{k_P n \mu}{(1+\Lambda_u^2 T)^2},$$

where $\Lambda_u^2 := k_P n$, which leads to the following lemma.

Lemma 4 *There exists a Lyapunov function $V(t,\boldsymbol{\phi})$ for the origin of the system dynamics in* (C.6) *of the form $V(t,\boldsymbol{\phi}) = \boldsymbol{\phi}^\top \boldsymbol{P}_{\mathbf{0_u}}(t)\boldsymbol{\phi}$, where $\boldsymbol{P}_{\mathbf{0_u}}(t)$ is a continuous, piecewise differentiable, symmetric matrix that satisfies*

$$\alpha_u \mathbb{I}_{\mathbf{n-1}} \leq \boldsymbol{P}_{\mathbf{0_u}} \leq \beta_u \mathbb{I}_{\mathbf{n-1}},$$

$$\dot{\boldsymbol{P}}_{\mathbf{0_u}} - k_P \left(\bar{\boldsymbol{L}} \boldsymbol{P}_{\mathbf{0_u}} + \boldsymbol{P}_{\mathbf{0_u}} \bar{\boldsymbol{L}}\right) \leq -\chi_u \mathbb{I}_{\mathbf{n-1}},$$

with $\alpha_u := \frac{\chi_u}{2\Lambda_u^2}$, $\beta_u := \frac{\delta_u}{2\gamma_u}$, and any $0 < \chi_u \leq \delta_u$.

Proof. This proof is similar to the proof of Theorem 4.12 in [108]. □

C.2.1 GUES of the Coordination and Rate Errors

Tools from Lyapunov stability, algebraic graph theory, and Lemma 4 help prove the following result.

Lemma 5 *Assume the information flow $\mathcal{G}(t)$ satisfies Assumptions 3 through 5. Then, there exists a Lyapunov function V_u for the system in* (C.5) *that satisfies*

$$a_u \|\boldsymbol{z_u}\|^2 \leq V_u \leq b_u \|\boldsymbol{z_u}\|^2, \tag{C.8}$$

$$\dot{V}_u \leq -2\lambda_u V_u, \tag{C.9}$$

with known control gains k_P and k_I, constants a_u and b_u, and rate of convergence $\lambda_u = \nu \underline{\lambda}_u$, for any $\nu \in (0,1]$, and a guaranteed rate of convergence

$$\underline{\lambda}_u := \frac{k_P n \mu}{(1+k_P n T)^2} \left(1 + \eta_{I,u} \frac{n}{n_\ell}\right)^{-1},$$

with design parameter $\eta_{I,u} \geq 2$.

Proof. Consider the Lyapunov function candidate[1]

$$V_u(t, \boldsymbol{z_u}) := \boldsymbol{z_u}^\top \boldsymbol{P_u}(t)\boldsymbol{z_u}, \qquad \boldsymbol{P_u} := \begin{bmatrix} \boldsymbol{P_{0_u}} & \mathbf{0} \\ \mathbf{0} & \left(\frac{k_P}{k_I}\right)^2 \beta_u \boldsymbol{\Pi_e}^{-1} \end{bmatrix}. \tag{C.10}$$

The time derivative of V_u along the trajectories of (C.5) is

$$\dot{V}_u := \frac{\partial V_u}{\partial t} = \boldsymbol{z_u}^\top \left(\dot{\boldsymbol{P}}_{\boldsymbol{u}} + \boldsymbol{A}_{\boldsymbol{z_u}}^\top \boldsymbol{P_u} + \boldsymbol{P_u} \boldsymbol{A}_{\boldsymbol{z_u}} \right) \boldsymbol{z_u}.$$

Then, proving $\dot{V}_u \leq -2\lambda_u V_u$ can be reduced to ensuring positive semi-definiteness of

$$-2\lambda_u \boldsymbol{P_u} - \dot{\boldsymbol{P}}_{\boldsymbol{u}} - \boldsymbol{A}_{\boldsymbol{z_u}}^\top \boldsymbol{P_u} - \boldsymbol{P_u} \boldsymbol{A}_{\boldsymbol{z_u}} \geq \boldsymbol{M_u},$$

where $\boldsymbol{M_u}$ can be derived noting that $\boldsymbol{\Pi_e}^{-1} \leq \frac{n}{n_\ell} \mathbb{I}_{\mathbf{n}-\mathbf{n}_\ell}$

$$\boldsymbol{M_u} = \begin{bmatrix} -2\lambda_u \boldsymbol{P_{0_u}} - \dot{\boldsymbol{P}}_{\boldsymbol{0_u}} + k_P\left(\bar{\boldsymbol{L}}\boldsymbol{P_{0_u}} + \boldsymbol{P_{0_u}}\bar{\boldsymbol{L}}\right) - \frac{k_I}{k_P}(\boldsymbol{W_e}\boldsymbol{P_{0_u}} + \boldsymbol{P_{0_u}}\boldsymbol{W_e}) & (\beta_u \mathbb{I}_{n-1} - \boldsymbol{P_{0_u}})\mathbf{Q}\mathbf{C_e} \\ \mathbf{C_e}^\top \mathbf{Q}^\top (\beta_u \mathbb{I}_{n-1} - \boldsymbol{P_{0_u}}) & 2\frac{k_P}{k_I}\beta_u \left(1 - \lambda_u \frac{k_P}{k_I}\frac{n}{n_\ell}\right) \mathbb{I}_{n-n_\ell} \end{bmatrix}.$$

Taking the Schur complement, see Definition 6.1.8 in [111], setting

$$\chi_u = \delta_u, \qquad k_P > 0, \qquad \frac{k_I}{k_P} = \eta_{I,u} \frac{n}{n_\ell} \lambda_u, \tag{C.11}$$

with design parameter $\eta_{I,u} \geq 2$, noting that $\|\mathbf{Q}\mathbf{C_e}\| \leq 1$ and $\frac{n_\ell}{n}\mathbb{I}_{n-n_\ell} \leq \boldsymbol{\Pi_e} \leq \mathbb{I}_{n-n_\ell}$, leveraging Lemma 4, and Remark 1 in [112], yields $\boldsymbol{M_u} \geq 0$, for $\lambda_u = \nu \underline{\lambda}_u$ with $\nu \in (0, 1]$. For further details see Section D.1. Finally, the definition of $\boldsymbol{P_u}$, and Lemma 4 lead to

$$a_u := \beta_u \min\left\{ \frac{\gamma_u}{\Lambda_u^2}, \left(\frac{k_P}{k_I}\right)^2 \right\}, \qquad b_u := \beta_u \max\left\{ 1, \frac{n}{n_\ell}\left(\frac{k_P}{k_I}\right)^2 \right\}.$$

□

Application of the comparison lemma, Lemma 3.4 in [108], to (C.9) yields

$$V_u(t) \leq V_u(0) e^{-2\lambda_u t}. \tag{C.12}$$

[1]This derivation is similar to the proof of Theorem 6 in [26]. However, [26] sets a specific value for $\chi_u = \delta_u$, whereas this proof holds for all $0 < \chi_u = \delta_u$. This is later leveraged in the proof for relaxed temporal constraints.

Algebraic manipulation of the components of $\boldsymbol{\zeta_u}(t)$, Equations (C.4), (C.8), and (C.12) yield the following lemma.

Lemma 6 *Assume the information flow $\mathcal{G}(t)$ satisfies Assumptions 3 through 5. Then, the origin of the system in* (C.5) *is GUES, and satisfies*

$$|\xi_i(t) - \xi_j(t)| \leq \kappa_{c,u} \|\boldsymbol{\zeta_{u_0}}\| \, e^{-\lambda_u t},$$
$$|\dot{\xi}_i(t) - \rho| \leq \kappa_{r,u} \|\boldsymbol{\zeta_{u_0}}\| \, e^{-\lambda_u t},$$

for all $t \geq 0$, and all i, $j \in \mathcal{I}$, with constants

$$\kappa_{c,u} := \sqrt{2}\kappa_u, \qquad \kappa_{r,u} := (k_P n + 1)\, \kappa_u, \qquad \textit{and} \qquad \kappa_u := \sqrt{\frac{b_u}{a_u}} \|\boldsymbol{S_u^{-1}}\| \|\boldsymbol{S_u}\|.$$

Proof. Equations (C.8) and (C.12) imply

$$\|\boldsymbol{z_u}(t)\| \leq \sqrt{\frac{b_u}{a_u}} \|\boldsymbol{z_u}(0)\| e^{-\lambda_u t}. \tag{C.13}$$

Then, leveraging the change of variables in (C.4) yields

$$\|\boldsymbol{\zeta_u}(t)\| \leq \kappa_u \|\boldsymbol{\zeta_{u_0}}\| e^{-\lambda_u t}. \tag{C.14}$$

Then, partitioning $\mathbf{Q}$ as $\mathbf{Q} = [\boldsymbol{q_1}, \boldsymbol{q_2}, \ldots, \boldsymbol{q_n}]$, noting[2] that $\|\boldsymbol{q_i} - \boldsymbol{q_j}\| = \sqrt{2}$, and using (C.14) yields

$$|\xi_i(t) - \xi_j(t)| = |(\boldsymbol{q_i}^\top - \boldsymbol{q_j}^\top)\, \boldsymbol{\zeta_c}(t)| \leq \|\boldsymbol{q_i}^\top - \boldsymbol{q_j}^\top\| \|\boldsymbol{\zeta_u}(t)\| \leq \kappa_{c,u} \|\boldsymbol{\zeta_{u_0}}\| e^{-\lambda_u t}.$$

Finally, the dynamics in (5.4) for unenforced temporal constraints and ideal target tracking, 2-norm and ∞-norm equivalence properties, $\mathbf{L}\,(t)\,\mathbf{\Pi} = \mathbf{L}\,(t)$, $\|\mathbf{L}\,(t)\,\| \leq n$, and Equation (C.14) lead to

$$|\dot{\xi}_i(t) - \rho| \leq \|\dot{\boldsymbol{\xi}}(t) - \mathbf{1_n}\rho\|_\infty \leq k_P \|\mathbf{L}\,(t)\,\| \|\boldsymbol{\zeta_c}(t)\| + \|\boldsymbol{\zeta_r}(t)\| \leq \kappa_{r,u} \|\boldsymbol{\zeta_{u_0}}\| e^{-\lambda_u t}.$$

□

[2] In the best case scenario, this property reduces the value of $\kappa_{c,u}$ proposed in [4] by 29.29%, when $n \to \infty$. In the worst case scenario, when $n = 2$, $\kappa_{c,u}$ maintains the same value as in [4].

C.2.2 UB for Temporal Error

To prove that $|\xi_i(t) - \xi_R(t)|$ is UB, the dynamics of $\zeta_t(t)$ in (5.9a) are rewritten as

$$\dot{\zeta}_t(t) = -\boldsymbol{R}\,\boldsymbol{z_u}(t), \qquad \boldsymbol{R} := \tfrac{\mathbf{1}_{\mathbf{n-n_\ell}}^\top}{n}\left[\tfrac{k_I}{k_P}\mathbf{C_e}^\top\mathbf{Q}^\top \quad \mathbb{I}_{\mathbf{n-n_\ell}}\right], \tag{C.15}$$

which leads to the following lemma.

Lemma 7 *Assume the information flow $\mathcal{G}(t)$ satisfies Assumptions 3 through 5. Then, the temporal error $|\xi_i(t) - \xi_R(t)|$ is UB, and satisfies*

$$|\xi_i(t) - \xi_R(t)| \leq \kappa_{t,u_1}\|\boldsymbol{\zeta_{u_0}}\|e^{-\lambda_u t} + \kappa_{t,u_2}\|\boldsymbol{\zeta_{u_0}}\| + |\zeta_{t_0}|, \tag{C.16}$$

for all $t \geq 0$, and all $i \in \mathcal{I}$, with constants

$$\kappa_{t,u_1} := \sqrt{\frac{n-1}{n}}\kappa_u, \qquad \textit{and} \qquad \kappa_{t,u_2} := \frac{\kappa_u}{\lambda_u}\frac{\|\boldsymbol{R}\|}{\|\boldsymbol{S_u}^{-1}\|}.$$

Proof. Algebraic manipulation of the error states in (5.6), partitioning $\mathbf{Q}$ as $\mathbf{Q} = [\boldsymbol{q_1}, \boldsymbol{q_2}, \ldots, \boldsymbol{q_n}]$, and noting that $\|\boldsymbol{q_i}\| = \sqrt{\frac{n-1}{n}}$ lead to

$$|\xi_i(t) - \xi_R(t)| \leq |\boldsymbol{q_i}^\top\boldsymbol{\zeta_c}(t) - \zeta_t(t)| \leq \sqrt{\frac{n-1}{n}}\|\boldsymbol{\zeta_u}(t)\| + |\zeta_t(t)|. \tag{C.17}$$

Integration of (C.15), and Equations (C.4) and (C.14) imply

$$|\zeta_t(t)| \leq |\zeta_{t_0}| + \frac{\kappa_u}{\lambda_u}\frac{\|\boldsymbol{R}\|}{\|\boldsymbol{S_u}^{-1}\|}\|\boldsymbol{\zeta_{u_0}}\|\left(1 - e^{-\lambda_u t}\right). \tag{C.18}$$

Combining the results above with Equation (C.14) yields Equation (C.16). □

Finally, choosing the largest lower bound for the rate of convergence, $\lambda_u = \underline{\lambda}_u$, Lemmas 6 and 7 imply Theorem 1. ■

C.3 Proof of Theorem 2

Consider the change of variables:

$$\boldsymbol{z}(t) = \boldsymbol{S}\boldsymbol{\zeta}(t), \qquad \boldsymbol{S} := \begin{bmatrix} 1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbb{I}_{\mathbf{n-1}} & \mathbf{0} \\ \frac{k_I}{k_P}\mathbf{1}_{\mathbf{n-n_\ell}} & -\frac{k_I}{k_P}\mathbf{C_e}^\top\mathbf{Q}^\top & \mathbb{I}_{\mathbf{n-n_\ell}} \end{bmatrix}. \tag{C.19}$$

Then, the dynamics in (5.7) when $\boldsymbol{u}_{\mathcal{T}_\models}(t) \equiv 0$, and the change of variables above yield

$$\dot{\boldsymbol{z}}(t) = \boldsymbol{A}_{\boldsymbol{z}}(t)\boldsymbol{z}(t), \qquad \boldsymbol{A}_{\boldsymbol{z}}(t) := \boldsymbol{S}\boldsymbol{A}(t)\boldsymbol{S}^{-1}, \tag{C.20}$$

$$\boldsymbol{A}_{\boldsymbol{z}}(t) = \begin{bmatrix} -k_R\frac{\tilde{\omega}}{n} + \frac{k_I}{k_P}\frac{n-n_\ell}{n} & \frac{1}{n}\left(k_R\boldsymbol{\omega}^\top(t) + \frac{k_I}{k_P}\boldsymbol{v}^\top\right)\mathbf{Q}^\top & -\frac{\mathbf{1}^\top_{\mathbf{n-n_\ell}}}{n} \\ \mathbf{Q}\left(k_R\boldsymbol{\omega}(t) + \frac{k_I}{k_P}\boldsymbol{v}\right) & -k_P\widehat{\mathbf{L}}(t) + \frac{k_I}{k_P}\boldsymbol{W_e} & \mathbf{QC_e} \\ \left(\frac{k_I}{k_P}\right)^2\mathbf{1}_{\mathbf{n-n_\ell}} & -\left(\frac{k_I}{k_P}\right)^2\mathbf{C_e}^\top\mathbf{Q}^\top & -\frac{k_I}{k_P}\mathbb{I}_{n-n_\ell} \end{bmatrix}.$$

Now, consider the system dynamics

$$\dot{\boldsymbol{\phi}}(t) = -k_P\widehat{\mathbf{L}}(t)\,\boldsymbol{\phi}(t), \qquad \boldsymbol{\phi}(t) \in \mathbb{R}^{n-1}, \tag{C.21}$$

where $\widehat{\mathbf{L}}(t)$ also satisfies the PE-like condition in Equation (2.16). This can be easily proven noting that $\mathbf{Q}\boldsymbol{\Omega}(t)\mathbf{Q}^\top \geq 0$. Then, a similar argument as in Appendix C.2 concludes that the origin of the system dynamics in (C.21) is GUES with guaranteed rate of convergence $\gamma_r := \frac{k_P n\mu}{(1+\Lambda_r^2 T)^2}$ and $\Lambda_r^2 := k_P n + k_R$, which leads to the following lemma.

Lemma 8 *There exists a Lyapunov function $V(t,\boldsymbol{\phi})$ for the origin of the system dynamics in* (C.21) *of the form $V(t,\boldsymbol{\phi}) = \boldsymbol{\phi}^\top\boldsymbol{P}_{\mathbf{0_r}}(t)\boldsymbol{\phi}$, where $\boldsymbol{P}_{\mathbf{0_r}}(t)$ is a continuous, piecewise differentiable, symmetric matrix that satisfies*

$$\alpha_r\mathbb{I}_{\mathbf{n-1}} \leq \boldsymbol{P}_{\mathbf{0_r}} \leq \beta_r\mathbb{I}_{\mathbf{n-1}},$$

$$\dot{\boldsymbol{P}}_{\mathbf{0_r}} - k_P\left(\widehat{\boldsymbol{L}}\boldsymbol{P}_{\mathbf{0_r}} + \boldsymbol{P}_{\mathbf{0_r}}\widehat{\boldsymbol{L}}\right) \leq -\chi_r\mathbb{I}_{\mathbf{n-1}},$$

with $\alpha_r := \frac{\chi_r}{2\Lambda_r^2}$, $\beta_r := \frac{\delta_r}{2\gamma_r}$, and any $0 < \chi_r \leq \delta_r$.

Proof. Similar to the proof of Theorem 4.12 in [108]. □

C.3.1 GUES with $\tilde{\omega}(t) \geq 1$

To address the stability of (C.20) in mode ☁, one can combine tools from Lyapunov stability, algebraic graph theory, and Lemma 8 to arrive at the following result.

Lemma 9 *Assume the information flow $\mathcal{G}(t)$ satisfies Assumptions 3 through 5, and $\tilde{\omega}(t) \geq 1$. Then, there exists a Lyapunov function V_r for the system in* (C.20) *that satisfies*

$$a_r \|\boldsymbol{z}\|^2 \leq V_r \leq b_r \|\boldsymbol{z}\|^2, \tag{C.22}$$

$$\dot{V}_r \leq -2\lambda_r V_r, \tag{C.23}$$

with known control gains k_R, k_P, and k_I, constants a_r, b_r, and a guaranteed rate of convergence λ_r that is the only positive real root of the equation

$$\lambda_r \left(1 + k_P n T + \eta_{R,r} n \xi_r T \lambda_r\right)^2 - \frac{k_P n \mu}{1 + \eta_{I,r} + f_r} = 0 \tag{C.24}$$

with known ξ_r and f_r, and design parameters $\eta_{I,r} \geq 2$ and $\eta_{R,r} > 1$.

Proof. Consider the Lyapunov candidate

$$V_r(t, \boldsymbol{z}) = \boldsymbol{z}^\top \boldsymbol{P_r}(t)\, \boldsymbol{z}, \qquad \boldsymbol{P_r} := \begin{bmatrix} n\beta_r & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{P_{0_r}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \beta_r \left(\frac{k_P}{k_I}\right)^2 \mathbb{I}_{\mathbf{n-n_\ell}} \end{bmatrix}. \tag{C.25}$$

The time derivative of V_r along the trajectories of (C.20) is

$$\dot{V}_r := \frac{\partial V_r}{\partial t} = \boldsymbol{z}^\top \left(\dot{\boldsymbol{P}}_{\boldsymbol{r}} + \boldsymbol{A}_{\boldsymbol{z}}^\top \boldsymbol{P_r} + \boldsymbol{P_r} \boldsymbol{A_z} \right) \boldsymbol{z}.$$

Then, proving that $\dot{V}_r \leq -2\lambda_r V_r$ can be reduced to ensuring positive semi-definiteness of

$$\boldsymbol{M_r} := -2\lambda_r \boldsymbol{P_r} - \dot{\boldsymbol{P}}_{\boldsymbol{r}} - \boldsymbol{A}_{\boldsymbol{z}}^\top \boldsymbol{P_r} - \boldsymbol{P_r} \boldsymbol{A_z},$$

$$\boldsymbol{M_r} = \begin{bmatrix} 2\beta_r \left(k_R\, \tilde{\omega} - \frac{k_I}{k_P}(n - n_\ell) - n\lambda_r\right) & -\left(k_R \boldsymbol{\omega}^\top + \frac{k_I}{k_P} \boldsymbol{v}^\top\right) \mathbf{Q}^\top (\beta_r \mathbb{I}_{\mathbf{n-1}} + \boldsymbol{P_{0_r}}) & \mathbf{0} \\ -(\beta_r \mathbb{I}_{\mathbf{n-1}} + \boldsymbol{P_{0_r}}) \mathbf{Q} \left(k_R \boldsymbol{\omega} + \frac{k_I}{k_P} \boldsymbol{v}\right) & -2\lambda_r \boldsymbol{P_{0_r}} - \dot{\boldsymbol{P}}_{\boldsymbol{0_r}} + k_P (\widehat{\boldsymbol{L}} \boldsymbol{P_{0_r}} + \boldsymbol{P_{0_r}} \widehat{\boldsymbol{L}}) - \frac{k_I}{k_P} (\boldsymbol{W_e} \boldsymbol{P_{0_r}} + \boldsymbol{P_{0_r}} \boldsymbol{W_e}) & (\beta_r \mathbb{I}_{\mathbf{n-1}} - \boldsymbol{P_{0_r}}) \mathbf{Q} \mathbf{C_e} \\ \mathbf{0} & \mathbf{C_e}^\top \mathbf{Q}^\top (\beta_r \mathbb{I}_{\mathbf{n-1}} - \boldsymbol{P_{0_r}}) & 2\beta_r \frac{k_P}{k_I} \left(1 - \lambda_r \frac{k_P}{k_I}\right) \mathbb{I}_{\mathbf{n-n_\ell}} \end{bmatrix}.$$

Now, iteratively applying Schur complements to $\boldsymbol{M_r}$ until the block-matrix inequality is reduced to three scalar inequalities, and setting

$$\chi_r = \delta_r, \qquad k_P > 0, \qquad \frac{k_I}{k_P} = \eta_{I,r}\lambda_r, \qquad k_R = \eta_{R,r} n \xi_r \lambda_r, \qquad \text{with} \qquad \xi_r := \eta_{I,r}\left(1 - \frac{n_\ell}{n}\right) + 1,$$

considering bounds $\|\mathbf{Q}\mathbf{C_e}\| \leq 1$, $\|\mathbf{Q}\boldsymbol{\omega}\| \leq u_\omega$, $\|\mathbf{Q}\boldsymbol{v}\| = u_v$, and $\tilde{\omega} \leq n_\ell$ with[3]

$$u_\omega^2 := \sum_{i=1}^{n_\ell} \frac{n-i}{n-i+1}, \qquad u_v^2 := \sum_{i=1}^{n_\ell} \frac{n-i}{n-i+1}\left(1 - \frac{n_\ell - i}{n-i}\right)^2,$$

leveraging Lemma 8, and using Remark 1 in [112] yields $\boldsymbol{M_r} \geq 0$ for a λ_r that satisfies Equation (C.24) with

$$f_r := \frac{(\eta_{R,r} n \xi_r u_\omega + \eta_{I,r} u_v)^2}{n\xi_r(\eta_{R,r} - 1)}.$$

For further details see Section D.2. Using Routh's stability criterion, one can verify that (C.24) always has a single positive real root. Finally, the definition of $\boldsymbol{P_r}(t)$ and Lemma 8 lead to

$$a_r := \beta_r \min\left\{n, \frac{\gamma_r}{\Lambda_r^2}, \left(\frac{k_P}{k_I}\right)^2\right\}, \qquad b_r := \beta_r \max\left\{n, \left(\frac{k_P}{k_I}\right)^2\right\}.$$

□

Application of the comparison lemma, Lemma 3.4 in [108], to (C.23) yields

$$V_r(t) \leq V_r(0) e^{-2\lambda_r t}. \tag{C.26}$$

Algebraic manipulation of the components of $\boldsymbol{\zeta}(t)$, Equations (C.19), (C.22), and (C.26) lead to the following lemma.

Lemma 10 *Assume the information flow $\mathcal{G}(t)$ satisfies Assumptions 3 through 5, and $\tilde{\omega} \geq 1$. Then, the origin of the system dynamics in* (C.20) *is GUES, and satisfies*

$$|\xi_i(t) - \xi_R(t)| \leq \kappa_{t,r} \|\boldsymbol{\zeta_0}\| \, e^{-\lambda_r t},$$
$$|\xi_i(t) - \xi_j(t)| \leq \kappa_{c,r} \|\boldsymbol{\zeta_0}\| \, e^{-\lambda_r t},$$

[3]Simpler but less tight bounds are $u_\omega, u_v \leq \sqrt{n_\ell}$.

$$|\dot{\xi}_i(t) - \rho| \leq \kappa_{r,r} \|\boldsymbol{\zeta_0}\| \, e^{-\lambda_r t},$$

for all $t \geq 0$*, and all* i*,* $j \in \mathcal{I}$*, with constants*

$$\kappa_{t,r} := \left(1 + \sqrt{\tfrac{n-1}{n}}\right) \kappa_r, \qquad \kappa_{c,r} := \sqrt{2}\kappa_r, \qquad \kappa_{r,r} := \left(k_P n + \max\left\{1, k_R\left(1 + \sqrt{\tfrac{n-1}{n}}\right)\right\}\right) \kappa_r,$$

and $\kappa_r := \sqrt{\frac{b_r}{a_r}} \|\boldsymbol{S}^{-1}\| \|\boldsymbol{S}\|$.

Proof. The change of variables in (C.19), and Equations (C.22) and (C.26) imply

$$\|\boldsymbol{\zeta}(t)\| \leq \kappa_r \|\boldsymbol{\zeta_0}\| e^{-\lambda_r t}. \tag{C.27}$$

Algebraic manipulation of the error states in (5.6), partitioning $\mathbf{Q}$ as $\mathbf{Q} = [\boldsymbol{q_1}, \boldsymbol{q_2}, \ldots, \boldsymbol{q_n}]$, noting that $\|\boldsymbol{q_i}\| = \sqrt{\frac{n-1}{n}}$, and Equation (C.27) lead to

$$|\xi_i(t) - \xi_R(t)| = |\boldsymbol{q_i}^\top \boldsymbol{\zeta_c}(t) - \zeta_t(t)| \leq \kappa_{t,r}\| \leq (\|\boldsymbol{q_i}^\top\| + 1)\|\boldsymbol{\zeta}(t)\| \leq \kappa_{t,r} \|\boldsymbol{\zeta_0}\| e^{-\lambda_r t}. \tag{C.28}$$

Next, $\mathbf{Q} = [\boldsymbol{q_1}, \boldsymbol{q_2}, \ldots, \boldsymbol{q_n}]$, noting that $\|\boldsymbol{q_i} - \boldsymbol{q_j}\| = \sqrt{2}$, and (C.27) yield

$$|\xi_i(t) - \xi_j(t)| = |(\boldsymbol{q_i}^\top - \boldsymbol{q_j}^\top)\, \boldsymbol{\zeta_c}(t)| \leq \|\boldsymbol{q_i}^\top - \boldsymbol{q_j}^\top\| \|\boldsymbol{\zeta}(t)\| \leq \kappa_{c,r} \|\boldsymbol{\zeta_0}\| e^{-\lambda_r t}.$$

Finally, the dynamics in (5.4) for relaxed temporal constraints and ideal target tracking, 2-norm and ∞-norm equivalence properties, $\mathbf{L}\,(t)\,\boldsymbol{\Pi} = \mathbf{L}\,(t)$, $\|\mathbf{L}\,(t)\,\| \leq n$, (C.27) and (C.28) imply

$$|\dot{\xi}_i(t) - \rho| \leq k_P \|\mathbf{L}\,(t)\,\| \|\boldsymbol{\zeta}(t)\| + \max\left\{k_R \|\boldsymbol{\xi}(t) - \mathbf{1_n} \xi_R(t)\|_\infty, \|\boldsymbol{\zeta_r}(t)\|_\infty\right\} \leq \kappa_{r,r} \|\boldsymbol{\zeta_0}\| e^{-\lambda_r t}.$$

□

Lemmas 9 and 10 imply Lemma 2.

The following section combines these results, Appendix C.2, and switched systems theory to analyze the switched system between modes Ø and ☁.

C.3.2 Dwell Time

The dynamics in (5.11) and the change of variables in (C.19) yield the following switched system dynamics:

$$\boldsymbol{z}(t) = \boldsymbol{A}_{\boldsymbol{z}_\gamma}(t)\boldsymbol{z}(t), \qquad \gamma \in \mathcal{S}, \tag{C.29}$$

with the same switching signal and switching times as in Section 5.3.2, $\boldsymbol{A}_{\boldsymbol{z}_\emptyset}(t) := \boldsymbol{S}\boldsymbol{A}_\emptyset(t)\boldsymbol{S}^{-1}$, and $\boldsymbol{A}_{\boldsymbol{z}_{\text{☁}}}(t) := \boldsymbol{S}\boldsymbol{A}_{\text{☁}}(t)\boldsymbol{S}^{-1}$. Recall that in mode $\emptyset$ the dynamics for the temporal error $\zeta_t(t)$ are decoupled from the coordination and rate errors $\boldsymbol{z}_{\boldsymbol{u}}(t) := \boldsymbol{S}_{\boldsymbol{u}}[\boldsymbol{\zeta}_{\boldsymbol{c}}(t)^\top, \boldsymbol{\zeta}_{\boldsymbol{r}}(t)^\top]^\top$, and thus the stability analyses for $\zeta_t(t)$ and $\boldsymbol{z}_{\boldsymbol{u}}(t)$ were performed separately. However, in mode ☁ the Lyapunov function V_r is expressed in terms of $\boldsymbol{z}(t)$, a state that lumps together the temporal, coordination, and rate errors. As a result, to find the evolution of V_r in mode $\emptyset$, one must find a relation between all these states. To this end, define $\tilde{\boldsymbol{z}}(t) := [\zeta_t(t), \boldsymbol{z}_{\boldsymbol{u}}^\top(t)]^\top$ and the corresponding change of variables

$$\boldsymbol{z}(t) = \boldsymbol{T}_{\boldsymbol{z}}\tilde{\boldsymbol{z}}(t), \qquad \boldsymbol{T}_{\boldsymbol{z}} := \begin{bmatrix} 1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbb{I}_{n-1} & \mathbf{0} \\ \frac{k_I}{k_P}\mathbf{1}_{\mathbf{n}-\mathbf{n}_\ell} & \mathbf{0} & \mathbb{I}_{n-n_\ell} \end{bmatrix}. \tag{C.30}$$

To rewrite V_r in terms of V_u, define $\tilde{\boldsymbol{P}}_{\boldsymbol{r}}(t) := \boldsymbol{T}_{\boldsymbol{z}}^\top \boldsymbol{P}_{\boldsymbol{r}}(t)\boldsymbol{T}_{\boldsymbol{z}}$ such that $V_r(t, \tilde{\boldsymbol{z}}) = \tilde{\boldsymbol{z}}^\top \tilde{\boldsymbol{P}}_{\boldsymbol{r}}(t)\tilde{\boldsymbol{z}}$. Then, the evolution of V_r in mode $\emptyset$ can be expressed as

$$V_r(t, \zeta_t, \boldsymbol{\zeta}_{\boldsymbol{u}}) = V_u(t, \boldsymbol{z}_{\boldsymbol{u}}) + \sigma(t, \zeta_t, \boldsymbol{z}_{\boldsymbol{u}}), \tag{C.31}$$

where σ is a storage function

$$\sigma(t, \zeta_t, \boldsymbol{z}_{\boldsymbol{u}}) := (2n - n_\ell)\beta_r\zeta_t^2(t) + 2\beta_r\frac{k_P}{k_I}\zeta_t(t)\begin{bmatrix}\mathbf{0} & \mathbf{1}_{\mathbf{n}-\mathbf{n}_\ell}^\top\end{bmatrix}\boldsymbol{z}_{\boldsymbol{u}}(t) + \boldsymbol{z}_{\boldsymbol{u}}^\top(t)\boldsymbol{\Delta}\boldsymbol{P}(t)\,\boldsymbol{z}_{\boldsymbol{u}}(t),$$

with

$$\boldsymbol{\Delta}\boldsymbol{P}(t) := \begin{bmatrix} \boldsymbol{P}_{\mathbf{0}_{\boldsymbol{r}}}(t) - \boldsymbol{P}_{\mathbf{0}_{\boldsymbol{u}}}(t) & \mathbf{0} \\ \mathbf{0} & \left(\frac{k_P}{k_I}\right)^2\left(\beta_r\mathbb{I}_{n-n_\ell} - \beta_u\boldsymbol{\Pi}_{\boldsymbol{e}}^{-1}\right) \end{bmatrix}.$$

Equations (C.18) and (C.13) imply that $\zeta_t(t)$ and $\boldsymbol{z}_{\boldsymbol{u}}(t)$ are UB and GUES in mode $\emptyset$, respectively. Hence, Equation (C.31) proves that V_r can increase in mode $\emptyset$, and this increase is bounded.

The following lemma uses the results above, switched systems theory, and Lemmas 4 through 9 to prove that if system (C.29) persistently switches between modes Ø and ☁, that is $n_s \to \infty$ as $t \to \infty$, then there is a dwell time τ_{R_1} that renders the origin of the switched system in (C.29) with slow switching constraints in (5.12) asymptotically stable.

Lemma 11 *Assume the information flow $\mathcal{G}(t)$ satisfies Assumptions 3 through 5, and consider the switched system* (C.29)*, with slow switching constraints* (5.12)*, and dwell time*

$$\tau_{R_1} = \eta_\tau + \max\left\{0, \frac{1}{2\lambda_r}\ln a_\tau\right\}, \tag{C.32}$$

with known constant a_τ, and design parameter $\eta_\tau > 0$. Then the temporal, coordination, and rate errors satisfy

$$\begin{aligned} |\xi_i(t_{2k}) - \xi_R(t_{2k})| &\leq \kappa_{t,r}\|\boldsymbol{\zeta_0}\|\, e^{-\lambda_r \eta_\tau k}, \\ |\xi_i(t_{2k}) - \xi_j(t_{2k})| &\leq \kappa_{c,r}\|\boldsymbol{\zeta_0}\|\, e^{-\lambda_r \eta_\tau k}, \\ |\dot{\xi}_i(t_{2k}) - \rho| &\leq \kappa_{r,r}\|\boldsymbol{\zeta_0}\|\, e^{-\lambda_r \eta_\tau k}. \end{aligned} \tag{C.33}$$

Hence, if $n_s \to \infty$ as $t \to \infty$ the origin of (C.29) *is asymptotically stable.*

Proof. Using Equations (C.26) and (5.12), and leveraging Figure C.1 to aid in the visualization of this proof, the evolution of V_r in mode ☁ from t_{2k-1} to t_{2k} satisfies

$$V_r(t_{2k}) < V_r(t_{2k-1})e^{-2\lambda_r \tau_{R_1}}.$$

Define the time spent in mode Ø for the kth Ø-☁ cycle as $\Delta t_{u,k} := t_{2k-1} - t_{2k-2}$, and use (C.12)

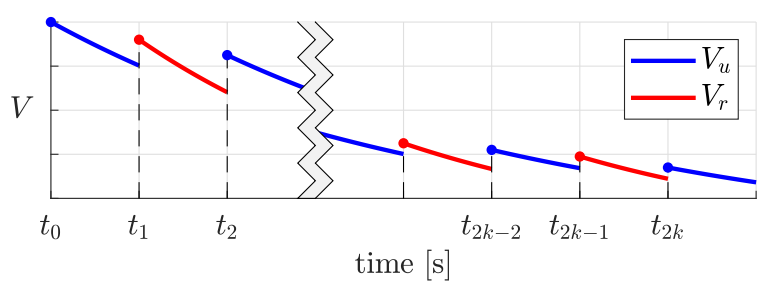


Figure C.1: Evolution of the Lyapunov functions V_u and V_r.

and (C.31) to bound the evolution of V_r in mode Ø from t_{2k-2} to t_{2k-1}

$$V_r(t_{2k-1}) \leq V_u(t_{2k-2})e^{-2\lambda_u \Delta t_{u,k}} + \sigma(t_{2k-1}).$$

Using Lemmas 4 and 8, noting that $\boldsymbol{\Pi}_{\boldsymbol{e}}^{-1} \geq \mathbb{I}_{n-n_\ell}$, and letting $\beta_r = \alpha_u$, one can prove that $\boldsymbol{\Delta P}(t) \leq 0$, which leads to the following bound:

$$\sigma(t) \leq (2n - n_\ell)\beta_r \zeta_t^2(t) + 2\beta_r \frac{k_P}{k_I}\sqrt{n - n_\ell}|\zeta_t(t)|\|\boldsymbol{z_u}(t)\|.$$

The results above, the dynamics in (C.15), Lemmas 5 and 9, the proof of Lemma 7, and Equations (C.30) and (5.12) imply

$$V_r(t_{2k}) - V_r(t_{2k-2}) < \left(a_\tau\, e^{-2\lambda_r \tau_{R_1}} - 1\right) V_r(t_{2k-2}),$$

where

$$a_\tau := \frac{1}{a_r} \sup_{\Delta t_{u,k} \geq 0} \left((2n - n_\ell)\beta_r f^2(\Delta t_{u,k}) + 2\sqrt{n - n_\ell}\, \beta_r \frac{k_P}{k_I}\sqrt{\frac{b_u}{a_u}}\|\boldsymbol{T_z}^{-1}\| f(\Delta t_{u,k}) + b_u \|\boldsymbol{T_z}^{-1}\|^2 e^{-2\lambda_u \Delta t_{u,k}} \right),$$

with

$$f(\Delta t_{u,k}) := 1 + d_a\left(1 - e^{-\lambda_u \Delta t_{u,k}}\right), \qquad d_a := \frac{\|\boldsymbol{R}\|}{\lambda_u}\sqrt{\frac{b_u}{a_u}}\|\boldsymbol{T_z}^{-1}\|.$$

Then,the sequence $V_r(t_{2k-2})$, $V_r(t_{2k})$, ... is decreasing and positive for the dwell time in (C.32). Next, an inductive argument leads to $V_r(t_{2k}) < V_r(0)\, a_\tau^k\, e^{-2\lambda_r k \tau_{R_1}}$, and therefore

$$V_r(t_{2k}) < V_r(0)e^{-2k\lambda_r \eta_\tau}. \tag{C.34}$$

Thus, if $k \to \infty$ as $t \to \infty$, then $V_r(t_{2k})$ converges asymptotically to the origin. A similar process as in the proof of Lemma 10 can be used to derive the inequalities in (C.33). □

Remark 10 *Lemma 11 does not consider the link-weight logic for relaxed temporal constraints described in Table 5.1. It simply assumes infinitely many switches subject to the slow switching constraints in* (5.12)*. However, the logic for relaxed temporal constraints in Table 5.1 is state-dependent, but also implements a temporal logic that guarantees* (5.12) *is met.*

The following section leverages this result to prove that the system dynamics in (C.29), subject to the link-weight logic for relaxed temporal constraints in Table 5.1, yields finitely many switches and satisfies the design objectives defined in Equation (2.15).

C.3.3 Switching Logic

The next lemma shows that there exists a sufficiently small set containing the origin of (C.29) such that no agent can escape the temporal window. To this end, define $\underline{\Delta}_t$ satisfying $0 < \underline{\Delta}_t \leq \Delta_t(t)$.

Lemma 12 *Assume the information flow $\mathcal{G}(t)$ satisfies Assumptions 3 through 5, and consider the switched system* (C.29)*, with slow switching constraints* (5.12)*. If $n_s \to \infty$ as $t \to \infty$, then there exist a time t_δ and a known constant δ such that*

$$V_r(t_\delta) \leq \delta^2 \Rightarrow |\xi_i(t) - \xi_R(t)| < \Delta_t(t), \qquad \forall t > t_\delta, \qquad \forall i \in \mathcal{I}. \tag{C.35}$$

Proof. Lemma 11 implies that there exists a time t_{Δ_1} when the last peer crosses the boundary $|\xi_i(t_{\Delta_1}) - \xi_R(t_{\Delta_1})| = \underline{\Delta}_t$, as sketched in Figure C.2. Now, assume

$$\exists\, t_{\Delta_2} > t_\delta \;\mid\; |\xi_i(t_{\Delta_2}) - \xi_R(t_{\Delta_2})| = \Delta_t(t_{\Delta_2}), \qquad i \in \mathcal{I},$$

as depicted in Figure C.2. Algebraic manipulation of the error states in (5.6) and $\|\boldsymbol{q_i}\| = \sqrt{\frac{n-1}{n}}$ yield

$$\Delta_t(t_{\Delta_2}) \leq \sqrt{\frac{n-1}{n}}\|\boldsymbol{\zeta_c}(t_{\Delta_2})\| + |\zeta_t(t_{\Delta_2})|.$$

Considering the following cases

$$\begin{aligned}
&t_{\Delta_2} > t_{\Delta_1} + \tau_{R_1} \;\wedge\; t_\delta > t_{\Delta_1} + \tau_{R_1}, \\
&t_{\Delta_2} > t_{\Delta_1} + \tau_{R_1} \;\wedge\; t_\delta \leq t_{\Delta_1} + \tau_{R_1}, \\
&t_{\Delta_2} \leq t_{\Delta_1} + \tau_{R_1},
\end{aligned}$$

Lemmas 5 and 9, Equations (C.12), (C.26), and (C.30), noting that $\frac{b_u}{a_u} \geq 1$, $\|\boldsymbol{T_z}^{-1}\| \geq 1$, and

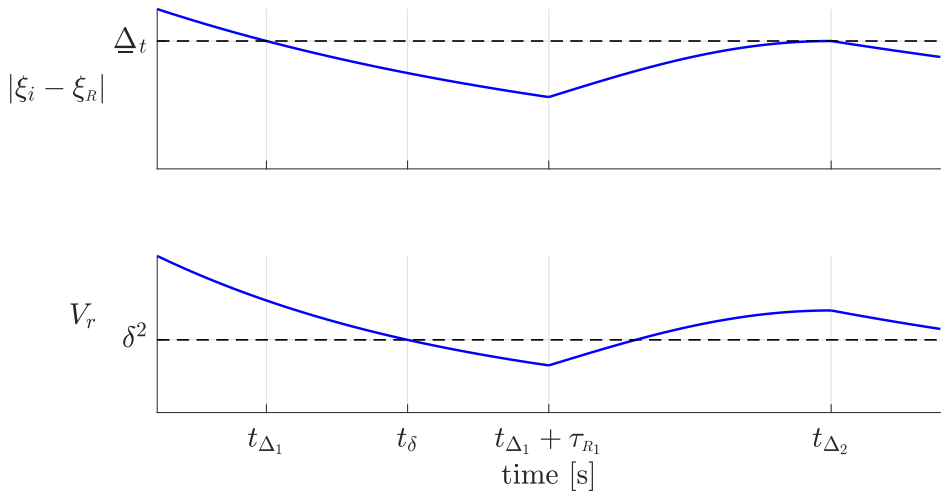


Figure C.2: Evolution of V_r and $|\xi_i(t) - \xi_R(t)|$.

$V_r(t_\delta) \leq \delta^2$ imply

$$\|\boldsymbol{\zeta_c}(t_{\Delta_2})\| \leq \sqrt{\frac{b_u}{a_u}}\|\boldsymbol{T_z}^{-1}\|\frac{\delta}{\sqrt{a_r}},$$
$$|\zeta_t(t_{\Delta_2})| \leq \left(1 + \frac{\|\boldsymbol{R}\|}{\lambda_u}\sqrt{\frac{b_u}{a_u}}\|\boldsymbol{T_z}^{-1}\|\right)\frac{\delta}{\sqrt{a_r}}.$$

The results above lead to $\Delta_t(t_{\Delta_2}) \leq a_\delta\,\delta$ with

$$a_\delta := \frac{1}{\sqrt{a_r}}\left(1 + \frac{\|\boldsymbol{R}\|}{\lambda_u}\sqrt{\frac{b_u}{a_u}}\|\boldsymbol{T_z}^{-1}\| + \sqrt{\frac{n-1}{n}\frac{b_u}{a_u}}\|\boldsymbol{T_z}^{-1}\|\right).$$

Then, choosing a sufficiently small $\delta = \epsilon\frac{\Delta_{t_\ell}}{a_\delta}$ with $\epsilon \in (0,1)$ leads to a contradiction, $\Delta_t(t_{\Delta_2}) \not\leq \epsilon\underline{\Delta}_t$. Thus, there exist t_δ and δ such that (C.35) holds. □

The following result merges Lemmas 11 and 12 with the link-weight logic for relaxed temporal constraints in Table 5.1.

Lemma 13 *Assume the information flow $\mathcal{G}(t)$ satisfies Assumptions 3 through 5, and consider the switched system* (C.29) *with the link-weight logic for relaxed temporal constraints in Table 5.1. Then*

$$\exists\, t_{\Delta_t} \leq t_\delta \;\mid\; |\xi_i(t) - \xi_R(t)| < \Delta_t(t) \;\wedge\; \gamma(t) = \emptyset, \qquad \forall t \geq t_{\Delta_t}, \qquad \forall i \in \mathcal{I}.$$

Proof. System (C.29) and the logic for relaxed constraints in Table 5.1 can be represented as a

finite state machine, shown in Figure C.3. As a result, the proof is divided into the following cases[4]:

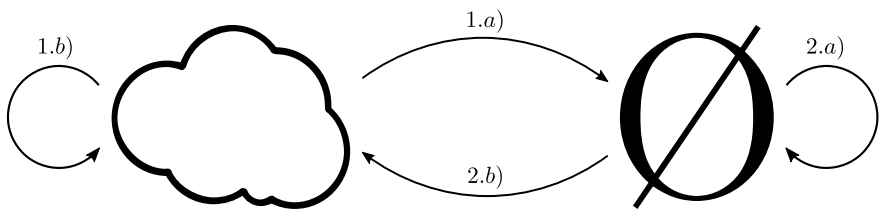


Figure C.3: Finite state machine.

1) If (C.29) is in mode ☁, then Lemma 10 ensures that $|\xi_i(t) - \xi_R(t)|$ converges to 0 for all $i \in \mathcal{I}$. However, only the link peers implement the switching logic, and thus $\exists\, t \geq 0$ such that

$$|\xi_i(t) - \xi_R(t)| < \Delta_t(t), \qquad \forall i \in \mathcal{I}_\ell, \tag{C.36}$$

Since only the link peers satisfy the condition above, some end peers could be outside of the temporal window. This gives rise to two possibilities:

a) After τ_{R_1} seconds the link peers still satisfy (C.36), and the logic in Table 5.1 switches (C.29) to mode Ø.

b) Otherwise, the system remains in mode ☁. However, Lemma 10 implies that (C.29) cannot remain in this mode indefinitely, and the system eventually switches to mode Ø.

2) If (C.29) is in mode Ø, then Lemma 6 ensures that $|\xi_i(t) - \xi_j(t)|$ converges to 0 for all $i,\ j \in \mathcal{I}$, whereas Lemma 7 shows that $|\xi_i(t) - \xi_R(t)|$ can increase by a bounded amount, which leads to two possibilities:

a) All agents converge to the temporal window

$$|\xi_i(t) - \xi_R(t)| < \Delta_t(t), \qquad \forall i \in \mathcal{I}, \qquad \forall t \geq t_{\Delta_t}, \tag{C.37}$$

with $t_{\Delta_t} \leq t_\delta$, and the system remains in mode Ø.

[4]This can be proven more succinctly by contradiction. However, the following proof has been chosen because it provides a better insight into the behavior of the distributed switched system.

b) Otherwise, some peers escape the temporal window and the system switches to mode ☁. However, Lemmas 11 and 12 imply that (C.29) cannot switch between modes Ø and ☁ indefinitely. Thus, $\exists\, t_{\Delta_t} \leq t_\delta$ such that (C.37) holds.

Hence, mode Ø is the only stable state in the finite state machine shown in Figure C.3. □

Next, Lemmas 12 and 13 are leveraged to find an upper bound for the number of mode switches required to ensure that no vehicle escapes the desired temporal window, $|\xi_i(t) - \xi_R(t)| < \Delta_t(t)$.

Lemma 14 *Assume the information flow $\mathcal{G}(t)$ satisfies Assumptions 3 through 5, and consider the switched system* (C.29)*, with the link-weight logic for relaxed temporal constraints in Table 5.1. Then*

$$n_s < 2\left\lceil \frac{1}{\lambda_r \eta_\tau} \ln\left(\kappa_{n_s} \frac{\|\boldsymbol{\zeta_0}\|}{\underline{\Delta}_t}\right)\right\rceil \tag{C.38}$$

with known constant $\kappa_{n_s} := \frac{a_\delta}{\epsilon}\sqrt{b_r}$, and $\epsilon \in (0,1)$.

Proof. Assuming $\gamma(t_0) = \emptyset$, one can use Equations (C.22) and (C.34) as well as Lemma 12 to find an upper bound for the number of Ø-☁ cycles required to ensure $V_r(t_{2k}) \leq \delta^2$, and thus guarantee that $|\xi_i(t) - \xi_R(t)| \leq \underline{\Delta}_t$ for all $i \in \mathcal{I}$, which yields

$$k \leq \frac{1}{\lambda_r \eta_\tau} \ln\left(\sqrt{b_r}\frac{\|\boldsymbol{\zeta_0}\|}{\delta}\right)$$

Then, the expression for δ defined in the proof of Lemma 12, and the relation between k and n_s yields (C.38). A similar procedure can be applied assuming $\gamma(t_0) =$ ☁ to check that inequality (C.38) still holds. □

Lemmas 6, 10, 11, 13, and 14 imply Theorem 2. However, Lemmas 5 and 9 define different sets for the control gain k_I. The following section finds a common set of gains that works for modes Ø and ☁ simultaneously.

C.3.4 Compatible Control Gains for the Switched System

Choose values for the tuning parameters k_P, $\eta_{I,r}$, and $\eta_{R,r}$ satisfying the conditions in Lemma 9, and use (C.24) to find λ_r. Note that Lemma 5 holds for all $\lambda_u := \nu\bar{\lambda}_u$, with $\nu \in (0,1]$ being a

design parameter, and set

$$\frac{k_I}{k_P} = \eta_{I,u}\frac{n}{n_\ell}\nu\bar{\lambda}_u = \eta_{I,r}\lambda_r.$$

Then, solving for $\eta_{I,u}$, noting that (C.24) is equivalent to $\lambda_r = \gamma_r(1+\eta_{I,r}+f_r)^{-1}$, and using the expressions for γ_u and γ_r, one can prove that any $\nu > 0$ satisfying

$$\nu \leq \min\left\{1, \frac{\eta_{I,r}}{2}\frac{\lambda_r}{\gamma_u}\left(2+\frac{n_\ell}{n}\right)\right\},$$

guarantees that $\eta_{I,u} \geq 2$. This provides a set of gains that simultaneously satisfies the conditions in Lemmas 5 and 9. ■

C.4 Proof of Theorem 3

The proof is a particular case of Lemmas 8, 9, and 10, when $\tilde{\omega}(t) \equiv n_\ell$, and $\boldsymbol{\omega}(t) \equiv \boldsymbol{v}$, that leads to

$$\begin{aligned}
\xi_s &:= \eta_{I,s}\left(1-\tfrac{n_\ell}{n}\right)+1, & f_s &:= \tfrac{u_v^2}{n}\frac{\left(\eta_{R,s}\frac{n}{n_\ell}\xi_s+\eta_{I,s}\right)^2}{\xi_s(\eta_{R,s}-1)},\\
\kappa_{t,s} &:= \left(1+\sqrt{\tfrac{n-1}{n}}\right)\kappa_s, & \kappa_{c,s} &:= \sqrt{2}\,\kappa_s,\\
\kappa_{r,s} &:= \left(k_P n + \max\left\{1, k_R\left(1+\sqrt{\tfrac{n-1}{n}}\right)\right\}\right)\kappa_s, & \kappa_s &:= \sqrt{\tfrac{b_s}{a_s}}\|\boldsymbol{S}^{-1}\|\|\boldsymbol{S}\|,
\end{aligned}$$

with

$$\frac{b_s}{a_s} = \frac{\max\left\{n, \left(\frac{k_P}{k_I}\right)^2\right\}}{\min\left\{n, \frac{\gamma_s}{\Lambda_s^2}, \left(\frac{k_P}{k_I}\right)^2\right\}}, \qquad \gamma_s := \frac{k_P n\mu}{(1+\Lambda_s^2 T)^2}, \qquad \Lambda_s^2 := k_P n + k_R,$$

and control gains

$$k_P > 0, \qquad \frac{k_I}{k_P} = \eta_{I,s}\,\lambda_s, \qquad k_R = \eta_{R,s}\frac{n}{n_\ell}\xi_s\lambda_s.$$

■

C.5 Proof of Theorem 4

This section builds upon the proof for Theorem 1, where unenforced temporal constraints with ideal target-tracking capabilities are addressed, and leverages the same change of variables, and Lyapunov function candidate as in Section C.2.

C.5.1 ISS of the Coordination and Rate Errors

Using the dynamics in (6.2b) and the change of variables in Equation (C.4) yields the system dynamics

$$\dot{\boldsymbol{z}}_{\boldsymbol{u}}(t) = \boldsymbol{A}_{\boldsymbol{z_u}}(t)\boldsymbol{z}_{\boldsymbol{u}}(t) + \boldsymbol{B}_{\boldsymbol{z_u}}\boldsymbol{u}_{\mathcal{T}_{\models}}(t), \tag{C.39}$$

where $\boldsymbol{A}_{\boldsymbol{z_u}}(t)$ is given in Equation (C.5), and $\boldsymbol{B}_{\boldsymbol{z_u}} := \boldsymbol{S}_{\boldsymbol{u}}\boldsymbol{B}_{\boldsymbol{u}}$. The following lemma uses the dynamics in (C.39) and the Lyapunov function in (C.10) to prove that the system with the target-tracking error feedback is ISS with respect to $\boldsymbol{u}_{\mathcal{T}_{\models}}(t)$. See [113] for the characterization of ISS-Lyapunov functions.

Lemma 15 *Assume the information flow $\mathcal{G}(t)$ satisfies Assumptions 3 through 5. Then, there exists a function W_u for the system in* (C.39) *that satisfies*

$$\sqrt{a_u}\|\boldsymbol{z}_{\boldsymbol{u}}\| \leq W_u \leq \sqrt{b_u}\|\boldsymbol{z}_{\boldsymbol{u}}\|, \tag{C.40}$$

$$\dot{W}_u \leq -\lambda_u W_u + \frac{1}{\sqrt{a_u}}\|\boldsymbol{P}_{\boldsymbol{u}}\boldsymbol{B}_{\boldsymbol{z_u}}\|\|\boldsymbol{u}_{\mathcal{T}_{\models}}\|, \tag{C.41}$$

with known control gains k_P and k_I, constants a_u and b_u, and rate of convergence $\lambda_u = \nu\underline{\lambda}_u$.

Proof. The time derivative of the Lyapunov function candidate V_u, defined in Equation (C.10), along the trajectories of (C.39) is

$$\dot{V}_u = \frac{\partial V_u}{\partial t} = \boldsymbol{z}_{\boldsymbol{u}}^{\top}\left(\dot{\boldsymbol{P}}_{\boldsymbol{u}} + \boldsymbol{A}_{\boldsymbol{z_u}}^{\top}\boldsymbol{P}_{\boldsymbol{u}} + \boldsymbol{P}_{\boldsymbol{u}}\boldsymbol{A}_{\boldsymbol{z_u}}\right)\boldsymbol{z}_{\boldsymbol{u}} + \boldsymbol{u}_{\mathcal{T}_{\models}}^{\top}\boldsymbol{B}_{\boldsymbol{z_u}}^{\top}\boldsymbol{P}_{\boldsymbol{u}}\boldsymbol{z}_{\boldsymbol{u}} + \boldsymbol{z}_{\boldsymbol{u}}^{\top}\boldsymbol{P}_{\boldsymbol{u}}\boldsymbol{B}_{\boldsymbol{z_u}}\boldsymbol{u}_{\mathcal{T}_{\models}}.$$

A similar argument as in the proof of Lemma 5 leads to the following bound:

$$\dot{V}_u \leq -2\lambda_u V_u + 2\|\boldsymbol{P}_{\boldsymbol{u}}\boldsymbol{B}_{\boldsymbol{z_u}}\|\|\boldsymbol{u}_{\mathcal{T}_{\models}}\|\sqrt{\frac{V_u}{a_u}}. \tag{C.42}$$

To obtain a linear differential inequality, introduce the change of variables

$$W_u(t, \boldsymbol{z}_{\boldsymbol{u}}) := \sqrt{V_u(t, \boldsymbol{z}_{\boldsymbol{u}})},$$

and note that its time derivative is $\dot{W}_u = \frac{\dot{V}_u}{2\sqrt{V_u}}$. Similar to the proof in Section C.1, this yields

Equation (C.41). Equation (C.40) follows from the definition of W_u and Equation (C.8). $\square$

Application of the comparison lemma, Lemma 3.4 in [108], produces the following bound:

$$W_u(t) \leq W_u(0)e^{-\lambda_u t} + \frac{1}{\sqrt{a_u}}\|\boldsymbol{P_u B_{z_u}}\| \int_0^t e^{-\lambda_u(t-\tau)}\|\boldsymbol{u}_{\mathcal{T}_{\models}}(\tau)\| \, \mathrm{d}\tau.$$

Next, propagating the effect of $\|\boldsymbol{u}_{\mathcal{T}_{\models}}(t)\|$ as a perturbation of the form detailed in Equation (6.1) yields

$$\begin{aligned} W_u(t) &\leq W_u(0)e^{-\lambda_u t} + \kappa_{W_u}\left(\|\boldsymbol{e_{p_0}}\| \int_0^t e^{-\lambda_u(t-\tau)}e^{-\underline{k}_{PF}\tau}\mathrm{d}\tau + \frac{\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|}{\underline{k}_{PF}} \int_0^t e^{-\lambda_u(t-\tau)}\left(1 - e^{-\bar{k}_{PF}\tau}\right)\mathrm{d}\tau\right) \\ &\leq W_u(0)e^{-\lambda_u t} + \kappa_{W_u}\frac{\|\boldsymbol{e_{p_0}}\|}{\underline{k}_{PF} - \lambda_u}\left(e^{-\lambda_u t} - e^{-\underline{k}_{PF}t}\right) + \dots \\ &\qquad + \kappa_{W_u}\frac{\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|}{\underline{k}_{PF}}\left(\frac{1}{\lambda_u}\left(1 - e^{-\lambda_u t}\right) - \frac{1}{\bar{k}_{PF} - \lambda_u}\left(e^{-\lambda_u t} - e^{-\bar{k}_{PF}t}\right)\right) \end{aligned}$$

where $\kappa_{W_u} := k_{e_p}\frac{b_u}{\sqrt{a_u}}\|\boldsymbol{B_{z_u}}\|$. Notice that the exponentially decaying term

$$-\frac{1}{\bar{k}_{PF} - \lambda_u}\left(e^{-\lambda_u t} - e^{-\bar{k}_{PF}t}\right) \leq 0, \qquad \forall t \geq 0.$$

Henceforth, for the sake of simplicity, and at the expense of a less tight transient bound, the following expression will be used:

$$W_u(t) \leq W_u(0)e^{-\lambda_u t} + \kappa_{W_u}\left(\frac{\|\boldsymbol{e_{p_0}}\|}{\underline{k}_{PF} - \lambda_u}\left(e^{-\lambda_u t} - e^{-\underline{k}_{PF}t}\right) + \frac{\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|}{\underline{k}_{PF}\lambda_u}\right). \tag{C.43}$$

The next lemma leverages Equation (C.43) to analyze the effect on the individual coordination and rate errors of a perturbation with the structure detailed in Equation (6.1).

Lemma 16 *Assume the underlying speed-tracking controller for all agents satisfies Assumption 1, and the information flow $\mathcal{G}(t)$ satisfies Assumptions 3 through 5. Then, the origin of the system dynamics is ISS with respect to $\boldsymbol{e_v}$, and satisfies*

$$\begin{aligned} |\xi_i(t) - \xi_j(t)| &\leq \kappa_{c,u}\|\boldsymbol{\zeta_{u_0}}\|e^{-\lambda_u t} + \frac{\kappa_{c,u_p}}{\underline{k}_{PF} - \lambda_u}\|\boldsymbol{e_{p_0}}\|\left(e^{-\lambda_u t} - e^{-\underline{k}_{PF}t}\right) + \kappa_{c,u_v}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|, \qquad \text{(C.44)} \\ |\dot{\xi}_i(t) - \rho| &\leq \kappa_{r,u}\|\boldsymbol{\zeta_{u_0}}\|e^{-\lambda_u t} + \frac{\kappa_{r,u_p}}{\underline{k}_{PF} - \lambda_u}\|\boldsymbol{e_{p_0}}\|\left(e^{-\lambda_u t} - e^{-\underline{k}_{PF}t}\right) + \kappa_{r,u_v}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| + \dots \end{aligned}$$

$$+ k_{e_p}\|\boldsymbol{e_{p_0}}\|e^{-\underline{k}_{PF}t} + \frac{k_{e_p}}{\underline{k}_{PF}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|, \tag{C.45}$$

for all $t \geq 0$, all i, $j \in \mathcal{I}$, with constants

$$\kappa_{c,u} := \sqrt{2}\kappa_u, \qquad \kappa_{c,u_p} := \sqrt{2}\kappa_{u_p}, \qquad \kappa_{c,u_v} := \sqrt{2}\kappa_{u_v},$$
$$\kappa_{r,u} := (k_P n + 1)\kappa_u, \qquad \kappa_{r,u_p} := (k_P n + 1)\kappa_{u_p}, \qquad \kappa_{r,u_v} := (k_P n + 1)\kappa_{u_v}.$$

Proof. Equation (C.40) and the change of variables in Equation (C.4) imply

$$\|\boldsymbol{\zeta_u}(t)\| \leq \kappa_u\|\boldsymbol{\zeta_{u_0}}\|e^{-\lambda_u t} + \frac{\kappa_{u_p}}{\underline{k}_{PF} - \lambda_u}\|\boldsymbol{e_{p_0}}\|\left(e^{-\lambda_u t} - e^{-\underline{k}_{PF}t}\right) + \kappa_{u_v}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|, \tag{C.46}$$

where κ_u is defined in Lemma 6, and $\kappa_{u,p}$ and $\kappa_{u,v}$ are

$$\kappa_{u_p} := k_{e_p}\frac{b_u}{a_u}\|\boldsymbol{S_u}^{-1}\|\|\boldsymbol{B_{z_u}}\|, \qquad \kappa_{u_v} := \frac{1}{\lambda_u}\frac{k_{e_p}}{\underline{k}_{PF}}\frac{b_u}{a_u}\|\boldsymbol{S_u}^{-1}\|\|\boldsymbol{B_{z_u}}\|.$$

A similar argument as in the proof of Lemma 6 yields the individual coordination and rate errors in (C.44) and (C.45). □

C.5.2 Drift of the Temporal Errors

To find a bound for the temporal error $|\xi_i(t) - \xi_R(t)|$, recall the bound in Equation (C.17)

$$|\xi_i(t) - \xi_R(t)| \leq \sqrt{\frac{n-1}{n}}\|\boldsymbol{\zeta_u}(t)\| + |\zeta_t(t)|,$$

and study the evolution of $|\zeta_t(t)|$ with the target-tracking feedback. To this end, consider the dynamics in (6.2a) and the change of variables in (C.4) to rewrite the dynamics as

$$\dot{\zeta}_t(t) = -\boldsymbol{R}\boldsymbol{z_u}(t) - \frac{1}{n}\mathbf{1}_{\mathbf{n}-\mathbf{n}_\ell}^\top \boldsymbol{u_{\tau_\vDash}}(t), \tag{C.47}$$

where $\boldsymbol{R}$ is defined in (C.15). The following lemma leverages Equation (C.47) to analyze the effect on the individual temporal error of a perturbation with the structure detailed in Equation (6.1).

Lemma 17 *Assume the underlying speed-tracking controller for all agents satisfies Assumption 1,*

and the information flow $\mathcal{G}(t)$ satisfies Assumptions 3 through 5. Then, the error $|\xi_i(t) - \xi_R(t)|$ can drift with time and satisfies:

$$\begin{aligned}|\xi_i(t) - \xi_R(t)| \leq |\zeta_{t_0}| &+ \|\boldsymbol{\zeta_{u_0}}\| \left(\kappa_{t,u_1} e^{-\lambda_u t} + \kappa_{t,u_2}\right) + \dots \\ &+ \|\boldsymbol{e_{p_0}}\| \left(\frac{\kappa_{t,u_{p_1}}}{\underline{k}_{PF} - \lambda_u}\left(e^{-\lambda_u t} - e^{-\underline{k}_{PF} t}\right) + \kappa_{t,u_{p_2}}\right) + \dots \\ &+ \|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \left(\kappa_{t,u_{v_1}} + \kappa_{t,u_{v_2}} t\right),\end{aligned}$$

for all $t \geq 0$, all $i \in \mathcal{I}$, with constants

$$\begin{aligned}&\kappa_{t,u_1} := \sqrt{\tfrac{n-1}{n}}\kappa_u, \qquad \kappa_{t,u_{p_1}} := \sqrt{\tfrac{n-1}{n}}\kappa_{u_p}, \qquad \kappa_{t,u_{v_1}} := \sqrt{\tfrac{n-1}{n}}\kappa_{u_v}, \\ &\kappa_{t,u_2} := \kappa_{t_u}, \qquad \kappa_{t,u_{p_2}} := \kappa_{t_p}, \qquad \kappa_{t,u_{v_2}} := \kappa_{t_v}.\end{aligned}$$

Proof. The 1-norm and 2-norm equivalence properties, and Equation (C.47) yield

$$|\dot{\zeta}_t(t)| \leq \|\boldsymbol{R}\|\|\boldsymbol{z_u}(t)\| + \frac{\sqrt{n}}{n}\|\boldsymbol{u}_{\boldsymbol{\mathcal{T}}_{\models}}(t)\|.$$

Integrating the result above leads to

$$|\zeta_t(t)| \leq |\zeta_{t_0}| + \|\boldsymbol{R}\| \int_0^t \|\boldsymbol{z_u}(\tau)\| \,\mathrm{d}\tau + \frac{\sqrt{n}}{n}\int_0^t \|\boldsymbol{u}_{\boldsymbol{\mathcal{T}}_{\models}}(\tau)\| \,\mathrm{d}\tau.$$

Bounds for the integral expressions above can be derived using the definition of W_u, and Equations (6.1), (C.8), and (C.43)

$$\begin{aligned}\int_0^t \|\boldsymbol{z_u}(\tau)\|\mathrm{d}\tau &\leq \kappa_u \frac{\|\boldsymbol{\zeta_{u_0}}\|}{\|\boldsymbol{S_u}^{-1}\|}\int_0^t e^{-\lambda_u \tau}\mathrm{d}\tau + \frac{\kappa_{u_p}}{\underline{k}_{PF} - \lambda_u}\frac{\|\boldsymbol{e_{p_0}}\|}{\|\boldsymbol{S_u}^{-1}\|}\int_0^t \left(e^{-\lambda_u \tau} - e^{-\underline{k}_{PF}\tau}\right)\mathrm{d}\tau + \kappa_{u_v}\frac{\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|}{\|\boldsymbol{S_u}^{-1}\|}t \\ &\leq \frac{\kappa_u}{\lambda_u}\frac{\|\boldsymbol{\zeta_{u_0}}\|}{\|\boldsymbol{S_u}^{-1}\|} + \frac{\kappa_{u_p}}{\underline{k}_{PF}\lambda_u}\frac{\|\boldsymbol{e_{p_0}}\|}{\|\boldsymbol{S_u}^{-1}\|} + \kappa_{u_v}\frac{\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|}{\|\boldsymbol{S_u}^{-1}\|}t, \\ \int_0^t \|\boldsymbol{u}_{\boldsymbol{\mathcal{T}}_{\models}}(\tau)\|\mathrm{d}\tau &\leq k_{e_p}\|\boldsymbol{e_{p_0}}\|\int_0^t e^{-\underline{k}_{PF}\tau}\mathrm{d}\tau + \frac{k_{e_p}}{\underline{k}_{PF}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|\int_0^t \left(1 - e^{-\bar{k}_{PF}\tau}\right)\mathrm{d}\tau \\ &\leq \frac{k_{e_p}}{\underline{k}_{PF}}\|\boldsymbol{e_{p_0}}\| + \frac{k_{e_p}}{\underline{k}_{PF}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|t.\end{aligned}$$

Combining these results yields the following bound for the collective temporal error:

$$|\zeta_t(t)| \leq |\zeta_{t_0}| + \kappa_{t_u}\|\boldsymbol{\zeta_{u_0}}\| + \kappa_{t_p}\|\boldsymbol{e_{p_0}}\| + \kappa_{t_v}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|t, \tag{C.48}$$

with known constants

$$\kappa_{t_u} := \frac{\kappa_u}{\lambda_u}\frac{\|\boldsymbol{R}\|}{\|\boldsymbol{S_u}^{-1}\|}, \qquad \kappa_{t_p} := \left(\frac{\kappa_{u_p}}{\underline{k}_{PF}\lambda_u}\frac{\|\boldsymbol{R}\|}{\|\boldsymbol{S_u}^{-1}\|} + \frac{\sqrt{n}}{n}\frac{k_{e_p}}{\underline{k}_{PF}}\right), \qquad \kappa_{t_v} := \left(\kappa_{u_v}\frac{\|\boldsymbol{R}\|}{\|\boldsymbol{S_u}^{-1}\|} + \frac{\sqrt{n}}{n}\frac{k_{e_p}}{\underline{k}_{PF}}\right).$$

The bound for the individual temporal error in this lemma follows from Equations (C.17), (C.46), and (C.48). □

C.5.3 Conditions on the Speed-Tracking Precision and Initial States

To ensure $\|\boldsymbol{v_{cmd,i}}(t)\| \leq v_{\max,i}$ for all $t \geq 0$ and all $i \in \mathcal{I}$ use Equations (2.10) and (2.11) to infer

$$\|\boldsymbol{v_{cmd,i}}(t)\| \leq |\dot{\xi}_i(t)|\|\boldsymbol{v_{d,i}}(\xi_i(t))\| + k_{PF,i}\|\boldsymbol{e_{p,i}}(t)\|.$$

Next, Lemma 1 and a similar argument as the one used to derive Equation (C.45) yield the steady-state bound

$$\lim_{t\to\infty}\|\boldsymbol{v_{cmd,i}}(t)\| \leq \rho\, v_{d_{\max},i} + \left(1 + \left(\kappa_{r,u_v} + \frac{k_{e_p}}{\underline{k}_{PF}}\right) v_{d_{\max},i}\right)\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|,$$

where $v_{d_{\max},i} := \sup_{\zeta_i\in[0,1]}\|\boldsymbol{v_{d,i}}(\zeta_i)\|$ is the maximum desired speed assigned by the trajectory generation algorithm to the ith vehicle. This implies that to guarantee $\lim_{t\to\infty}\|\boldsymbol{v_{cmd,i}}(t)\| < v_{\max,i}$ for all $i \in \mathcal{I}$ the collective speed-tracking precision must satisfy

$$\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| < \min_{i\in\mathcal{I}} \frac{v_{max,i} - \rho v_{d_{max},i}}{1 + \left(\kappa_{r,u_v} + \frac{k_{e_p}}{\underline{k}_{PF}}\right) v_{d_{\max},i}}. \tag{C.49}$$

Note that for this to be possible the reference agent and the trajectory generation algorithms must assign a maximum desired speed and mission rates that satisfy $v_{\max,i} - \rho\, v_{d_{\max},i} > 0$ for all $i \in \mathcal{I}$. This is considered in Assumption 2. Now, to derive a bound for $\|\boldsymbol{v_{cmd,i}}(t)\|$ that holds for both the

transient and steady-state responses, note that the term

$$\frac{1}{\underline{k}_{PF} - \lambda_u}\left(e^{-\lambda_u t} - e^{-\underline{k}_{PF} t}\right)$$

in Equation (C.45) reaches its maximum value ψ_u at time $t = \frac{1}{\underline{k}_{PF}-\lambda_u}\ln\left(\frac{\underline{k}_{PF}}{\lambda_u}\right)$ with

$$\psi_u := \frac{1}{\underline{k}_{PF} - \lambda_u}\left(\left(\frac{\underline{k}_{PF}}{\lambda_u}\right)^{-\frac{\lambda_u}{\underline{k}_{PF}-\lambda_u}} - \left(\frac{\underline{k}_{PF}}{\lambda_u}\right)^{-\frac{\underline{k}_{PF}}{\underline{k}_{PF}-\lambda_u}}\right). \tag{C.50}$$

The result above, Lemma 1, and a similar argument as in the derivation of Equation (C.45) yield

$$\|\boldsymbol{v_{cmd,i}}(t)\| \leq \rho v_{d_{\max},i} + \kappa_{\Omega_u,i}\|\boldsymbol{\zeta_{u_0}}\| + \kappa_{\Omega_{u_p},i}\|\boldsymbol{e_{p_0}}\| + \kappa_{\Omega_{u_v},i}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|, \qquad \forall t \geq 0.$$

This defines a set Ω_{u_0} where all $(\boldsymbol{\zeta_{u_0}}, \boldsymbol{e_{p_0}}) \in \Omega_{u_0}$ guarantee that $\|\boldsymbol{v_{cmd,i}}(t)\| \leq v_{\max,i}$ is met for all $t \geq 0$ and all $i \in \mathcal{I}$ with

$$\Omega_{u_0} := \left\{(\boldsymbol{\zeta_{u_0}}, \boldsymbol{e_{p_0}}) \in \mathbb{R}^{2n-n_\ell-1} \times \mathbb{R}^{3n} \,\middle|\, \kappa_{\Omega_u,i}\|\boldsymbol{\zeta_{u_0}}\| + \kappa_{\Omega_{u_p},i}\|\boldsymbol{e_{p_0}}\| \leq v_{max,i} - \rho v_{d_{max},i} - \kappa_{\Omega_{u_v},i}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|,\ \forall i \in \mathcal{I}\right\},$$

and constants

$$\kappa_{\Omega_u,i} := \kappa_{r_u} v_{d_{\max},i},$$
$$\kappa_{\Omega_{u_p},i} := k_{PF,i} + \left(\kappa_{r,u_p}\psi_u + k_{e_p}\right) v_{d_{\max},i},$$
$$\kappa_{\Omega_{u_v},i} := 1 + \left(\kappa_{r,u_v} + \frac{k_{e_p}}{\underline{k}_{PF}}\right) v_{d_{\max},i}.$$

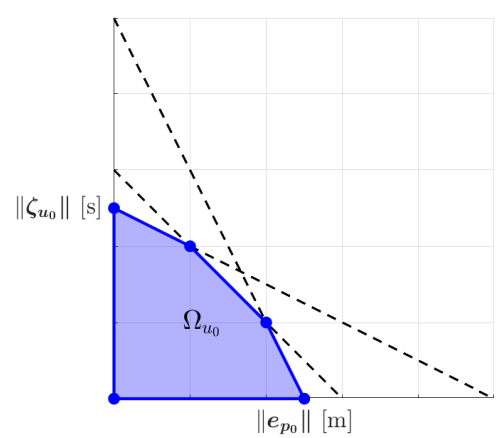


Figure C.4: Visualization of Ω_{u_0}.

Figure C.4 illustrates the geometry of Ω_{u_0}, highlighted in blue, for a group of 3 vehicles. Note that Ω_{u_0} is the intersection of the area below the 3 dahsed lines, one per vehicle. This set cannot be empty since $\kappa_{\Omega_u,i}, \kappa_{\Omega_{u_p},i} > 0$, and $v_{max,i} - \rho v_{d_{max},i} - \kappa_{\Omega_{u_v},i}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| > 0$ for all $i \in \mathcal{I}$. This last inequality can easily be proved using Assumption 2 and the inequality in (C.49).

Finally, choosing the same control gains as in Equation (C.11), the largest possible guaranteed rate of convergence $\lambda_u = \underline{\lambda}_u$, Lemmas 16 and 17 yield Theorem 4. ■

C.6 Proof of Theorem 5

This section builds upon the proof for Theorem 2, where relaxed temporal constraints with ideal target-tracking capabilities are addressed, and leverages the same change of variables, and Lyapunov function candidate as in Section C.3.

C.6.1 ISS with $\tilde{\omega}(t) \geq 1$

Using the dynamics in (5.7), the change of variables in Equation (C.19), and assuming $\tilde{\omega}(t) \geq 1$ yields the system dynamics

$$\dot{\boldsymbol{z}}(t) = \boldsymbol{A_z}(t)\boldsymbol{z}(t) + \boldsymbol{B_z}\boldsymbol{u}_{\mathcal{T}_\models}(t), \tag{C.51}$$

where $\boldsymbol{A_z}(t)$ is defined in (C.20), and $\boldsymbol{B_z} := \boldsymbol{SB}$. The following lemma uses the dynamics in (C.51) and the Lyapunov function in (C.25) to prove that the system with the target-tracking error feedback is ISS with respect to $\boldsymbol{u}_{\mathcal{T}_\models}(t)$. See [113] for the characterization of ISS-Lyapunov functions.

Lemma 18 *Assume the information flow $\mathcal{G}(t)$ satisfies Assumptions 3 through 5, and $\tilde{\omega}(t) \geq 1$. Then, there exists a function W_r for the system in* (C.51) *that satisfies*

$$\sqrt{a_r}\|\boldsymbol{z}\| \leq W_r \leq \sqrt{b_r}\|\boldsymbol{z}\|, \tag{C.52}$$

$$\dot{W}_r \leq -\lambda_r W_r + \frac{1}{\sqrt{a_r}}\|\boldsymbol{P_r}\boldsymbol{B_z}\|\|\boldsymbol{u}_{\mathcal{T}_\models}\|, \tag{C.53}$$

with known control gains k_R, k_P and k_I, constants a_r and b_r, and rate of convergence λ_r.

Proof. The time derivative of the Lyapunov function candidate V_r, defined in Equation (C.25), along the trajectories of (C.51) is

$$\dot{V}_r = \frac{\partial V_r}{\partial t} = \boldsymbol{z}^\top\left(\dot{\boldsymbol{P}}_{\boldsymbol{r}} + \boldsymbol{A}_{\boldsymbol{z}}^\top\boldsymbol{P_r} + \boldsymbol{P_r}\boldsymbol{A_z}\right)\boldsymbol{z} + \boldsymbol{u}_{\mathcal{T}_\models}^\top\boldsymbol{B}_{\boldsymbol{z}}^\top\boldsymbol{P_r}\boldsymbol{z} + \boldsymbol{z}^\top\boldsymbol{P_r}\boldsymbol{B_z}\boldsymbol{u}_{\mathcal{T}_\models},$$

and the proof of Lemma 9 leads to the following bound:

$$\dot{V}_r \leq -2\lambda_r V_r + 2\|\boldsymbol{P_r}\boldsymbol{B_z}\|\|\boldsymbol{u}_{\mathcal{T}_\models}\|\sqrt{\frac{V_r}{a_r}}. \tag{C.54}$$

To obtain a linear differential inequality, introduce the change of variables

$$W_r(t, \boldsymbol{z}) := \sqrt{V_r(t, \boldsymbol{z})},$$

and note that its time derivative is $\dot{W}_r = \frac{\dot{V}_r}{2\sqrt{V_r}}$. Similar to the proof in Section C.1, this yields Equation (C.53). Equation (C.52) follows from the definition of W_r and Equation (C.22). □

Application of the comparison lemma, Lemma 3.4 in [108], produces the following bound:

$$W_r(t) \leq W_r(0)e^{-\lambda_r t} + \frac{1}{\sqrt{a_r}}\|\boldsymbol{P_r}\boldsymbol{B_z}\| \int_0^t e^{-\lambda_r(t-\tau)}\|\boldsymbol{u}_{\boldsymbol{\mathcal{T}}_{\models}}(\tau)\| \, \mathrm{d}\tau,$$

and a similar reasoning as in Section C.5.1 yields

$$W_r(t) \leq W_r(0)e^{-\lambda_r t} + \kappa_{W_r}\left(\frac{\|\boldsymbol{e_{p_0}}\|}{\underline{k}_{PF} - \lambda_r}\left(e^{-\lambda_r t} - e^{-\underline{k}_{PF} t}\right) + \frac{\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|}{\underline{k}_{PF}\lambda_r}\right), \tag{C.55}$$

where $\kappa_{W_r} := k_{e_p}\frac{b_r}{\sqrt{a_r}}\|\boldsymbol{P_r}\boldsymbol{B_z}\|$. The next lemma leverages Equation (C.55) to analyze the effect on the individual temporal, coordination and rate errors of a perturbation with the structure detailed in Equation (6.1).

Lemma 19 *Assume the underlying speed-tracking controller for all agents satisfies Assumption 1, the information flow $\mathcal{G}(t)$ satisfies Assumptions 3 through 5, and $\tilde{\omega}(t) \geq 1$. Then, the origin of the system dynamics in* (C.51) *is ISS with respect to $\boldsymbol{e_v}$, and satisfies*

$$|\xi_i(t) - \xi_R(t)| \leq \kappa_{t,r}\|\boldsymbol{\zeta_0}\|e^{-\lambda_r t} + \frac{\kappa_{t,r_p}}{\underline{k}_{PF} - \lambda_r}\|\boldsymbol{e_{p_0}}\|\left(e^{-\lambda_r t} - e^{-\underline{k}_{PF} t}\right) + \kappa_{t,r_v}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|, \tag{C.56}$$

$$|\xi_i(t) - \xi_j(t)| \leq \kappa_{c,r}\|\boldsymbol{\zeta_0}\|e^{-\lambda_r t} + \frac{\kappa_{c,r_p}}{\underline{k}_{PF} - \lambda_r}\|\boldsymbol{e_{p_0}}\|\left(e^{-\lambda_r t} - e^{-\underline{k}_{PF} t}\right) + \kappa_{c,r_v}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|, \tag{C.57}$$

$$\begin{aligned}|\dot{\xi}_i(t) - \rho| \leq \kappa_{r,r}\|\boldsymbol{\zeta_0}\|e^{-\lambda_r t} + \frac{\kappa_{r,r_p}}{\underline{k}_{PF} - \lambda_r}\|\boldsymbol{e_{p_0}}\|\left(e^{-\lambda_r t} - e^{-\underline{k}_{PF} t}\right) + \kappa_{r,r_v}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| + \dots \\ + k_{e_p}\|\boldsymbol{e_{p_0}}\|e^{-\underline{k}_{PF} t} + \frac{k_{e_p}}{\underline{k}_{PF}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|,\end{aligned} \tag{C.58}$$

for all $t \geq 0$, and all i, $j \in \mathcal{I}$, with constants

$$\begin{aligned}
\kappa_{t,r} &:= \left(1+\sqrt{\tfrac{n-1}{n}}\right)\kappa_r, & \kappa_{c,r} &:= \sqrt{2}\kappa_r, & \kappa_{r,r} &:= \left(k_P n + \max\left\{1, k_R\left(1+\sqrt{\tfrac{n-1}{n}}\right)\right\}\right)\kappa_r,\\
\kappa_{t,r_p} &:= \left(1+\sqrt{\tfrac{n-1}{n}}\right)\kappa_{r_p}, & \kappa_{c,r_p} &:= \sqrt{2}\kappa_{r_p}, & \kappa_{r,r_p} &:= \left(k_P n + \max\left\{1, k_R\left(1+\sqrt{\tfrac{n-1}{n}}\right)\right\}\right)\kappa_{r_p},\\
\kappa_{t,r_v} &:= \left(1+\sqrt{\tfrac{n-1}{n}}\right)\kappa_{r_v}, & \kappa_{c,r_v} &:= \sqrt{2}\kappa_{r_v}, & \kappa_{r,r_v} &:= \left(k_P n + \max\left\{1, k_R\left(1+\sqrt{\tfrac{n-1}{n}}\right)\right\}\right)\kappa_{r_v}.
\end{aligned}$$

Proof. Equation (C.52) and the change of variables in Equation (C.19) imply

$$\|\boldsymbol{\zeta}(t)\| \leq \kappa_r \|\boldsymbol{\zeta_0}\| e^{-\lambda_r t} + \frac{\kappa_{r_p}}{\underline{k}_{PF} - \lambda_r}\|\boldsymbol{e_{p_0}}\|\left(e^{-\lambda_r t} - e^{-\underline{k}_{PF} t}\right) + \kappa_{r_v}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|, \tag{C.59}$$

where κ_r is defined in Lemma 10, and $\kappa_{r,p}$ and $\kappa_{r,v}$ are

$$\kappa_{r_p} := k_{e_p}\frac{b_r}{a_r}\|\boldsymbol{S}^{-1}\|\|\boldsymbol{B_z}\|, \qquad \kappa_{r_v} := \frac{1}{\lambda_r}\frac{k_{e_p}}{\underline{k}_{PF}}\frac{b_r}{a_r}\|\boldsymbol{S}^{-1}\|\|\boldsymbol{B_z}\|.$$

A similar argument as in the proof of Lemma 10 yields the individual temporal, coordination and rate errors in (C.56), (C.57) and (C.58). □

This concludes the proof of Lemma 3.

Conditions on the Speed-Tracking Precision and Initial States, $\tilde{\omega} \geq 1$

A similar procedure as in Section C.5.3 can be used, along with derivation of Equation (C.58), to infer the following condition on the collective speed-tracking precision:

$$\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| < \min_{i\in\mathcal{I}} \frac{v_{max,i} - \rho v_{d_{max},i}}{1 + \left(\kappa_{r,r_v} + \frac{k_{e_p}}{\underline{k}_{PF}}\right) v_{d_{\max},i}}, \tag{C.60}$$

which guarantees that the steady-state response satisfies $\lim_{t\to\infty}\|\boldsymbol{v_{cmd,i}}(t)\| < v_{\max,i}$ for all $i \in \mathcal{I}$. Note that for this to be possible the reference agent and the trajectory generation algorithms must assign a maximum desired speed and mission rates that satisfy $v_{\max,i} - \rho\, v_{d_{\max},i} > 0$ for all $i \in \mathcal{I}$. This is considered in Assumption 2. Now, similar to Section C.5.3, to derive a bound for $\|\boldsymbol{v_{cmd,i}}(t)\|$

that holds for both the transient and steady-state responses, note that the term

$$\frac{1}{\underline{k}_{PF} - \lambda_r}\left(e^{-\lambda_r t} - e^{-\underline{k}_{PF} t}\right)$$

in Equation (C.58) reaches its maximum value ψ_r at time $t = \frac{1}{\underline{k}_{PF}-\lambda_r}\ln\left(\frac{\underline{k}_{PF}}{\lambda_r}\right)$ with

$$\psi_r := \frac{1}{\underline{k}_{PF} - \lambda_r}\left(\left(\frac{\underline{k}_{PF}}{\lambda_r}\right)^{-\frac{\lambda_r}{\underline{k}_{PF}-\lambda_r}} - \left(\frac{\underline{k}_{PF}}{\lambda_r}\right)^{-\frac{\underline{k}_{PF}}{\underline{k}_{PF}-\lambda_r}}\right). \tag{C.61}$$

The result above, Lemma 1, and a similar argument as in the derivation of Equation (C.58) yield

$$\|\boldsymbol{v_{cmd,i}}(t)\| \leq \rho v_{d_{\max},i} + \kappa_{\Omega_r,i}\|\boldsymbol{\zeta_0}\| + \kappa_{\Omega_{r_p},i}\|\boldsymbol{e_{p_0}}\| + \kappa_{\Omega_{r_v},i}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|.$$

This defines a set Ω_{r_0} where all $(\boldsymbol{\zeta_0}, \boldsymbol{e_{p_0}}) \in \Omega_{r_0}$ guarantee that $\|\boldsymbol{v_{cmd,i}}(t)\| \leq v_{\max,i}$ is met for all $t \geq 0$ and all $i \in \mathcal{I}$ with

$$\Omega_{r_0} := \left\{(\boldsymbol{\zeta_0}, \boldsymbol{e_{p_0}}) \in \mathbb{R}^{2n-n_\ell-1} \times \mathbb{R}^{3n} \,\middle|\, \kappa_{\Omega_r,i}\|\boldsymbol{\zeta_0}\| + \kappa_{\Omega_{r_p},i}\|\boldsymbol{e_{p_0}}\| \leq v_{max,i} - \rho v_{d_{max},i} - \kappa_{\Omega_{r_v},i}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|,\ \forall i \in \mathcal{I}\right\},$$

and constants

$$\kappa_{\Omega_r,i} := \kappa_{r,r} v_{d_{\max},i},$$
$$\kappa_{\Omega_{r_p},i} := k_{PF,i} + \left(\kappa_{r,r_p}\psi_r + k_{e_p}\right) v_{d_{\max},i},$$
$$\kappa_{\Omega_{r_v},i} := 1 + \left(\kappa_{r,r_v} + \frac{k_{e_p}}{\underline{k}_{PF}}\right) v_{d_{\max},i}.$$

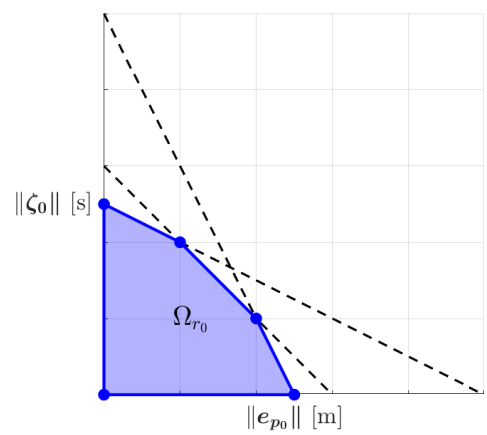


Figure C.5: Visualization of Ω_{r_0}.

Figure C.5 illustrates the geometry of Ω_{r_0}, highlighted in blue, for a group of 3 vehicles. The same interpretation of the geometry of the set as in Section C.5.3, Assumption 2, and the inequality in (C.60) lead to the same conclusion here: the set Ω_{r_0} cannot be empty. In addition, to ensure that vehicles can converge to the desired temporal window $\Delta_t(t)$ as $t \to \infty$ when $\tilde{\omega} \geq 1$,

Equation (C.56) can be used to derive the following condition on the speed-tracking precision:

$$\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| < \frac{\underline{\Delta}_t}{\kappa_{t,r_v}},$$

where $0 < \underline{\Delta}_t \leq \Delta_t(t)$ for all $t \geq 0$.

C.6.2 Dwell Time

The dynamics in (6.3) and the change of variables in (C.19) yield the following switched system dynamics:

$$\boldsymbol{z}(t) = \boldsymbol{A}_{\boldsymbol{z}_\gamma}(t)\boldsymbol{z}(t) + \boldsymbol{B}_{\boldsymbol{z}}\boldsymbol{u}_{\mathcal{T}_{\models}}(t), \qquad \gamma \in \mathcal{S}, \tag{C.62}$$

with the same switching signal and switching times as in Section 6.2.2, $\boldsymbol{A}_{\boldsymbol{z}_\emptyset}(t) := \boldsymbol{S}\boldsymbol{A}_\emptyset(t)\boldsymbol{S}^{-1}$, $\boldsymbol{A}_{\boldsymbol{z}_{\text{☁}}}(t) := \boldsymbol{S}\boldsymbol{A}_{\text{☁}}(t)\boldsymbol{S}^{-1}$, and $\boldsymbol{B}_{\boldsymbol{z}} := \boldsymbol{S}\boldsymbol{B}$. Contrary to the analysis in Section C.3.2, now in mode $\emptyset$ the collective temporal error $\zeta_t(t)$ can drift away from the origin, as shown in Equation (C.48), and the coordination and temporal errors embedded in $\boldsymbol{z}_{\boldsymbol{u}}(t)$ are ISS with respect to $\boldsymbol{u}_{\mathcal{T}_{\models}}(t)$. In addition, in mode ☁ the state that lumps together the temporal, coordination, and rate errors $\boldsymbol{z}(t)$ is also ISS with respect to $\boldsymbol{u}_{\mathcal{T}_{\models}}(t)$. To study this new behavior, this section will utilize functions W_u and W_r, as opposed to V_u and V_r used in Section C.3.2. The reason for this is that the linearity initially shown in (C.9) and (C.23) for ideal target tracking has now been lost, as demonstrated by Equations (C.42) and (C.54).

Next, assuming that (C.62) persistently switches between modes $\emptyset$ and ☁ with the slow switching constraints in (6.5), the following lemma uses switched theory for ISS and iISS systems with the proof of Lemmas 16, 17 and 19 to find individual dwell times $\tau_{R_{1,k}}$ that render the system dynamics in (C.62) ISS with respect to $\boldsymbol{u}_{\mathcal{T}_{\models}}(t)$.

Lemma 20 *Assume the underlying speed-tracking controller for all agents satisfies Assumption 1, the information flow $\mathcal{G}(t)$ satisfies assumptions 3 through 5. Then, the switched system* (C.62)*, with slow switching constraints* (6.5)*, and individual dwell times*

$$\tau^i_{R_1}(t) = \max\left\{\epsilon_\tau, \frac{1}{\lambda_r}\ln\Delta t^i_0(t)\right\} + \max\left\{0, \frac{1}{\lambda_r}\ln\kappa_{W_r}\right\}, \qquad \forall i \in \mathcal{I}_\ell,$$

with known constant κ_{W_r}, and design parameter $\epsilon_\tau > 0$, satisfies the following temporal, coordination, and rate errors bounds:

$$|\xi_i(t_{2k}) - \xi_R(t_{2k})| \leq \kappa_{t,r}\|\boldsymbol{\zeta_0}\|e^{-k\lambda_r\epsilon_\tau} + \tilde{\kappa}_{t,p}\|\boldsymbol{e_{p_0}}\|\, k\, e^{-k\tilde{\lambda}\epsilon_\tau} + \left(\tilde{\kappa}_{t,v_1} + \tilde{\kappa}_{t,v_2}e^{\lambda_r\epsilon_\tau}\right)\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \sum_{m=1}^{k} e^{-m\lambda_r\epsilon_\tau}, \quad \text{(C.63)}$$

$$|\xi_i(t_{2k}) - \xi_j(t_{2k})| \leq \kappa_{c,r}\|\boldsymbol{\zeta_0}\|e^{-k\lambda_r\epsilon_\tau} + \tilde{\kappa}_{c,p}\|\boldsymbol{e_{p_0}}\|\, k\, e^{-k\tilde{\lambda}\epsilon_\tau} + \left(\tilde{\kappa}_{c,v_1} + \tilde{\kappa}_{c,v_2}e^{\lambda_r\epsilon_\tau}\right)\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \sum_{m=1}^{k} e^{-m\lambda_r\epsilon_\tau}, \quad \text{(C.64)}$$

$$|\dot{\xi}_i(t_{2k}) - \rho| \leq \kappa_{r,r}\|\boldsymbol{\zeta_0}\|e^{-k\lambda_r\epsilon_\tau} + \tilde{\kappa}_{r,p}\|\boldsymbol{e_{p_0}}\|\, k\, e^{-k\tilde{\lambda}\epsilon_\tau} + \left(\tilde{\kappa}_{r,v_1} + \tilde{\kappa}_{r,v_2}e^{\lambda_r\epsilon_\tau}\right)\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \sum_{m=1}^{k} e^{-m\lambda_r\epsilon_\tau} + \dots$$
$$+ k_{e_p}\|\boldsymbol{e_{p_0}}\|e^{-k\,\underline{k}_{PF}\,\epsilon_\tau} + \frac{k_{e_p}}{\underline{k}_{PF}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|, \quad \text{(C.65)}$$

for all $i, j \in \mathcal{I}$ and $k \in \mathbb{N}$, with $\kappa_{t,r}$, $\kappa_{c,r}$, and $\kappa_{r,r}$ defined in Lemma 10, and remaining constants

$$\tilde{\kappa}_{t,p} := \left(1 + \sqrt{\tfrac{n-1}{n}}\right)\tfrac{\tilde{\kappa}_{W_p}}{\sqrt{a_r}}\|\boldsymbol{S}^{-1}\|, \quad \tilde{\kappa}_{c,p} := \sqrt{2}\tfrac{\tilde{\kappa}_{W_p}}{\sqrt{a_r}}\|\boldsymbol{S}^{-1}\|, \quad \tilde{\kappa}_{r,p} := \left(k_P n + \max\left\{1, k_R\left(1+\sqrt{\tfrac{n-1}{n}}\right)\right\}\right)\tfrac{\tilde{\kappa}_{W_p}}{\sqrt{a_r}}\|\boldsymbol{S}^{-1}\|,$$
$$\tilde{\kappa}_{t,v_1} := \left(1 + \sqrt{\tfrac{n-1}{n}}\right)\tfrac{\tilde{\kappa}_{W_{v_1}}}{\sqrt{a_r}}\|\boldsymbol{S}^{-1}\|, \quad \tilde{\kappa}_{c,v_1} := \sqrt{2}\tfrac{\tilde{\kappa}_{W_{v_1}}}{\sqrt{a_r}}\|\boldsymbol{S}^{-1}\|, \quad \tilde{\kappa}_{r,v_1} := \left(k_P n + \max\left\{1, k_R\left(1+\sqrt{\tfrac{n-1}{n}}\right)\right\}\right)\tfrac{\tilde{\kappa}_{W_{v_1}}}{\sqrt{a_r}}\|\boldsymbol{S}^{-1}\|,$$
$$\tilde{\kappa}_{t,v_2} := \left(1 + \sqrt{\tfrac{n-1}{n}}\right)\tfrac{\tilde{\kappa}_{W_{v_2}}}{\sqrt{a_r}}\|\boldsymbol{S}^{-1}\|, \quad \tilde{\kappa}_{c,v_2} := \sqrt{2}\tfrac{\tilde{\kappa}_{W_{v_2}}}{\sqrt{a_r}}\|\boldsymbol{S}^{-1}\|, \quad \tilde{\kappa}_{r,v_2} := \left(k_P n + \max\left\{1, k_R\left(1+\sqrt{\tfrac{n-1}{n}}\right)\right\}\right)\tfrac{\tilde{\kappa}_{W_{v_2}}}{\sqrt{a_r}}\|\boldsymbol{S}^{-1}\|.$$

Proof. Using Equations (C.55), (C.61), and (6.5), and leveraging Figure C.6 to aid in the visualization of this proof, the evolution of W_r in mode ☁ from t_{2k-1} to t_{2k} satisfies

$$W_r(t_{2k}) \leq W_r(t_{2k-1})e^{-\lambda_r\tau_{R_{1,k}}} + \kappa_{W_r}\left(\psi_r\|\boldsymbol{e_p}(t_{2k-1})\| + \frac{1}{\underline{k}_{PF}\lambda_r}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|\right). \quad \text{(C.66)}$$

Notice that in Figure C.6 the functions W_u and W_r can initially grow until the exponentially decaying terms dominate. However, in the previous analysis of the switched system with ideal target-tracking capabilities, these functions were monotonically decreasing, as depicted in Figure C.1. Next, the definition of W_r, inequality (C.22), and the change of variables in (C.30), along

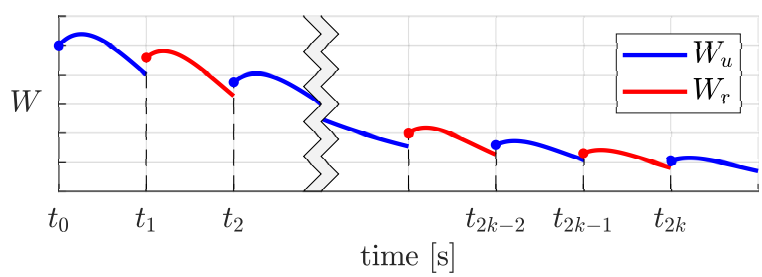


Figure C.6: Evolution of functions W_u and W_r.

with $\|\tilde{\boldsymbol{z}}(t_{2k-1})\| \leq |\zeta_t(t_{2k-1})| + \|\boldsymbol{z_u}(t_{2k-1})\|$ imply

$$W_r(t_{2k-1}) \leq \sqrt{b_r}\|\boldsymbol{T_z}\| \left(|\zeta_t(t_{2k-1})| + \|\boldsymbol{z_u}(t_{2k-1})\|\right). \tag{C.67}$$

Now, to find the evolution of $\zeta_t(t)$ in mode Ø from t_{2k-2} to t_{2k-1}, leverage the preceding steps to Equation (C.48) to arrive at

$$|\zeta_t(t_{2k-1})| \leq |\zeta_t(t_{2k-2})| + \kappa_{t_z}\|\boldsymbol{z_u}(t_{2k-2})\| + \kappa_{t_p}\|\boldsymbol{e_p}(t_{2k-2})\| + \kappa_{t_v}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|(t_{2k-1} - t_{2k-2}),$$

where $\kappa_{t_z} := \frac{\kappa_{t_u}}{\|\tilde{\boldsymbol{S}}_{\boldsymbol{u}}\|}$, and κ_{t_u}, κ_{t_p}, and κ_{t_v} are defined in Section C.5.2. The evolution of $\|\boldsymbol{z_u}(t)\|$ in mode Ø from t_{2k-2} to t_{2k-1} can be bounded using the definition of W_u, Equations (C.8), (C.43), (C.50), and (6.5). For the sake of simplicity, and at the expense of less tight bounds, all exponentially decaying terms have been bounded by a constant value, leading to

$$\|\boldsymbol{z_u}(t_{2k-1})\| \leq \sqrt{\frac{b_u}{a_u}}\|\boldsymbol{z_u}(t_{2k-2})\| + \frac{\kappa_{W_u}}{\sqrt{a_u}}\left(\psi_u\|\boldsymbol{e_p}(t_{2k-2})\| + \frac{1}{\underline{k}_{PF}\lambda_u}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|\right).$$

Hence, combining the bounds for $|\zeta_t(t_{2k-1})|$ and $\|\boldsymbol{z_u}(t_{2k-1})\|$ with Equation (C.67), noting that $\|\boldsymbol{z_u}(t_{2k-2})\| \leq \|\tilde{\boldsymbol{z}}(t_{2k-2})\| \leq \|\boldsymbol{T_z}^{-1}\|\|\boldsymbol{z}(t_{2k-2})\|$ and $|\zeta_t(t_{2k-2})| \leq \|\boldsymbol{z}(t_{2k-2})\|$, the definition of W_r, and Equation (C.22) yield

$$W_r(t_{2k-1}) \leq \kappa_{W_r}W_r(t_{2k-2}) + \kappa_{W_p}\|\boldsymbol{e_p}(t_{2k-2})\| + \kappa_{W_{v_1}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| + \kappa_{W_{v_2}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|\left(t_{2k-1} - t_{2k-2}\right), \tag{C.68}$$

with constants

$$\kappa_{W_r} := b_r\|\boldsymbol{T_z}\|\left(1 + \|\boldsymbol{T_z}^{-1}\|\left(\kappa_{t_z} + \sqrt{\frac{b_u}{a_u}}\right)\right), \qquad \kappa_{W_p} := \sqrt{b_r}\|\boldsymbol{T_z}\|\left(\kappa_{t_p} + \frac{\kappa_{W_u}}{\sqrt{a_u}}\psi_u\right),$$
$$\kappa_{W_{v_1}} := \sqrt{b_r}\|\boldsymbol{T_z}\|\frac{\kappa_{W_u}}{\sqrt{a_u}\,\underline{k}_{PF}\lambda_u}, \qquad \kappa_{W_{v_2}} := \sqrt{b_r}\|\boldsymbol{T_z}\|\kappa_{t_v}.$$

Define the time spend in mode Ø for the kth Ø-☁ cycle as $\Delta t_{u,k} := t_{2k-1} - t_{2k-2}$, and combine

Equations (C.68) and (C.68) to obtain

$$W_r(t_{2k}) \leq \left(\kappa_{W_r} W_r(t_{2k-2}) + \kappa_{W_p}\|\boldsymbol{e_p}(t_{2k-2})\| + \kappa_{W_{v_1}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| + \kappa_{W_{v_2}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|\Delta t_{u,k}\right) e^{-\lambda_r \tau_{R_{1,k}}} + \dots$$
$$+ \kappa_{W_r}\left(\psi_r\|\boldsymbol{e_p}(t_{2k-1})\| + \frac{1}{\underline{k}_{PF}\lambda_r}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|\right)$$

Then, an inductive argument leads to

$$W_r(t_{2k}) \leq s_{0,k} + s_{1,k} + s_{2,k} + s_{3,k} + s_{4,k} + s_{5,k},$$

where $s_{0,k}$ is defined by a single summand, and $s_{1,k}$ through $s_{5,k}$ are the following partial sums[5]:

$$s_{0,k} := W_r(t_0)\,\kappa_{W_r}^k\, e^{-\lambda_r \sum_{j=1}^{k} \tau_{R_{1,j}}},$$
$$s_{1,k} := \kappa_{W_p} \sum_{i=1}^{k} \|\boldsymbol{e_p}\left(t_{2(k-i)}\right)\|\, \kappa_{W_r}^{i-1}\, e^{-\lambda_r \sum_{j=1}^{i} \tau_{R_{1,k-j+1}}},$$
$$s_{2,k} := \kappa_{W_{v_1}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \sum_{i=1}^{k} \kappa_{W_r}^{i-1}\, e^{-\lambda_r \sum_{j=1}^{i} \tau_{R_{1,k-j+1}}},$$
$$s_{3,k} := \kappa_{W_{v_2}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \sum_{i=1}^{k} \Delta t_{u,k-i+1}\, \kappa_{W_r}^{i-1}\, e^{-\lambda_r \sum_{j=1}^{i} \tau_{R_{1,k-j+1}}},$$
$$s_{4,k} := \kappa_{W_r}\psi_r \sum_{i=1}^{k} \kappa_{W_r}^{i-1}\, \|\boldsymbol{e_p}(t_{2(k-i)+1})\|\, e^{-\lambda_r \sum_{j=1}^{i-1} \tau_{R_{1,k-j+1}}},$$
$$s_{5,k} := \frac{\kappa_{W_r}}{\underline{k}_{PF}\lambda_r}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \sum_{i=1}^{k} \kappa_{W_r}^{i-1}\, e^{-\lambda_r \sum_{j=1}^{i-1} \tau_{R_{1,k-j+1}}}.$$

To ensure that $s_{0,k} \to 0$ as $k \to \infty$, and $s_{1,k}$ through $s_{5,k}$ are convergent series, choose a dwell time

$$\tau_{R_{1,j}} \geq \eta_{\tau,j} + \max\left\{0, \frac{1}{\lambda_r}\ln \kappa_{W_r}\right\}, \quad \text{with} \quad \eta_{\tau,j} = \max\left\{\epsilon_\tau, \frac{1}{\lambda_r}\ln \Delta t_{u,j}\right\}, \tag{C.69}$$

and $\epsilon_\tau > 0$. This choice of a dwell time has two distinct terms with clear objectives:

i) The first term forces the system to stay in mode enough time to compensate for the linear drift in the collective temporal error $\zeta_t(t)$ that may have happened in mode $\emptyset$, which

[5]To correctly interpret the expressions for $s_{4,k}$ and $s_{5,k}$, the operator $\sum_{j=a}^{b} f(j)$ is defined as the sum of all values of $f(j)$ such that $a \leq j \leq b$. Hence, when $a > b$ the set of j values that meet the aforementioned condition is empty, and the operator returns $\sum_{j=a}^{b} f(j) = 0$. Accordingly, when $i = 1$ then $\sum_{j=1}^{i-1} \tau_{R_{1,k-j+1}} = 0$.

is represented by $\Delta t_{u,k-i+1}$ in $s_{3,k}$.

ii) The second term aims to cancel out the geometric growth that κ_{W_r} introduces in $s_{0,k}$ through $s_{5,k}$ due to the switching between modes Ø and ☁.

Now, leveraging Lemma 1, noting that at $t_{2(k-i)}$ and $t_{2(k-i)+1}$ the system has fully completed $k-i$ Ø-☁ cycles, and the duration of each Ø-☁ cycle is greater or equal than the corresponding dwell time, yields

$$\|\boldsymbol{e_p}(t_{2(k-i)})\| \leq \|\boldsymbol{e_p}(t_0)\| e^{-\underline{k}_{PF}(t_{2(k-i)}-t_0)} + \frac{\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|}{\underline{k}_{PF}} \leq \|\boldsymbol{e_p}(t_0)\| e^{-\underline{k}_{PF}\sum_{j=1}^{k-i}\tau_{R_{1,j}}} + \frac{\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|}{\underline{k}_{PF}},$$
$$\|\boldsymbol{e_p}(t_{2(k-i)+1})\| \leq \|\boldsymbol{e_p}(t_0)\| e^{-\underline{k}_{PF}(t_{2(k-i)+1}-t_0)} + \frac{\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|}{\underline{k}_{PF}} \leq \|\boldsymbol{e_p}(t_0)\| e^{-\underline{k}_{PF}\sum_{j=1}^{k-i}\tau_{R_{1,j}}} + \frac{\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|}{\underline{k}_{PF}},$$

The expressions for $s_{0,k}$ through $s_{5,k}$, the choice of $\tau_{R_{1,j}}$ in (C.69), and the result above imply

$$\begin{aligned}
s_{0,k} &\leq W_r(0) e^{-\lambda_r \sum_{j=1}^{k}\eta_{\tau,j}},\\
s_{1,k} &\leq \frac{\kappa_{W_p}}{\kappa_{W_r}}\|\boldsymbol{e_{p_0}}\| \sum_{i=1}^{k} e^{-\underline{k}_{PF}\sum_{j=1}^{k-i}\tau_{R_{1,j}} - \lambda_r \sum_{j=1}^{i}\eta_{\tau,k-j+1}} + \frac{\kappa_{W_p}}{\kappa_{W_r}}\frac{\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|}{\underline{k}_{PF}} \sum_{i=1}^{k} e^{-\lambda_r \sum_{j=1}^{i}\eta_{\tau,k-j+1}},\\
s_{2,k} &\leq \frac{\kappa_{W_{v_1}}}{\kappa_{W_r}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \sum_{i=1}^{k} e^{-\lambda_r \sum_{j=1}^{i}\eta_{\tau,k-j+1}},\\
s_{3,k} &\leq \frac{\kappa_{W_{v_2}}}{\kappa_{W_r}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \sum_{i=1}^{k} e^{-\lambda_r \sum_{j=1}^{i-1}\eta_{\tau,k-j+1}},\\
s_{4,k} &\leq \kappa_{W_r}\psi_r\|\boldsymbol{e_{p_0}}\| \sum_{i=1}^{k} e^{-\underline{k}_{PF}\sum_{j=1}^{k-i}\tau_{R_{1,j}} - \lambda_r \sum_{j=1}^{i-1}\eta_{\tau,k-j+1}} + \kappa_{W_r}\psi_r\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \sum_{i=1}^{k} e^{-\lambda_r \sum_{j=1}^{i-1}\eta_{\tau,k-j+1}},\\
s_{5,k} &\leq \frac{\kappa_{W_r}}{\underline{k}_{PF}\lambda_r}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \sum_{i=1}^{k} e^{-\lambda_r \sum_{j=1}^{i-1}\eta_{\tau,k-j+1}}.
\end{aligned}$$

To ease the interpretation of $s_{1,k}$ and $s_{4,k}$, the summands in the first partial sum of $s_{1,k}$ and $s_{4,k}$ can be bounded as follows:

$$\begin{aligned}
e^{-\underline{k}_{PF}\sum_{j=1}^{k-i}\tau_{R_{1,j}} - \lambda_r \sum_{j=1}^{i}\eta_{\tau,k-j+1}} &\leq e^{-\min\{\underline{k}_{PF},\lambda_r\}\left(\sum_{j=1}^{k-i}\eta_{\tau,j} + \sum_{j=1}^{i}\eta_{\tau,k-j+1}\right)},\\
e^{-\underline{k}_{PF}\sum_{j=1}^{k-i}\tau_{R_{1,j}} - \lambda_r \sum_{j=1}^{i-1}\eta_{\tau,k-j+1}} &\leq e^{-\min\{\underline{k}_{PF},\lambda_r\}\left(\sum_{j=1}^{k-i}\eta_{\tau,j} + \sum_{j=1}^{i-1}\eta_{\tau,k-j+1}\right)}.
\end{aligned}$$

Note also that $\eta_{\tau,j} \geq \epsilon_\tau$, and thus

$$\sum_{j=1}^{k-i} \eta_{\tau,j} + \sum_{j=1}^{i} \eta_{\tau,k-j+1} = \eta_{\tau,1} + \ldots + \eta_{\tau,k-i-1} + \eta_{\tau,k-i} + \eta_{\tau,k-i+1} + \cdots + \eta_{\tau,k} \leq k\epsilon_\tau,$$
$$\sum_{j=1}^{k-i} \eta_{\tau,j} + \sum_{j=1}^{i-1} \eta_{\tau,k-j+1} = \eta_{\tau,1} + \ldots + \eta_{\tau,k-i-1} + 2\eta_{\tau,k-i} + \eta_{\tau,k-i+1} + \cdots + \eta_{\tau,k} \leq (k+1)\epsilon_\tau,$$

and use these results to further simplify the bounds for $s_{0,k}$ through $s_{5,k}$, which yields

$$\begin{aligned}
s_{0,k} &\leq W_r(0)e^{-k\lambda_r\epsilon_\tau}, \\
s_{1,k} &\leq \frac{\kappa_{W_p}}{\kappa_{W_r}}\|\boldsymbol{e_{p_0}}\| \sum_{i=1}^{k} e^{-k\min\{\underline{k}_{PF},\lambda_r\}\epsilon_\tau} + \frac{\kappa_{W_p}}{\kappa_{W_r}}\frac{\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|}{\underline{k}_{PF}} \sum_{i=1}^{k} e^{-i\lambda_r\epsilon_\tau}, \\
s_{2,k} &\leq \frac{\kappa_{W_{v_1}}}{\kappa_{W_r}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \sum_{i=1}^{k} e^{-i\lambda_r\epsilon_\tau}, \\
s_{3,k} &\leq \frac{\kappa_{W_{v_2}}}{\kappa_{W_r}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \sum_{i=1}^{k} e^{-(i-1)\lambda_r\epsilon_\tau}, \\
s_{4,k} &\leq \kappa_{W_r}\psi_r\|\boldsymbol{e_{p_0}}\| \sum_{i=1}^{k} e^{-(k+1)\min\{k_{PF},\lambda_r\}\epsilon_\tau} + \kappa_{W_r}\psi_r\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \sum_{i=1}^{k} e^{-(i-1)\lambda_r\epsilon_\tau}, \\
s_{5,k} &\leq \frac{\kappa_{W_r}}{\underline{k}_{PF}\lambda_r}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \sum_{i=1}^{k} e^{-(i-1)\lambda_r\epsilon_\tau}.
\end{aligned}$$

The following conclusions follow:

i) The sequence $s_{0,1}, s_{0,2}, \ldots, s_{0,k}$ is positive, decreasing, and converges to the origin as $k \to \infty$.

ii) D'Alembert's criterion, also known as the ratio test, can be used to prove that the series $s_{1,k}$ through $s_{5,k}$ are convergent, see Theorem 24.2.4 in [114], and the corresponding infinite sums are bounded by:

$$\begin{aligned}
\lim_{k\to\infty} s_{1,k} &\leq \frac{\kappa_{W_p}}{\kappa_{W_r}\underline{k}_{PF}}\frac{1}{e^{\lambda_r\epsilon_\tau}-1}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|, \\
\lim_{k\to\infty} s_{2,k} &\leq \frac{\kappa_{W_{v_1}}}{\kappa_{W_r}}\frac{1}{e^{\lambda_r\epsilon_\tau}-1}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|, \\
\lim_{k\to\infty} s_{3,k} &\leq \frac{\kappa_{W_{v_2}}}{\kappa_{W_r}}\frac{e^{\lambda_r\epsilon_\tau}}{e^{\lambda_r\epsilon_\tau}-1}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|, \\
\lim_{k\to\infty} s_{4,k} &\leq \kappa_{W_r}\psi_r\frac{e^{\lambda_r\epsilon_\tau}}{e^{\lambda_r\epsilon_\tau}-1}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|,
\end{aligned}$$

$$\lim_{k\to\infty} s_{5,k} \leq \frac{\kappa_{W_r}}{\underline{k}_{PF}\lambda_r}\frac{e^{\lambda_r\epsilon_\tau}}{e^{\lambda_r\epsilon_\tau}-1}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|.$$

The results above imply that the switching system in Equation (C.62) is able to cancel out the effects of the initial position error $\|\boldsymbol{e}_{\boldsymbol{p_0}}\|$ as the number of Ø-☁ cycles increases. However, there is an error induced by the inability to exactly track the desired speed command that is propagated through the system dynamics and yields

$$\lim_{k\to\infty} W_r(t_{2k}) \leq \kappa_\infty\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|,$$

with

$$\kappa_\infty := \left(\frac{\kappa_{W_p}}{\kappa_{W_r}\underline{k}_{PF}} + \frac{\kappa_{W_{v_1}}}{\kappa_{W_r}}\right)\frac{1}{e^{\lambda_r\epsilon_\tau}-1} + \left(\frac{\kappa_{W_{v_2}}}{\kappa_{W_r}} + \kappa_{W_r}\psi_r + \frac{\kappa_{W_r}}{\underline{k}_{PF}\lambda_r}\right)\frac{e^{\lambda_r\epsilon_\tau}}{e^{\lambda_r\epsilon_\tau}-1}.$$

As expected, increasing the value of ϵ_τ, and hence the dwell time, cannot completely eliminate the error induced by $\|\bar{\boldsymbol{e}}_v\|$, but can reduce it to some extent since $\frac{1}{e^{\lambda_r\epsilon_\tau}-1}$ and $\frac{e^{\lambda_r\epsilon_\tau}}{e^{\lambda_r\epsilon_\tau}-1}$ are monotonically decreasing, and

$$\lim_{\epsilon_\tau\to\infty}\frac{1}{e^{\lambda_r\epsilon_\tau}-1} = 0, \qquad \text{and} \qquad \lim_{\epsilon_\tau\to\infty}\frac{e^{\lambda_r\epsilon_\tau}}{e^{\lambda_r\epsilon_\tau}-1} = 1.$$

At this point, it is important to recall the definition $\tau_{R_{1,k}} := \max_{i\in\mathcal{I}_\ell}\tau^i_{R1}(t_{2k-1})$, and the fact that link peers have no knowledge of $\Delta t_{u,k}$. As a result, the right hand side of Equation (C.69) is not implementable. However, one can leverage the following fact:

$$\Delta t_{u,k} \leq \max_{i\in\mathcal{I}_\ell}\Delta t^i_0(t_{2k-1}),$$

and assign individual dwell times

$$\tau^i_{R_1}(t) = \max\left\{\epsilon_\tau, \frac{1}{\lambda_r}\ln\Delta t^i_0(t)\right\} + \max\left\{0, \frac{1}{\lambda_r}\ln\kappa_{W_r}\right\}, \qquad \forall i\in\mathcal{I}_\ell,$$

which guarantee that the inequality in (C.69) is met. Then, gathering the bounds for $s_{0,k}$ through $s_{5,k}$, the evolution of W_r with the number of Ø-☁ cycles can be expressed as

$$W_r(t_{2k}) \leq W_r(t_0)e^{-k\lambda_r\epsilon_\tau} + \tilde{\kappa}_{W_p}\|\boldsymbol{e}_{\boldsymbol{p_0}}\|ke^{-k\tilde{\lambda}\epsilon_\tau} + \left(\tilde{\kappa}_{W_{v_1}} + \tilde{\kappa}_{W_{v_2}}e^{\lambda_r\epsilon_\tau}\right)\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|\sum_{i=1}^{k}e^{-i\lambda_r\epsilon_\tau}, \qquad \text{(C.70)}$$

where $\tilde{\lambda} := \min\{\underline{k}_{PF}, \lambda_r\}$, and constants $\tilde{\kappa}_{W_p}$, $\tilde{\kappa}_{W_{v_1}}$, and $\tilde{\kappa}_{W_{v_2}}$ are defined as follows:

$$\tilde{\kappa}_{W_p} := \frac{\kappa_{W_p}}{\kappa_{W_r}} + \kappa_{W_r}\psi_r e^{-\tilde{\lambda}\epsilon_\tau}, \quad \tilde{\kappa}_{W_{v_1}} := \frac{\kappa_{W_p}}{\kappa_{W_r}\underline{k}_{PF}} + \frac{\kappa_{W_{v_1}}}{\kappa_{W_r}}, \quad \tilde{\kappa}_{W_{v_2}} := \frac{\kappa_{W_{v_2}}}{\kappa_{W_r}} + \kappa_{W_r}\psi_r + \frac{\kappa_{W_r}}{\underline{k}_{PF}\lambda_r}.$$

Then, a similar procedure as in the proof of Lemmas 10 and 19 leads to the bounds in Equations (C.63), (C.64), and (C.65). □

C.6.3 Conditions on the Speed-Tracking Precision and Initial States

To ensure that $|\xi_i(t_{2k}) - \xi_R(t_{2k})|$ eventually converges to the desired temporal window for all $i \in \mathcal{I}$, Equation (C.65) yields

$$\lim_{k\to\infty} |\xi_i(t_{2k}) - \xi_R(t_{2k})| \leq \frac{\tilde{\kappa}_{t,v_1} + \tilde{\kappa}_{t,v_2} e^{\lambda_r \epsilon_\tau}}{e^{\lambda_r \epsilon_\tau} - 1} \|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|,$$

which imposes the following condition on the collective speed-tracking precision:

$$\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| < \frac{e^{\lambda_r \epsilon_\tau} - 1}{\tilde{\kappa}_{t,v_1} + \tilde{\kappa}_{t,v_2} e^{\lambda_r \epsilon_\tau}} \underline{\Delta}_t.$$

Now, to guarantee that the maximum speed constraints are not violated for all $t \geq 0$, mode Ø with $t_{2k} \leq t \leq t_{2k+1}$, and mode ☁ with $t_{2k+1} \leq t \leq t_{2k+2}$ are studied in the following subsections. To visualize the times that correspond to each mode, see Figures C.7 and C.8. In both modes Ø and ☁, the analysis leverages the following expression:

$$\|\boldsymbol{v}_{\boldsymbol{cmd,i}}(t)\| \leq |\dot{\xi}_i(t)| \|\boldsymbol{v}_{\boldsymbol{d,i}}(\xi_i(t))\| + k_{PF,i} \|\boldsymbol{e}_{\boldsymbol{p,i}}(t)\|, \tag{C.71}$$

and focuses on finding an upper bound for $|\dot{\xi}_i(t)|$ that holds for all $t \geq 0$.

Conditions on the Speed-Tracking Precision and Initial States, $\tilde{\omega} = 0$

The analysis of the system in mode Ø, and a similar argument as in the proof of Equation (C.45) imply

$$
\begin{aligned}
|\dot{\xi}_i(t)| &\leq \rho + \kappa_{r,u}\|\boldsymbol{\zeta_u}(t_{2k})\|e^{-\lambda_u(t-t_{2k})} + \frac{\kappa_{r,u_p}}{\underline{k}_{PF} - \lambda_u}\|\boldsymbol{e_p}(t_{2k})\| \left(e^{-\lambda_u(t-t_{2k})} - e^{-\underline{k}_{PF}(t-t_{2k})}\right) + \dots \\
&\quad + \kappa_{r,u_v}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| + k_{e_p}\|\boldsymbol{e_p}(t_{2k})\| + \frac{k_{e_p}}{\underline{k}_{PF}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \\
&\leq \rho + \kappa_{r,u}\|\boldsymbol{\zeta_u}(t_{2k})\| + \left(\kappa_{r,u_p}\psi_u + k_{e_p}\right)\|\boldsymbol{e_p}(t_{2k})\| + \left(\kappa_{r,u_v} + \frac{k_{e_p}}{\underline{k}_{PF}}\right)\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|,
\end{aligned}
$$

for all $t_{2k} \leq t \leq t_{2k+1}$. Figure C.7 illustrates how $|\dot{\xi}_i|$ may evolve between times t_{2k} and t_{2k+1}. Now, leveraging the analysis of the switched system in Section C.6.2, Equation (C.70), the bounds in (C.52), the change of variables in (C.19), and $\|\boldsymbol{\zeta_u}(t_{2k})\| \leq \|\boldsymbol{\zeta}(t_{2k})\|$ yield

$$
\|\boldsymbol{\zeta_u}(t_{2k})\| \leq \kappa_r\|\boldsymbol{\zeta_0}\|e^{-k\lambda_r\epsilon_\tau} + \kappa_{W_p}\frac{\|\boldsymbol{S}^{-1}\|}{\sqrt{a_r}}\|\boldsymbol{e_{p0}}\|ke^{-k\tilde{\lambda}\epsilon_\tau} + \frac{\kappa_{W_{v_1}} + \kappa_{W_{v_2}}e^{\lambda_r\epsilon_\tau}}{e^{\lambda_r\epsilon_\tau} - 1}\frac{\|\boldsymbol{S}^{-1}\|}{\sqrt{a_r}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|. \tag{C.72}
$$

To derive a bound that holds for all k, note that the term $ke^{-k\tilde{\lambda}\epsilon_\tau}$ above reaches its maximum value $\tilde{\psi}$ when $k = \frac{1}{\tilde{\lambda}\epsilon_\tau}$ with

$$
\tilde{\psi} := \frac{e^{-1}}{\tilde{\lambda}\epsilon_\tau}.
$$

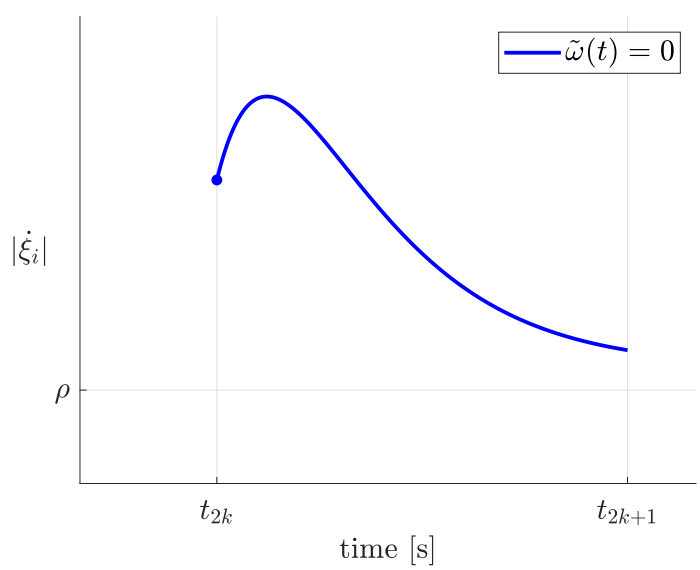


Figure C.7: Coordination rate in mode Ø.

The next inequality follows:

$$\|\boldsymbol{\zeta_u}(t_{2k})\| \leq \kappa_r \|\boldsymbol{\zeta_0}\| + \kappa_{W_p} \tilde{\psi} \frac{\|\boldsymbol{S}^{-1}\|}{\sqrt{a_r}} \|\boldsymbol{e_{p_0}}\| + \frac{\kappa_{W_{v_1}} + \kappa_{W_{v_2}} e^{\lambda_r \epsilon_\tau}}{e^{\lambda_r \epsilon_\tau} - 1} \frac{\|\boldsymbol{S}^{-1}\|}{\sqrt{a_r}} \|\boldsymbol{\bar{e}_v}\|. \tag{C.73}$$

Then, noting that

$$\|\boldsymbol{e_p}(t_{2k})\| \leq \|\boldsymbol{e_{p_0}}\| e^{-k\underline{k}_{PF}\epsilon_\tau} + \frac{1}{\underline{k}_{PF}} \|\boldsymbol{\bar{e}_v}\| \leq \|\boldsymbol{e_{p_0}}\| + \frac{1}{\underline{k}_{PF}} \|\boldsymbol{\bar{e}_v}\|,$$

and plugging the bounds above into the expression for $|\dot{\xi}_i(t)|$ yields

$$|\dot{\xi}_i(t)| \leq \rho + \tilde{\kappa}_{r,u} \|\boldsymbol{\zeta_0}\| + \tilde{\kappa}_{r,u_p} \|\boldsymbol{e_{p_0}}\| + \tilde{\kappa}_{r,u_v} \|\boldsymbol{\bar{e}_v}\|,$$

for all $t_{2k} \leq t \leq t_{2k+1}$, with constants

$$\tilde{\kappa}_{r,u} := \kappa_{r,u} \kappa_r, \qquad \tilde{\kappa}_{r,u_p} := \kappa_{r,u} \kappa_{W_p} \tilde{\psi} \frac{\|\boldsymbol{S}^{-1}\|}{\sqrt{a_r}} + \kappa_{r,u_p} \psi_u + k_{e_p},$$

$$\tilde{\kappa}_{r,u_v} := \kappa_{r,u} \frac{\kappa_{W_{v_1}} + \kappa_{W_{v_2}} e^{\lambda_r \epsilon_\tau}}{e^{\lambda_r \epsilon_\tau} - 1} \frac{\|\boldsymbol{S}^{-1}\|}{\sqrt{a_r}} + \frac{1}{\underline{k}_{PF}} \left(\kappa_{r,u_p} \psi_u + k_{e_p} \right) + \kappa_{r,u_v} + \frac{k_{e_p}}{\underline{k}_{PF}}.$$

The result above and Equation (C.71) imply

$$\|\boldsymbol{v_{cmd,i}}(t)\| \leq \rho v_{d_{\max},i} + \tilde{\kappa}_{\Omega_{u,i}} \|\boldsymbol{\zeta_0}\| + \tilde{\kappa}_{\Omega_{u_p,i}} \|\boldsymbol{e_{p_0}}\| + \tilde{\kappa}_{\Omega_{u_v,i}} \|\boldsymbol{\bar{e}_v}\|,$$

for all $t_{2k} \leq t \leq t_{2k+1}$, all $k \in \mathbb{N}$, and all $i \in \mathcal{I}$, with constants

$$\tilde{\kappa}_{\Omega_{u,i}} := \tilde{\kappa}_{r,u} v_{d_{\max},i}, \qquad \tilde{\kappa}_{\Omega_{u_p,i}} := k_{PF,i} + \tilde{\kappa}_{r,u_p} v_{d_{\max},i}, \qquad \tilde{\kappa}_{\Omega_{u_v,i}} := 1 + \tilde{\kappa}_{r,u_v} v_{d_{\max},i}.$$

As a result, if

$$\|\boldsymbol{\bar{e}_v}\| < \min_{i \in \mathcal{I}} \frac{v_{\max,i} - \rho v_{d_{\max},i}}{\tilde{\kappa}_{\Omega_{u_v,i}}},$$

then there exists a set $\tilde{\Omega}_{u_0}$ such that all $(\boldsymbol{\zeta_0}, \boldsymbol{e_{p_0}}) \in \tilde{\Omega}_{u_0}$ guarantee that $\|\boldsymbol{v_{cmd,i}}(t)\| \leq v_{\max,i}$ is met for all $t_{2k} \leq t \leq t_{2k+1}$, all $k \in \mathbb{N}$, and all $i \in \mathcal{I}$ with

$$\tilde{\Omega}_{u_0} := \left\{ (\boldsymbol{\zeta_0}, \boldsymbol{e_{p_0}}) \in \mathbb{R}^{2n-n_\ell} \times \mathbb{R}^{3n} \,\middle|\, \tilde{\kappa}_{\Omega_{u,i}} \|\boldsymbol{\zeta_0}\| + \tilde{\kappa}_{\Omega_{u_p,i}} \|\boldsymbol{e_{p_0}}\| \leq v_{\max,i} - \rho v_{d_{\max},i} - \tilde{\kappa}_{\Omega_{u_v,i}} \|\boldsymbol{\bar{e}_v}\|, \ \forall i \in \mathcal{I} \right\}.$$

Conditions on the Speed-Tracking Precision and Initial States, $\tilde{\omega} \geq 1$

The analysis of the system in mode ☁, and a similar argument as in the proof of Equation (C.58) imply

$$
\begin{aligned}
|\dot{\xi}_i(t)| &\leq \rho + \kappa_{r,r}\|\boldsymbol{\zeta}(t_{2k+1})\|e^{-\lambda_r(t-t_{2k+1})} + \frac{\kappa_{r,r_p}}{\underline{k}_{PF} - \lambda_r}\|\boldsymbol{e_p}(t_{2k+1})\|\left(e^{-\lambda_r(t-t_{2k+1})} - e^{-\underline{k}_{PF}(t-t_{2k+1})}\right) + \dots \\
&\quad + \kappa_{r,r_v}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| + k_{e_p}\|\boldsymbol{e_p}(t_{2k+1})\| + \frac{k_{e_p}}{\underline{k}_{PF}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \\
&\leq \rho + \kappa_{r,r}\|\boldsymbol{\zeta}(t_{2k+1})\| + \left(\kappa_{r,r_p}\psi_r + k_{e_p}\right)\|\boldsymbol{e_p}(t_{2k+1})\| + \left(\kappa_{r,r_v} + \frac{k_{e_p}}{\underline{k}_{PF}}\right)\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|,
\end{aligned}
$$

for all $t_{2k+1} \leq t \leq t_{2k+2}$. Figure C.8 illustrates how $|\dot{\xi}_i|$ may evolve between times t_{2k}, t_{2k+1}, and t_{2k+2} in the worst-case scenario. Now, to analyze the temporal evolution of $\|\boldsymbol{\zeta}(t)\|$ in mode Ø from t_{2k} to t_{2k+1}, recall that $\boldsymbol{\zeta}(t) := \left[\zeta_t(t), \boldsymbol{\zeta}_{\boldsymbol{u}}^{\top}(t)\right]^{\top}$. Equation (C.46) can be easily extended to find the following bound:

$$
\|\boldsymbol{\zeta_u}(t_{2k+1})\| \leq \kappa_u\|\boldsymbol{\zeta_u}(t_{2k})\| + \kappa_{u_p}\psi_u\|\boldsymbol{e_p}(t_{2k})\| + \kappa_{u_v}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|,
$$

whereas for $|\zeta_t(t)|$ Equation (C.48) indicates that in mode Ø the temporal error can drift with time. While an upper bound for the time spent in mode Ø cannot be derived with the existing

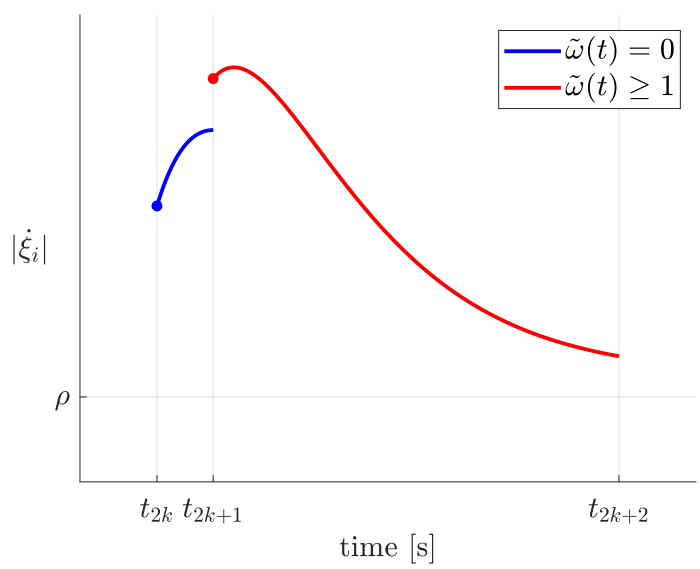


Figure C.8: Coordination rate in modes Ø and ☁.

assumptions, one can leverage the fact that the system switches from mode Ø to mode ☁ if at least one link peer escapes the temporal window. Hence, at t_{2k+1} there is at least one link peer that satisfies either

$$\xi_i(t_{2k+1}) - \xi_R(t_{2k+1}) = -\Delta_t(t_{2k+1}), \qquad \text{or} \qquad \xi_i(t_{2k+1}) - \xi_R(t_{2k+1}) = \Delta_t(t_{2k+1}).$$

Then, $\xi_i(t) - \xi_R(t) = \zeta_t(t) - \boldsymbol{q_i}^\top \boldsymbol{\zeta_c}(t)$, $\|\boldsymbol{q_i}^\top\| = \sqrt{\frac{n-1}{n}}$, $\|\boldsymbol{\zeta_c}(t)\| \leq \|\boldsymbol{\zeta_u}(t)\|$, and $\Delta_t \leq \overline{\Delta}_t < \infty$ can be utilized to consolidate the two equations above into a single expression

$$\begin{aligned}|\zeta_t(t_{2k+1})| &\leq |\Delta_t(t_{2k+1})| + \sqrt{\frac{n-1}{n}}\|\boldsymbol{\zeta_u}(t_{2k+1})\| \\ &\leq \overline{\Delta}_t + \sqrt{\frac{n-1}{n}}\left(\kappa_u\|\boldsymbol{\zeta_u}(t_{2k})\| + \kappa_{u_p}\psi_u\|\boldsymbol{e_p}(t_{2k})\| + \kappa_{u_v}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|\right).\end{aligned}$$

The next bound follows from the inequalities for $|\zeta_t(t_{2k+1})|$ and $\|\boldsymbol{\zeta_u}(t_{2k+1})\|$, and the definition of $\boldsymbol{\zeta}(t)$:

$$\begin{aligned}\|\boldsymbol{\zeta}(t_{2k+1})\| &\leq |\zeta_t(t_{2k+1})| + \|\boldsymbol{\zeta_u}(t_{2k+1})\| \\ &\leq \overline{\Delta}_t + \left(1 + \sqrt{\frac{n-1}{n}}\right)\left(\kappa_u\|\boldsymbol{\zeta_u}(t_{2k})\| + \kappa_{u_p}\psi_u\|\boldsymbol{e_p}(t_{2k})\| + \kappa_{u_v}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|\right).\end{aligned}$$

Now, noting that

$$\begin{aligned}\|\boldsymbol{e_p}(t_{2k})\| &\leq \|\boldsymbol{e_{p_0}}\|e^{-k\underline{k}_{PF}\epsilon_\tau} + \frac{1}{\underline{k}_{PF}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \leq \|\boldsymbol{e_{p_0}}\| + \frac{1}{\underline{k}_{PF}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|, \\ \|\boldsymbol{e_p}(t_{2k+1})\| &\leq \|\boldsymbol{e_{p_0}}\|e^{-k\underline{k}_{PF}\epsilon_\tau} + \frac{1}{\underline{k}_{PF}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| \leq \|\boldsymbol{e_{p_0}}\| + \frac{1}{\underline{k}_{PF}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|,\end{aligned}$$

plugging the bound for $\|\boldsymbol{\zeta}(t_{2k+1})\|$ into the expression for $|\dot{\xi}_i(t)|$, and using Equation (C.73) yields

$$|\dot{\xi}_i(t)| \leq \rho + \kappa_{r,r}\overline{\Delta}_t + \tilde{\kappa}_{r,r}\|\boldsymbol{\zeta_0}\| + \tilde{\kappa}_{r,r_p}\|\boldsymbol{e_{p_0}}\| + \tilde{\kappa}_{r,r_v}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|,$$

for all $t_{2k+1} \leq t \leq t_{2k+2}$, with constants

$$\tilde{\kappa}_{r,r} := \kappa_{r,r}^2\kappa_u\left(1 + \sqrt{\frac{n-1}{n}}\right), \qquad \tilde{\kappa}_{r,r_p} := \kappa_{r,r}\left(1 + \sqrt{\frac{n-1}{n}}\right)\left(\kappa_u\kappa_{W_p}\tilde{\psi}\frac{\|\boldsymbol{S}^{-1}\|}{\sqrt{a_r}} + \kappa_{u_p}\psi_u\right) + \kappa_{r,r_p}\psi_r + k_{e_p},$$

$$\tilde{\kappa}_{r,r_v} := \kappa_{r,r}\left(1+\sqrt{\frac{n-1}{n}}\right)\left(\kappa_u \frac{\kappa_{W_{v_1}}+\kappa_{W_{v_2}}e^{\lambda_r \epsilon_\tau}}{e^{\lambda_r \epsilon_\tau}-1}\frac{\|\boldsymbol{S}^{-1}\|}{\sqrt{a_r}}+\kappa_{u_p}\frac{\psi_u}{\underline{k}_{PF}}+\kappa_{u_v}\right)+\frac{1}{\underline{k}_{PF}}\left(\kappa_{r,r_p}\psi_r+k_{e_p}\right)+\kappa_{r,r_v}+\frac{k_{e_p}}{\underline{k}_{PF}}.$$

The result above and Equation (C.71) imply

$$\|\boldsymbol{v_{cmd,i}}(t)\| \leq \left(\rho+\kappa_{r,r}\overline{\Delta}_t\right)v_{d_{\max},i}+\tilde{\kappa}_{\Omega_{r,i}}\|\boldsymbol{\zeta_0}\|+\tilde{\kappa}_{\Omega_{r_p,i}}\|\boldsymbol{e_{p_0}}\|+\tilde{\kappa}_{\Omega_{r_v,i}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|,$$

for all $t_{2k+1} \leq t \leq t_{2k+2}$, all $k \in \mathbb{N}$, and all $i \in \mathcal{I}$, with constants

$$\tilde{\kappa}_{\Omega_{r,i}} := \tilde{\kappa}_{r,r}v_{d_{\max},i}, \qquad \tilde{\kappa}_{\Omega_{r_p,i}} := k_{PF,i}+\tilde{\kappa}_{r,r_p}v_{d_{\max},i}, \qquad \tilde{\kappa}_{\Omega_{r_v,i}} := 1+\tilde{\kappa}_{r,r_v}v_{d_{\max},i}.$$

As a result, if

$$\overline{\Delta}_t < \min_{i\in\mathcal{I}}\frac{1}{\kappa_{r,r}}\left(\frac{v_{\max,i}}{v_{d_{\max},i}}-\rho\right), \qquad \text{and} \qquad \|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| < \min_{i\in\mathcal{I}}\frac{v_{\max,i}-\left(\rho+\kappa_{r,r}\overline{\Delta}_t\right)v_{d_{\max},i}}{\tilde{\kappa}_{\Omega_{r_v,i}}},$$

then there exists a set $\tilde{\Omega}_{r_0}$ such that all $(\boldsymbol{\zeta_0}, \boldsymbol{e_{p_0}}) \in \tilde{\Omega}_{r_0}$ guarantee that $\|\boldsymbol{v_{cmd,i}}(t)\| \leq v_{\max,i}$ is met for all $t_{2k+1} \leq t \leq t_{2k+2}$, all $k \in \mathbb{N}$, and all $i \in \mathcal{I}$ with

$$\tilde{\Omega}_{r_0} := \left\{(\boldsymbol{\zeta_0}, \boldsymbol{e_{p_0}}) \in \mathbb{R}^{2n-n_\ell}\times\mathbb{R}^{3n} \,\middle|\, \tilde{\kappa}_{\Omega_{r,i}}\|\boldsymbol{\zeta_0}\|+\tilde{\kappa}_{\Omega_{r_p,i}}\|\boldsymbol{e_{p_0}}\| \leq v_{\max,i}-\left(\rho+\kappa_{r,r}\overline{\Delta}_t\right)v_{d_{\max},i}-\tilde{\kappa}_{\Omega_{r_v,i}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|,\ \forall i\in\mathcal{I}\right\}.$$

Finally, gathering the results from modes Ø and ☁, one can conclude that if

$$\|\boldsymbol{e_v}\| < \frac{e^{\lambda_r\epsilon_\tau}-1}{\tilde{\kappa}_{t,v_1}+\tilde{\kappa}_{t,v_2}e^{\lambda_r\epsilon_\tau}}\underline{\Delta}_t,$$

then there exists a $k_\Delta \in \mathbb{N}$ such that

$$|\xi_i(t_{2k})-\xi_R(t_{2k})| \leq \Delta_t(t_{2k}), \qquad \forall k \geq k_\Delta,\ \forall i \in \mathcal{I}.$$

Moreover, if

$$\overline{\Delta}_t < \min_{i\in\mathcal{I}}\frac{1}{\kappa_{r,r}}\left(\frac{v_{\max,i}}{v_{d_{\max},i}}-\rho\right), \quad \text{and} \quad \|\boldsymbol{e_v}\| < \min_{i\in\mathcal{I}}\left\{\frac{v_{\max,i}-\rho v_{d_{\max},i}}{\tilde{\kappa}_{\Omega_{u_v,i}}}, \frac{v_{\max,i}-\left(\rho+\kappa_{r,r}\overline{\Delta}_t\right)v_{d_{\max},i}}{\tilde{\kappa}_{\Omega_{r_v,i}}}\right\},$$

then there is a set $\tilde{\Omega}_0 := \tilde{\Omega}_{u_0} \cap \tilde{\Omega}_{r_0}$ where all $(\boldsymbol{\zeta_0}, \boldsymbol{e_{p_0}}) \in \tilde{\Omega}_0$ guarantee that

$$\|\boldsymbol{v_{cmd,i}}(t)\| \leq v_{\max,i}, \qquad \forall t \geq 0,\ \forall i \in \mathcal{I}.$$

This concludes the proof of Theorem 5. ■

C.6.4 Switching Logic, Proof of Corollary 2

Similar to the proof of Lemma 13, system (C.62) with the switching logic in (6.4) is represented as a state machine, shown in Figure C.9. As a result, the proof is divided into two cases:

1) If (C.62) is in mode ☁ and

$$\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\| < \frac{\underline{\Delta}_t}{\kappa_{t,r_v}}, \tag{C.74}$$

then Lemma 19 ensures that $|\xi_i(t) - \xi_R(t)|$ converges to the desired temporal window $\Delta_t(t)$ for all $i \in \mathcal{I}$. However, only the link peers implement the switching logic, and thus there exists a time $t \geq 0$ such that

$$|\xi_i(t) - \xi_R(t)| < \Delta_t(t), \qquad \forall i \in \mathcal{I}_\ell. \tag{C.75}$$

Since only the link peers may satisfy the condition above, some end peers could be outside of the temporal window, and could pull a link peer outside as well. This gives rise to two cases:

a) If after the individual dwell times of each link peer Equation (C.75) is still met, then the logic in (6.4) switches (C.62) to mode Ø.

b) Otherwise, the system remains in mode ☁. However, Lemma 19 and (C.74) imply that (C.62) cannot remain in this mode indefinitely, and eventually switches to mode Ø.

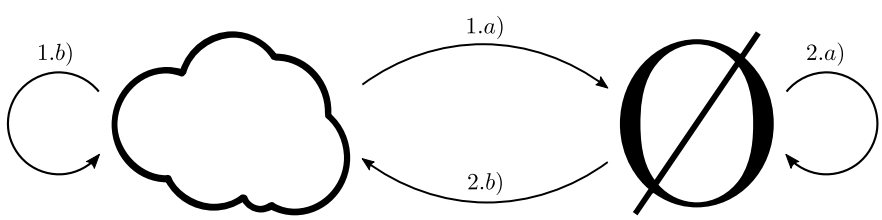


Figure C.9: Finite state machine.

2) If (C.62) is in mode Ø, then Lemma 17 indicates that $|\xi_i(t) - \xi_R(t)|$ can drift away from the origin for all $i \in \mathcal{I}$. Notice, however, that Lemma 17 provides an upper bound for the evolution of $|\xi_i(t) - \xi_R(t)|$, and therefore does not guarantee that the temporal errors will indeed drift. This gives rise to two options:

a) The temporal error for all the link peers stabilizes within the desired temporal window at time t_{Δ_t}, and the system remains in mode Ø

$$|\xi_i(t) - \xi_R(t)| < \Delta_t(t), \qquad \forall i \in \mathcal{I}_\ell, \qquad \forall t \geq t_{\Delta_t}.$$

Lemma 16 can be leveraged to prove that, in the worst-case scenario, the temporal error of the end peers may stabilize outside of the temporal window

$$\begin{aligned}|\xi_i(t) - \xi_R(t)| < \Delta_t(t) &+ \kappa_{c,u}\|\boldsymbol{\zeta_u}(t_{\Delta_t})\|e^{-\lambda_u(t-t_{\Delta_t})} + \dots \\ &+ \frac{\kappa_{c,u_p}}{\underline{k}_{PF} - \lambda_u}\|\boldsymbol{e_p}(t_{\Delta_t})\| \left(e^{-\lambda_u(t-t_{\Delta_t})} - e^{-\underline{k}_{PF}(t-t_{\Delta_t})}\right) + \dots \\ &+ \kappa_{c,u_v}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|, \qquad \forall i \in \mathcal{I}_e, \qquad \forall t \geq t_{\Delta_t}.\end{aligned}$$

b) If $|\xi_i(t) - \xi_R(t)|$ moves[6] sufficiently away from $\Delta_t(t)$ for one or more agents, then Lemma 16 shows that these peers will drag the other cooperating agents outside the temporal window. Hence, some link peers will depart $\Delta_t(t)$, and (C.62) will switch to mode ☁. This Ø-☁ cycle can repeat finitely many times on every bounded time interval. In addition, if

$$\|\boldsymbol{e_v}\| < \frac{e^{\lambda_r \epsilon_\tau} - 1}{\tilde{\kappa}_{t,v_1} + \tilde{\kappa}_{t,v_2} e^{\lambda_r \epsilon_\tau}} \underline{\Delta}_t,$$

then the results in Section C.6.3 imply that there exists a $k_{\Delta_t} \in \mathbb{N}^+$ such that

$$|\xi_i(t_{2k}) - \xi_R(t_{2k})| \leq \Delta_t(t_{2k}), \qquad \forall k \geq k_{\Delta_t}, \qquad \forall i \in \mathcal{I}.$$

Hence, there may not be a stable state in the finite state machine depicted in Figure C.9, and the system dynamics in (C.62) could either

[6]Note that the term drift is not used since a bounded increase in the temporal error can have the same effect.

i) remain in mode Ø; or

ii) persistently switch between modes Ø and ☁.

If the system permanently switches between modes Ø and ☁, one can leverage the switching logic in (6.4) and Lemma 16 to conclude that the temporal error in mode Ø can bounded by

$$
\begin{aligned}
&|\xi_i(t) - \xi_R(t)| < \Delta_t(t), && \forall i \in \mathcal{I}_\ell, \\
&|\xi_i(t) - \xi_R(t)| < \Delta_t(t) + \kappa_{c,u}\|\boldsymbol{\zeta_u}(t_{2k})\| + \kappa_{c,u_p}\psi_u\|\boldsymbol{e_p}(t_{2k})\| + \kappa_{c,u_v}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|, && \forall i \in \mathcal{I}_e,
\end{aligned}
$$

for all $t_{2k} \leq t < t_{2k+1}$. Whereas in mode ☁, similar to the analysis in Section C.6.3, the temporal error can be bounded by

$$
|\xi_i(t) - \xi_R(t)| \leq \kappa_{t,r}\|\boldsymbol{\zeta}(t_{2k+1})\| + \kappa_{t,r_p}\psi_r\|\boldsymbol{e_p}(t_{2k+1})\| + \kappa_{t,r_v}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|, \qquad \forall i \in \mathcal{I},
$$

for all $t_{2k+1} \leq t \leq t_{2k+2}$, where

$$
\|\boldsymbol{\zeta}(t_{2k+1})\| \leq \overline{\Delta}_t + \left(1 + \sqrt{\frac{n-1}{n}}\right)\left(\kappa_u\|\boldsymbol{\zeta_u}(t_{2k})\| + \kappa_{u_p}\psi_u\|\boldsymbol{e_p}(t_{2k})\| + \kappa_{u_v}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|\right).
$$

Combining the expressions above with Equation (C.72) and noting that

$$
\begin{aligned}
\|\boldsymbol{e_p}(t_{2k})\| &\leq \|\boldsymbol{e_{p_0}}\|e^{-k\underline{k}_{PF}\epsilon_\tau} + \frac{1}{\underline{k}_{PF}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|, \\
\|\boldsymbol{e_p}(t_{2k+1})\| &\leq \|\boldsymbol{e_{p_0}}\|e^{-k\underline{k}_{PF}\epsilon_\tau} + \frac{1}{\underline{k}_{PF}}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|,
\end{aligned}
$$

proves that the temporal error is bounded between t_{2k} and t_{2k+2} for all $k \in \mathbb{N}$. □

C.7 Proof of Theorem 6

This proof is a particular case of Lemmas 8, 18, and 19 when $\tilde{\omega}(t) \equiv n_\ell$ and $\boldsymbol{\omega}(t) \equiv \boldsymbol{v}$, as well as the analysis on the speed-tracking precision and initial states in Section C.6.1, which leads to the

same $\kappa_{t,s}$, $\kappa_{c,s}$, and $\kappa_{r,s}$ as in Section C.4

$$\kappa_{t,s_p} := \left(1+\sqrt{\tfrac{n-1}{n}}\right)\kappa_{s_p}, \qquad \kappa_{c,s_p} := \sqrt{2}\kappa_{s_p}, \qquad \kappa_{r,s_p} := \left(k_P n + \max\left\{1, k_R\left(1+\sqrt{\tfrac{n-1}{n}}\right)\right\}\right)\kappa_{s_p},$$
$$\kappa_{t,s_v} := \left(1+\sqrt{\tfrac{n-1}{n}}\right)\kappa_{s_v}, \qquad \kappa_{c,s_v} := \sqrt{2}\kappa_{r_v}, \qquad \kappa_{r,s_v} := \left(k_P n + \max\left\{1, k_R\left(1+\sqrt{\tfrac{n-1}{n}}\right)\right\}\right)\kappa_{s_v},$$

where constants κ_{s_p} and κ_{s_v} are defined as

$$\kappa_{s_p} := k_{e_p}\frac{b_s}{a_s}\|\boldsymbol{S}^{-1}\|\|\boldsymbol{B_z}\|, \qquad \kappa_{s_v} := \frac{1}{\lambda_s}\frac{k_{e_p}}{\underline{k}_{PF}}\frac{b_s}{a_s}\|\boldsymbol{S}^{-1}\|\|\boldsymbol{B_z}\|.$$

The ratio $\frac{b_s}{a_s}$, guaranteed rate of convergence λ_s, and the choice of control gains are the same as in Section C.4. Finally, the set Ω_{s_0} that ensures that $\|\boldsymbol{v_{cmd,i}}(t)\| \leq v_{max,i}$ for all $t \geq 0$, and all $i \in \mathcal{I}$ results

$$\Omega_{s_0} := \left\{(\boldsymbol{\zeta_0}, \boldsymbol{e_{p_0}}) \in \mathbb{R}^{2n-n_\ell-1} \times \mathbb{R}^{3n} \,\middle|\, \kappa_{\Omega_s,i}\|\boldsymbol{\zeta_0}\| + \kappa_{\Omega_{s_p},i}\|\boldsymbol{e_{p_0}}\| \leq v_{max,i} - \rho v_{d_{max},i} - \kappa_{\Omega_{s_v},i}\|\bar{\boldsymbol{e}}_{\boldsymbol{v}}\|,\ \forall i \in \mathcal{I}\right\},$$

with constants

$$\kappa_{\Omega_s,i} := \kappa_{r,s} v_{d_{\max},i}, \quad \kappa_{\Omega_{s_p},i} := k_{PF,i} + \left(\kappa_{r,s_p}\psi_s + k_{e_p}\right) v_{d_{\max},i}, \quad \kappa_{\Omega_{s_v},i} := 1 + \left(\kappa_{r,s_v} + \frac{k_{e_p}}{\underline{k}_{PF}}\right) v_{d_{\max},i},$$

$$\psi_s := \frac{1}{\underline{k}_{PF} - \lambda_s}\left(\left(\frac{\underline{k}_{PF}}{\lambda_s}\right)^{-\frac{\lambda_s}{\underline{k}_{PF}-\lambda_s}} - \left(\frac{\underline{k}_{PF}}{\lambda_s}\right)^{-\frac{\underline{k}_{PF}}{\underline{k}_{PF}-\lambda_s}}\right). \qquad \blacksquare$$

Appendix D

Properties of Relevant Matrices

D.1 Positive Semidefiniteness of M_u

Define $Y_u := 2\frac{k_P}{k_I}\beta_u\left(1-\lambda_u\frac{k_P}{k_I}\frac{n}{n_\ell}\right)$, and the Schur complement of $\boldsymbol{M_u}$

$$\begin{aligned}\boldsymbol{X_u} := &-2\lambda_u\boldsymbol{P_{0_u}} - \dot{\boldsymbol{P}}_{\boldsymbol{0_u}} + k_P\left(\bar{\boldsymbol{L}}\boldsymbol{P_{0_u}} + \boldsymbol{P_{0_u}}\bar{\boldsymbol{L}}\right) - \frac{k_I}{k_P}\left(\boldsymbol{W_e}\boldsymbol{P_{0_u}} + \boldsymbol{P_{0_u}}\boldsymbol{W_e}\right) + \dots \\ &- Y_u^{-1}\left(\beta_u\mathbb{I}_{\mathbf{n-1}} - \boldsymbol{P_{0_u}}\right)\boldsymbol{W_e}\left(\beta_u\mathbb{I}_{\mathbf{n-1}} - \boldsymbol{P_{0_u}}\right).\end{aligned}$$

The properties of Schur complements indicate that if $Y_u > 0$, then $\boldsymbol{M_u} \geq 0$ if and only if $\boldsymbol{X_u} \geq 0$. The following choice of control gains:

$$\frac{k_I}{k_P} = \eta_{I,u}\frac{n}{n_\ell}\lambda_u,$$

with $\eta_{I,u} \geq 2$ ensures that $Y_u > 0$ as well as $\beta_u Y_u^{-1} - \frac{k_I}{k_P} \leq 0$, and along with Lemma 4, $\boldsymbol{W_e} \leq \mathbb{I}_{\mathbf{n-1}}$, $\boldsymbol{W_e}\boldsymbol{P_{0_u}} + \boldsymbol{P_{0_u}}\boldsymbol{W_e} \leq 2\beta_u\mathbb{I}_{\mathbf{n-1}}$, and $\boldsymbol{P_{0_u}}\boldsymbol{W_e}\boldsymbol{P_{0_u}} \leq \beta_u^2\mathbb{I}_{\mathbf{n-1}}$ yield the following bound:

$$\boldsymbol{X_u} \geq \left(\chi_u - 2\lambda_u\beta_u - 2\frac{k_I}{k_P}\beta_u\right)\mathbb{I}_{\mathbf{n-1}}.$$

Recall now that, as stated in Lemma 4, $\beta_u := \frac{\delta_u}{2\gamma_u}$ and $0 < \chi_u \leq \delta_u$. Then, choosing the largest possible $\chi_u = \delta_u$, and a guaranteed rate of convergence

$$\lambda_u = \nu\gamma_u\left(1 + \eta_{I,u}\frac{n}{n_\ell}\right)^{-1},$$

with $\nu \in (0, 1]$, implies that

$$\boldsymbol{X_u} \geq \delta_u\left(1-\nu\right)\mathbb{I}_{\mathbf{n-1}} \geq 0.$$

This proves that $\boldsymbol{M_u}$ is positive semidefinite. ■

D.2 Positive Semidefiniteness of M_r

Define $Y_{r,1} := 2\beta_r \frac{k_P}{k_I}\left(1 - \lambda_r \frac{k_P}{k_I}\right)$, and the Schur complement of $\boldsymbol{M_r}$

$$\boldsymbol{X_{r,1}} := \begin{bmatrix} 2\beta_r\left(k_R\tilde{\omega} - \frac{k_I}{k_P}(n-n_\ell) - n\lambda_r\right) & -\left(k_R\boldsymbol{\omega}^\top + \frac{k_I}{k_P}\boldsymbol{v}^\top\right)\mathbf{Q}^\top\left(\beta_r\mathbb{I}_{\mathbf{n-1}} + \boldsymbol{P_{0_r}}\right) \\ -\left(\beta_r\mathbb{I}_{\mathbf{n-1}} + \boldsymbol{P_{0_r}}\right)\mathbf{Q}\left(k_R\boldsymbol{\omega} + \frac{k_I}{k_P}\boldsymbol{v}\right) & -2\lambda_r\boldsymbol{P_{0_r}} - \dot{\boldsymbol{P}}_{\mathbf{0_r}} + k_P\left(\widehat{\mathbf{L}}\boldsymbol{P_{0_r}} + \boldsymbol{P_{0_r}}\widehat{\mathbf{L}}\right) - \frac{k_I}{k_P}\left(\boldsymbol{W_e}\boldsymbol{P_{0_r}} + \boldsymbol{P_{0_r}}\boldsymbol{W_e}\right) - Y_{r,1}^{-1}\left(\beta_r\mathbb{I}_{\mathbf{n-1}} - \boldsymbol{P_{0_r}}\right)\boldsymbol{W_e}\left(\beta_r\mathbb{I}_{\mathbf{n-1}} - \boldsymbol{P_{0_r}}\right) \end{bmatrix}.$$

Now, define $Y_{r,2} := 2\beta_r\left(k_R\tilde{\omega} - \frac{k_I}{k_P}(n-n_\ell) - n\lambda_r\right)$, and the Schur complement of $\boldsymbol{X_{r,1}}$

$$\begin{aligned}
\boldsymbol{X_{r,2}} := &- 2\lambda_r\boldsymbol{P_{0_r}} - \dot{\boldsymbol{P}}_{\mathbf{0_r}} + k_P\left(\widehat{\mathbf{L}}\boldsymbol{P_{0_r}} + \boldsymbol{P_{0_r}}\widehat{\mathbf{L}}\right) - \frac{k_I}{k_P}\left(\boldsymbol{W_e}\boldsymbol{P_{0_r}} + \boldsymbol{P_{0_r}}\boldsymbol{W_e}\right) - Y_{r,1}^{-1}\left(\beta_r\mathbb{I}_{\mathbf{n-1}} - \boldsymbol{P_{0_r}}\right)\boldsymbol{W_e}\left(\beta_r\mathbb{I}_{\mathbf{n-1}} - \boldsymbol{P_{0_r}}\right) + \dots \\
&- Y_{r,2}^{-1}\left(\beta_r\mathbb{I}_{\mathbf{n-1}} + \boldsymbol{P_{0_r}}\right)\mathbf{Q}\left(k_R\boldsymbol{\omega} + \frac{k_I}{k_P}\boldsymbol{v}\right)\left(k_R\boldsymbol{\omega}^\top + \frac{k_I}{k_P}\boldsymbol{v}^\top\right)\mathbf{Q}^\top\left(\beta_r\mathbb{I}_{\mathbf{n-1}} + \boldsymbol{P_{0_r}}\right) \\
= &- 2\lambda_r\boldsymbol{P_{0_r}} - \dot{\boldsymbol{P}}_{\mathbf{0_r}} + k_P\left(\widehat{\mathbf{L}}\boldsymbol{P_{0_r}} + \boldsymbol{P_{0_r}}\widehat{\mathbf{L}}\right) + \left(\beta_r Y_{r,1}^{-1} - \frac{k_I}{k_P}\right)\left(\boldsymbol{W_e}\boldsymbol{P_{0_r}} + \boldsymbol{P_{0_r}}\boldsymbol{W_e}\right) - Y_{r,1}^{-1}\left(\beta_r^2\boldsymbol{W_e} + \boldsymbol{P_{0_r}}\boldsymbol{W_e}\boldsymbol{P_{0_r}}\right) + \dots \\
&- Y_{r,2}^{-1}\beta_r^2\left(k_R^2\mathbf{Q}\boldsymbol{\omega}\boldsymbol{\omega}^\top\mathbf{Q}^\top + k_R\frac{k_I}{k_P}\left(\mathbf{Q}\boldsymbol{v}\boldsymbol{\omega}^\top\mathbf{Q}^\top + \mathbf{Q}\boldsymbol{\omega}\boldsymbol{v}^\top\mathbf{Q}^\top\right) + \frac{k_I^2}{k_P^2}\mathbf{Q}\boldsymbol{v}\boldsymbol{v}^\top\mathbf{Q}^\top\right) + \dots \\
&- Y_{r,2}^{-1}\beta_r\left(k_R^2\left(\boldsymbol{P_{0_r}}\mathbf{Q}\boldsymbol{\omega}\boldsymbol{\omega}^\top\mathbf{Q}^\top + \mathbf{Q}\boldsymbol{\omega}\boldsymbol{\omega}^\top\mathbf{Q}^\top\boldsymbol{P_{0_r}}\right) + k_R\frac{k_I}{k_P}\left(\boldsymbol{P_{0_r}}\mathbf{Q}\boldsymbol{v}\boldsymbol{\omega}^\top\mathbf{Q}^\top + \mathbf{Q}\boldsymbol{\omega}\boldsymbol{v}^\top\mathbf{Q}^\top\boldsymbol{P_{0_r}} + \boldsymbol{P_{0_r}}\mathbf{Q}\boldsymbol{\omega}\boldsymbol{v}^\top\mathbf{Q}^\top + \mathbf{Q}\boldsymbol{v}\boldsymbol{\omega}^\top\mathbf{Q}^\top\boldsymbol{P_{0_r}}\right) + \frac{k_I^2}{k_P^2}\left(\boldsymbol{P_{0_r}}\mathbf{Q}\boldsymbol{v}\boldsymbol{v}^\top\mathbf{Q}^\top + \mathbf{Q}\boldsymbol{v}\boldsymbol{v}^\top\mathbf{Q}^\top\boldsymbol{P_{0_r}}\right)\right) + \dots \\
&- Y_{r,2}^{-1}\left(k_R^2\boldsymbol{P_{0_r}}\mathbf{Q}\boldsymbol{\omega}\boldsymbol{\omega}^\top\mathbf{Q}^\top\boldsymbol{P_{0_r}} + k_R\frac{k_I}{k_P}\left(\boldsymbol{P_{0_r}}\mathbf{Q}\boldsymbol{v}\boldsymbol{\omega}^\top\mathbf{Q}^\top\boldsymbol{P_{0_r}} + \boldsymbol{P_{0_r}}\mathbf{Q}\boldsymbol{\omega}\boldsymbol{v}^\top\mathbf{Q}^\top\boldsymbol{P_{0_r}}\right) + \frac{k_I^2}{k_P^2}\boldsymbol{P_{0_r}}\mathbf{Q}\boldsymbol{v}\boldsymbol{v}^\top\mathbf{Q}^\top\boldsymbol{P_{0_r}}\right).
\end{aligned}$$

Next, concatenating the properties of Schur complements, if $Y_{r,1} > 0$ and $Y_{r,2} > 0$, then $\boldsymbol{M_r} \geq 0$ if and only if $\boldsymbol{X_{r,2}} \geq 0$. Note that

$$\begin{aligned}
\mathbf{Q}\boldsymbol{v}\boldsymbol{v}^\top\mathbf{Q}^\top &\leq u_v^2\mathbb{I}_{\mathbf{n-1}}, & \mathbf{Q}\boldsymbol{\omega}\boldsymbol{\omega}^\top\mathbf{Q}^\top &\leq u_\omega^2\mathbb{I}_{\mathbf{n-1}}, & \boldsymbol{W_e} &\leq \mathbb{I}_{\mathbf{n-1}} \\
\boldsymbol{P_{0_r}}\mathbf{Q}\boldsymbol{v}\boldsymbol{v}^\top\mathbf{Q}^\top\boldsymbol{P_{0_r}} &\leq \beta_r^2 u_v^2\mathbb{I}_{\mathbf{n-1}}, & \boldsymbol{P_{0_r}}\mathbf{Q}\boldsymbol{\omega}\boldsymbol{\omega}^\top\mathbf{Q}^\top\boldsymbol{P_{0_r}} &\leq \beta_r^2 u_\omega^2\mathbb{I}_{\mathbf{n-1}}, & \boldsymbol{P_{0_r}}\boldsymbol{W_e}\boldsymbol{P_{0_r}} &\leq \beta_r^2\mathbb{I}_{\mathbf{n-1}}, \\
\boldsymbol{P_{0_r}}\mathbf{Q}\boldsymbol{v}\boldsymbol{v}^\top\mathbf{Q}^\top + \mathbf{Q}\boldsymbol{v}\boldsymbol{v}^\top\mathbf{Q}^\top\boldsymbol{P_{0_r}} &\leq 2\beta_r u_v^2\mathbb{I}_{\mathbf{n-1}}, & \boldsymbol{P_{0_r}}\mathbf{Q}\boldsymbol{\omega}\boldsymbol{\omega}^\top\mathbf{Q}^\top + \mathbf{Q}\boldsymbol{\omega}\boldsymbol{\omega}^\top\mathbf{Q}^\top\boldsymbol{P_{0_r}} &\leq 2\beta_r u_\omega^2\mathbb{I}_{\mathbf{n-1}}, & \boldsymbol{P_{0_r}}\boldsymbol{W_e} + \boldsymbol{W_e}\boldsymbol{P_{0_r}} &\leq 2\beta_r\mathbb{I}_{\mathbf{n-1}},
\end{aligned}$$

$$\begin{aligned}
\mathbf{Q}\boldsymbol{v}\boldsymbol{\omega}^\top\mathbf{Q}^\top + \mathbf{Q}\boldsymbol{\omega}\boldsymbol{v}^\top\mathbf{Q}^\top &\leq 2u_v u_\omega\mathbb{I}_{\mathbf{n-1}}, \\
\boldsymbol{P_{0_r}}\mathbf{Q}\boldsymbol{v}\boldsymbol{\omega}^\top\mathbf{Q}^\top + \mathbf{Q}\boldsymbol{\omega}\boldsymbol{v}^\top\mathbf{Q}^\top\boldsymbol{P_{0_r}} &\leq 2u_v u_\omega\beta_r\mathbb{I}_{\mathbf{n-1}}, \\
\boldsymbol{P_{0_r}}\mathbf{Q}\boldsymbol{\omega}\boldsymbol{v}^\top\mathbf{Q}^\top + \mathbf{Q}\boldsymbol{v}\boldsymbol{\omega}^\top\mathbf{Q}^\top\boldsymbol{P_{0_r}} &\leq 2u_v u_\omega\beta_r\mathbb{I}_{\mathbf{n-1}}, \\
\boldsymbol{P_{0_r}}\mathbf{Q}\boldsymbol{\omega}\boldsymbol{v}^\top\mathbf{Q}^\top\boldsymbol{P_{0_r}} + \boldsymbol{P_{0_r}}\mathbf{Q}\boldsymbol{v}\boldsymbol{\omega}^\top\mathbf{Q}^\top\boldsymbol{P_{0_r}} &\leq 2u_v u_\omega\beta_r^2\mathbb{I}_{\mathbf{n-1}},
\end{aligned}$$

and choose

$$\frac{k_I}{k_P} = \eta_{I,r}\lambda_r, \qquad k_R = \eta_{R,r} n \xi_r \lambda_r,$$

with $\eta_{I,r} \geq 2$ and $\eta_{R,r} > 1$. This ensures that $Y_{r,1} > 0$, $Y_{r,2} > 0$, and $\beta_r Y_{r,1}^{-1} - \frac{k_I}{k_P} \leq 0$. Lemma 8, these matrix properties, and the choice of control gains yield the following bound:

$$\boldsymbol{X_{r,2}} \geq \left(\chi_r - 2\lambda_r\beta_r - 2\frac{k_I}{k_P}\beta_r - 2\lambda_r f_r \beta_r\right)\mathbb{I}_{\mathbf{n-1}}.$$

Recall now that, as stated in Lemma 8, $\beta_r := \frac{\delta_r}{2\gamma_r}$ and $0 < \chi_r \leq \delta_r$. Then, choosing the largest possible $\chi_r = \delta_r$, and a guaranteed rate of convergence

$$\lambda_r = \gamma_r \left(1 + \eta_{I,r} + f_r\right)^{-1},$$

implies that $\boldsymbol{X_{r,2}} \geq 0$. This proves that $\boldsymbol{M_r}$ is positive semidefinite. However, the value for λ_r above is not as straightforward to compute as it may seem at first glance. Recall that

$$\gamma_r := \frac{k_P n \mu}{\left(1 + \left(k_P n + k_R\right) T\right)^2}.$$

Since k_R is expressed in terms of λ_r, further algebraic manipulation of the expressions above yields the following polynomial equation:

$$\lambda_r \left(1 + k_P n T + \eta_{R,r} n \xi_r T \lambda_r\right)^2 - \frac{k_P n \mu}{1 + \eta_{I,r} + f_r} = a_3\lambda_r^3 + a_2\lambda_r^2 + a_1\lambda_r + a_0 = 0,$$

with coefficients $a_3, a_2, a_1 > 0$ and $a_0 < 0$

$$a_3 := \left(\eta_{R,r} n \xi_r T\right)^2, \quad a_2 := 2\eta_{R,r} n \xi_r T \left(1 + k_P n T\right), \quad a_1 := \left(1 + k_P n T\right)^2, \quad a_0 := -\frac{k_P n \mu}{1 + \eta_{I,r} + f_r}.$$

Routh's stability criterion can be used to conclude that the polynomial equation above always has a single positive real root, and two complex conjugate roots. Hence, for every choice of design parameters $\eta_{I,r} \geq 2$ and $\eta_{R,r} > 1$ there exists a $\lambda_r > 0$ that renders $\boldsymbol{M_r} \geq 0$. ■

References

[1] F. Augugliaro, A. P. Schoellig, and R. D'Andrea, "Generation of collision-free trajectories for a quadcopter fleet: A sequential convex programming approach," in *International Conference on Intelligent Robots and Systems (IROS)*, Vilamoura, Algarve, Portugal, October 2012, pp. 1917–1922.

[2] R. Choe, "Distributed cooperative trajectory generation for multiple autonomous vehicles using Pythagorean Hodograph Bézier curves," Ph.D. dissertation, University of Illinois at Urbana-Champaign, Urbana, IL, USA, 2017.

[3] H. Bai, M. Arcak, and J. Wen, *Cooperative Control Design, A Systematic, Passivity-Based Approach.* New York, NY, USA: Springer, 2011.

[4] E. Xargay, "Time-critical cooperative path-following control of multiple unmanned aerial vehicles," Ph.D. dissertation, University of Illinois at Urbana-Champaign, Urbana, IL, USA, 2013.

[5] P. Doherty, J. Kvarnström, and F. Heintz, "A temporal logic-based planning and execution monitoring framework for unmanned aircraft systems," *Autonomous Agents and Multi-Agent Systems*, vol. 19, no. 3, pp. 332–377, 2009.

[6] J. Wang, C. Jiang, Z. Han, Y. Ren, R. G. Maunder, and L. Hanzo, "Taking drones to the next level: Cooperative distributed unmanned-aerial-vehicular networks for small and mini drones," *IEEE Vehicular Technology Magazine*, vol. 12, no. 3, pp. 73–82, September 2017.

[7] J. Tisdale, Z. Kim, and J. K. Hedrick, "An autonomous system for cooperative search and localization using unmanned vehicles," in *AIAA Guidance, Navigation and Control Conference*, Honolulu, HI, USA, August 2008.

[8] Y. Ma, Y. Zhao, X. Qi, Y. Zheng, and R. Gan, "Cooperative communication framework design for the unmanned aerial vehicles - unmanned surface vehicles formation," *Advances in Mechanical Engineering*, vol. 10, no. 5, January 2018.

[9] S. M. Lavalle, "Rapidly-exploring random trees: A new tool for path planning," Iowa State University, Tech. Rep., 1998.

[10] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[11] A. H. Qureshi and Y. Ayaz, "Potential functions based sampling heuristic for optimal path planning," *Autonomous Robots*, vol. 40, no. 6, pp. 1079–1093, August 2016.

[12] A. Perez, R. Platt, G. Konidaris, L. P. Kaelbling, and T. Lozano-Pérez, "LQR-RRT*: Optimal sampling-based motion planning with automatically derived extension heuristics," in *IEEE International Conference on Robotics and Automation (ICRA)*, Saint Paul, MN, USA, 2012, pp. 2537–2542.

[13] C. Urmson and R. Simmons, "Approaches for heuristically biasing RRT growth," in *International Conference on Intelligent Robots and Systems (IROS)*, vol. 2, Las Vegas, NV, USA, October 2003, pp. 1178–1183.

[14] B. Akgun and M. Stilman, "Sampling heuristics for optimal motion planning in high dimensions," in *International Conference on Intelligent Robots and Systems (IROS)*, San Francisco, CA, USA, September 2011, pp. 2640–2645.

[15] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *IEEE International Conference on Robotics and Automation (ICRA)*, vol. 2, San Francisco, CA, USA, April 2000, pp. 995–1001.

[16] M. Du, J. Chen, P. Zhao, H. Liang, Y. Xin, and T. Mei, "An improved RRT-based motion planner for autonomous vehicle in cluttered environments," in *IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, China, May 2014, pp. 4674–4679.

[17] N. M. Amato, O. B. Bayazit, L. K. Dale, C. Jones, and D. Vallejo, "OBPRM: An obstacle-based PRM for 3D workspaces," in *Algorithmic Foundations of Robotics*, Natick, MA, USA, 1998.

[18] L. Zhang and D. Manocha, "An efficient retraction-based RRT planner," in *IEEE International Conference on Robotics and Automation (ICRA)*, Pasadena, CA, USA, May 2008, pp. 3743–3750.

[19] J. Denny, R. Sandström, A. Bregger, and N. M. Amato, "Dynamic region-biased rapidly-exploring random trees," in *Algorithmic Foundations of Robotics*, San Francisco, CA, USA, December 2016, pp. 640–655.

[20] R. Carli and S. Zampieri, "Network clock synchronization based on the second-order linear consensus algorithm," *IEEE Transactions on Automatic Control*, vol. 59, no. 2, pp. 33–43, February 2014.

[21] R. Olfati-Saber and R. M. Murray, "Consensus protocols for networks of dynamic agents," in *American Control Conference*, Denver, CO, USA, June 2003, pp. 951–956.

[22] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1520–1533, September 2004.

[23] E. Xargay, I. Kaminer, A. M. Pascoal, N. Hovakimyan, V. Dobrokhodov, V. Cichella, A. P. Aguiar, and R. Ghabcheloo, "Time-critical cooperative path following of multiple unmanned aerial vehicles over time-varying networks," *Journal of Guidance, Control and Dynamics*, vol. 36, no. 2, pp. 499–516, March 2013.

[24] R. Ghabcheloo, "Coordinated path following of multiple autonomous vehicles," Ph.D. dissertation, Technical University of Lisbon, Lisbon, Portugal, 2007.

[25] T. W. McLain and R. W. Beard, "Coordination variables, coordination functions, and cooperative timing missions," *Journal of Guidance, Control and Dynamics*, vol. 28, no. 1, pp. 150–161, January 2005.

[26] E. Xargay, R. Choe, N. Hovakimyan, and I. Kaminer, "Multi-leader coordination algorithm for networks with switching topology and quantized information," *Automatica*, vol. 50, no. 3, pp. 841–851, March 2014.

[27] N. R. Chowdhury, S. Sukumar, and N. Balachandran, "Persistence based convergence rate analysis of consensus protocols for dynamic graph networks," *European Journal of Control*, vol. 29, pp. 33–43, May 2016.

[28] K. S. Narendra and A. M. Annaswamy, *Stable Adaptive Systems*. Mineola, NY, USA: Dover, 2005.

[29] S. A. Kumar, N. Chowdhury, S. Srikant, and J. Raisch, "Consensus analysis of systems with time-varying interactions : An event-triggered approach," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 9349–9354, July 2017.

[30] J. Puig-Navarro, N. Hovakimyan, and B. D. Allen, "Time-coordination strategies and control laws for multi-agent unmanned systems," in *AIAA Aviation Forum*, Denver, CO, USA, June 2017, AIAA 2017-3990.

[31] L. Lapierre, D. Soetanto, and A. Pascoal, "Coordinated motion control of marine robots," *IFAC Proceedings Volumes*, vol. 36, no. 21, pp. 217–222, 2003.

[32] V. Cichella, R. Choe, S. B. Mehdi, E. Xargay, N. Hovakimyan, V. Dobrokhodov, I. Kaminer, A. M. Pascoal, and A. P. Aguiar, "Safe coordinated maneuvering of teams of multirotor unmanned aerial vehicles: A cooperative control framework for multivehicle, time-critical missions," *IEEE Control Systems Magazine*, vol. 36, no. 4, pp. 59–82, August 2016.

[33] J. Puig-Navarro, N. Hovakimyan, and B. D. Allen, "Time-coordination strategies and control laws for multi-agent unmanned systems," in *AIAA Aviation Technology, Integration, and Operations Conference*, Denver, CO, USA, June 2017.

[34] J. Puig-Navarro, N. Hovakimyan, N. Alexandrov, and B. D. Allen, "Silhouette-informed trajectory generation through a wire maze for UAS," in *AIAA Aviation Technology, Integration, and Operations Conference*, Atlanta, GA, USA, June 2018.

[35] A. P. Aguiar, J. P. Hespanha, and P. V. Kokotović, "Performance limitations in reference tracking and path following for nonlinear systems," *Automatica*, vol. 44, no. 3, pp. 598–610, 2008.

[36] "Vision for Trajectory Based Operations," https://www.faa.gov/air_traffic/technology/tbo/documents/, September 2017, online accessed: 7 April 2021.

[37] R. T. Farouki, *Pythagorean-Hodograph Curves*. Davis, CA, USA: Springer, 2008.

[38] B. A. Barsky and T. D. DeRose, "Geometric continuity of parametric curves: Three equivalent characterizations," *IEEE Computer Graphics & Applications*, vol. 9, no. 6, pp. 60–68, November 1989.

[39] D. Soetanto, L. Lapierre, and A. Pascoal, "Adaptive, non-singular path-following control of dynamic wheeled robots," in *IEEE International Conference on Decision and Control*, vol. 2, Maui, HI, USA, December 2003, pp. 1765–1770.

[40] V. T. Taranenko, "Experience of employment of Ritz's, Poincare's, and Lyapunov's methods for solving flight dynamics problems," in *Air Force Engineering Academy Press*, Moscow, Russia, 1968, (in Russian).

[41] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic," in *International Conference on Intelligent Robots and Systems (IROS)*, Chicago, IL, USA, September 2014, pp. 2997–3004.

[42] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Batch informed trees (BIT*): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs," in *IEEE International Conference on Robotics and Automation (ICRA)*, Seattle, WA, USA, May 2015, pp. 3067–3074.

[43] V. Cichella, "Cooperative autonomous systems: Motion planning and coordinated tracking control for multi-vehicle missions," Ph.D. dissertation, University of Illinois at Urbana-Champaign, Urbana, IL, USA, 2018.

[44] M. Arcak, "Passivity as a design tool for group coordination," *IEEE Transactions on Automatic Control*, vol. 52, no. 8, pp. 1380–1390, August 2007.

[45] M. Studinger, "Operation IceBridge, NASA Blog," http://blogs.nasa.gov/icebridge/tag/studinger/, April 2010, online accessed: 15 October 2019.

[46] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations.* Cambridge, MA, USA: MIT Press, 2005.

[47] B. Chazelle, "Convex partitions of polyhedra: A lower bound and worst-case optimal algorithm," *SIAM Journal on Computing*, vol. 13, no. 3, pp. 488–507, August 1984.

[48] F. P. Preparata and S. J. Hong, "Convex hulls of finite sets of points in two and three dimensions," *Communications of the ACM*, vol. 20, no. 2, p. 87–93, February 1977.

[49] F. P. Preparata and M. I. Shamos, *Computational geometry.* New York, USA: Springer, 1985.

[50] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Transactions on Mathematical Software*, vol. 22, no. 4, p. 469–483, December 1996.

[51] B. Chazelle, "An optimal convex hull algorithm in any fixed dimension," *Discrete & Computational Geometry*, vol. 10, no. 4, p. 377–409, December 1993.

[52] B. Grünbaum, *Convex Polytopes.* New York, USA: Springer, 2003.

[53] D. Muller and F. Preparata, "Finding the intersection of two convex polyhedra," *Theoretical Computer Science*, vol. 7, no. 2, pp. 217–236, March 1978.

[54] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars, *Computational Geometry, Algorithms and Applications*, 3rd ed. Berlin, Germany: Springer, 2008.

[55] G. van den Bergen, *Collision Detection in Interactive 3D Environments.* San Francisco, CA, USA: Morgan Kaufmann, 2003.

[56] S. Bernstein, "Demonstration of a theorem of Weierstrass based on the calculus of probabilities," *Communications of the Kharkov Mathematical Society*, vol. 13, pp. 1–2, 1912, (in Russian).

[57] P. Bézier, "Essai de définition numérique des courbes et surfaces experimentale," Ph.D. dissertation, Université de Paris, Paris, France, 1977, (in French).

[58] E. G. Gilbert, D. W. Johnson, and S. S. Keerthi, "A fast procedure for computing the distance between complex objects in three-dimensional space," *IEEE Journal on Robotics and Automation*, vol. 4, no. 2, pp. 193–203, April 1988.

[59] Chong Jin Ong and E. G. Gilbert, "The Gilbert-Johnson-Keerthi distance algorithm: a fast version for incremental motions," in *IEEE International Conference on Robotics and Automation (ICRA)*, vol. 2, Albuquerque, NM, USA, April 1997, pp. 1183–1189.

[60] M. Montanari, N. Petrinic, and E. Barbieri, "Improving the GJK algorithm for faster and more reliable distance queries between convex objects," *ACM Transactions on Graphics*, vol. 36, no. 3, pp. 30:1–30:17, June 2017.

[61] J.-W. Chang, Y.-K. Choi, M.-S. Kim, and W. Wang, "Computation of the minimum distance between two Bézier curves/surfaces," *Computers & Graphics*, vol. 35, no. 3, pp. 677–684, June 2011.

[62] T. D. DeRose, "Composing Bézier simplexes," *ACM Transactions on Graphics*, vol. 7, no. 3, p. 198–221, July 1988.

[63] M. Chandarana, E. L. Meszaros, and B. D. Allen, "A planning pipeline for large multi-agent missions," in *12th International Conference on Advances in Computer-Human Interactions*, Athens, Greece, February 2019, pp. 74–79.

[64] R. T. Farouki and T. N. T. Goodman, "On the optimal stability of the Bernstein basis," *Mathematics of Computation*, vol. 65, no. 216, pp. 1553–1566, October 1996.

[65] T. Pavlidis, *Algorithms for graphics and image processing*, 1st ed. Rockville, MD, USA: Computer Science Press, 1982.

[66] J. M. Lane and R. F. Riesenfeld, "Bounds on a polynomial," *BIT Numerical Mathematics*, vol. 21, no. 1, pp. 112–117, March 1981.

[67] T. Nishita, T. W. Sederberg, and M. Kakimoto, "Ray tracing trimmed rational surface patches," *Computer Graphics*, vol. 24, no. 4, pp. 337–345, September 1990.

[68] J. Stoer and R. Bulirsch, *Introduction to numerical analysis*, 3rd ed. New York. NY, USA: Springer-Verlag, 2002.

[69] M. Moscato, L. Titolo, A. Dutle, and C. A. Muñoz, "Automatic estimation of verified floating-point round-off errors via static analysis," in *Computer Safety, Reliability, and Security*, Trento, Italy, September 2017, pp. 213–229.

[70] L. Titolo, M. A. Feliú, M. Moscato, and C. A. Muñoz, "An abstract interpretation framework for the round-off error analysis of floating-point programs," in *Verification, Model Checking, and Abstract Interpretation*, Los Angeles, CA, USA, January 2018, pp. 516–537.

[71] E. Larsen, S. Gottschalk, M. C. Lin, and D. Manocha, "Fast proximity queries with swept sphere volumes," University of North Carolina Chapel Hill, Tech. Rep. TR99-018, 1999.

[72] S. A. Ehmann and M. C. Lin, "Accelerated proximity queries between convex polyhedra by multi-level Voronoi marching," in *International Conference on Intelligent Robots and Systems (IROS)*, Takamatsu, Japan, November 2000, pp. 2101–2106.

[73] J. Pan, S. Chitta, and D. Manocha, "FCL: A general purpose library for collision and proximity queries," in *IEEE International Conference on Robotics and Automation (ICRA)*, Saint Paul, MN, USA, May 2012, pp. 3859–3866.

[74] S. Cameron, "Enhancing GJK: Computing minimum and penetration distances between convex polyhedra," in *IEEE International Conference on Robotics and Automation (ICRA)*, vol. 4, Albuquerque, NM, USA, April 1997, pp. 3112–3117.

[75] M. C. Lin and J. F. Canny, "A fast algorithm for incremental distance calculation," in *IEEE International Conference on Robotics and Automation (ICRA)*, Sacramento, CA, USA, April 1991, pp. 1008–1014.

[76] L. Barba and S. Langerman, "Optimal detection of intersections between convex polyhedra," in *ACM-SIAM Symposium on Discrete Algorithms*, San Diego, CA, USA, January 2015, pp. 1641–1654.

[77] D. P. Dobkin and D. G. Kirkpatrick, "A linear algorithm for determining the separation of convex polyhedra," *Journal of Algorithms*, vol. 6, no. 3, pp. 381–392, September 1985.

[78] N. M. Patrikalakis and T. Maekawa, *Shape Interrogation for Computer Aided Design and Manufacturing*. New York, NY, USA: Springer-Verlag, 2010.

[79] C.-H. Park, G. Elber, K.-J. Kim, G.-Y. Kim, and J.-K. Seong, "A hybrid parallel solver for systems of multivariate polynomials using CPUs and GPUs," *Computer-Aided Design*, vol. 43, no. 11, pp. 1360–1369, November 2011.

[80] X.-D. Chen, L. Chen, Y. Wang, G. Xu, J.-H. Yong, and J.-C. Paul, "Computing the minimum distance between two Bézier curves," *Journal of Computational and Applied Mathematics*, vol. 229, no. 1, pp. 94–301, July 2009.

[81] A. Lakshmanan, A. Patterson, V. Cichella, and N. Hovakimyan, "Proximity queries for absolutely continuous parametric curves," in *Proceedings of Robotics: Science and Systems*, Freiburgim-Breisgau, Germany, June 2019.

[82] S. Redon, A. Kheddar, and S. Coquillart, "Fast continuous collision detection between rigid bodies," *Computer Graphics Forum*, vol. 21, no. 3, pp. 279–287, September 2002.

[83] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed. Cambridge, MA, USA: The MIT Press, 2009.

[84] T. M. Chan, "Optimal output-sensitive convex hull algorithms in two and three dimensions," *Discrete & Computational Geometry*, vol. 16, no. 4, pp. 361–368, April 1996.

[85] M. Kleinbort, O. Salzman, and D. Halperin, "Collision detection or nearest-neighbor search? on the computational bottleneck in sampling-based motion planning," in *Algorithmic Foundations of Robotics*, San Francisco, CA, USA, December 2020, pp. 624–639.

[86] D. Hsu, L. E. Kavraki, J.-C. Latombe, R. Motwani, and S. Sorkin, "On finding narrow passages with probabilistic roadmap planners," in *Robotics: the algorithmic perspective*, Houston, TX, USA, August 1998, p. 141–153.

[87] L. E. Kavraki, P. Švestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, August 1996.

[88] J. E. Goodman, J. O'Rourke, and C. Tóth, *Handbook of Discrete and Computational Geometry*, 3rd ed. New York, USA: Taylor & Francis Group, 2018.

[89] S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

[90] S. M. LaValle and J. James J. Kuffner, "Randomized kinodynamic planning," *The International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[91] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization*. Mineola, NY, USA: Dover, 1998.

[92] R. T. Farouki and K. M. Nittler, "Efficient high-speed cornering motions based on continuously-variable feedrates. I. real-time interpolator algorithms," *The International Journal of Advanced Manufacturing Technology*, vol. 87, no. 9, pp. 3557–3568, December 2016.

[93] R. T. Farouki, "Efficient high-speed cornering motions based on continuously-variable feedrates. II. implementation and performance analysis," *The International Journal of Advanced Manufacturing Technology*, vol. 88, no. 1, pp. 159–174, January 2017.

[94] R. T. Farouki, C. Giannelli, M. L. Sampoli, and A. Sestini, "Rotation-minimizing osculating frames," *Computer Aided Geometric Design*, vol. 31, no. 1, pp. 27–42, January 2014.

[95] R. T. Farouki, "The Bernstein polynomial basis: A centennial retrospective," *Computer Aided Geometry Design*, vol. 29, no. 6, pp. 379–419, August 2012.

[96] S. Klemm, J. Oberländer, A. Hermann, A. Roennau, T. Schamm, J. M. Zollner, and R. Dillmann, "RRT*-Connect: Faster, asymptotically optimal motion planning," in *International Conference on Robotics and Biomimetics*, Zhuhai, China, December 2015, pp. 1670–1677.

[97] J. W. Vervoorst, "A modular simulation environment for the improved dynamic simulation of multirotor unmanned aerial vehicles," M.S. thesis, University of Illinois at Urbana-Champaign, Urbana, IL, USA, 2016.

[98] V. Stepanyan, K. S. Krishnakumar, and A. Bencomo, "Identification and reconfigurable control of impaired multi-rotor drones," in *AIAA Guidance, Navigation and Control Conference*, San Diego, CA, USA, January 2016, AIAA 2016-1384. pp. 1–20.

[99] V. Stepanyan and K. S. Krishnakumar, "Estimation, navigation and control of multi-rotor drones in an urban wind field," in *AIAA Information Systems*, Grapevine, TX, USA, January 2017, AIAA 2017-0670. pp. 1–26.

[100] T. Lee, M. Leok, and N. H. McClamroch, "Geometric tracking control of a quadrotor UAV on SE(3)," in *IEEE Conference on Decision and Control*, Atlanta, GA, USA, December 2010, pp. 5420–5425.

[101] T. Lee, "Geometric tracking control of the attitude dynamics of a rigid body on SO(3)," in *American Control Conference*, San Francisco, CA, USA, June 2011, pp. 1200–1205.

[102] T. Lee, "Robust global exponential attitude tracking controls on SO(3)," in *American Control Conference*, Washington, DC, USA, June 2013, pp. 2103–2108.

[103] T. Lee, "Global exponential attitude tracking controls on $SO(3)$," *IEEE Transactions on Automatic Control*, vol. 60, no. 10, pp. 2837–2842, February 2015.

[104] J. Wang, I. A. Kapitaniuk, S. A. Chepinskiy, D. Liu, and A. J. Krasnov, "Geometric path following control in a moving frame," *IFAC-PapersOnLine*, vol. 48, no. 11, pp. 150–155, September 2015.

[105] S. Kulumani and T. Lee, "Constrained geometric attitude control on SO(3)," *International Journal of Control, Automation and Systems*, vol. 15, no. 6, pp. 2796–2809, December 2017.

[106] Y. Zhu, X. Chen, and C. Li, "Some discussions about the error functions on SO(3) and SE(3) for the guidance of a UAV using the screw algebra theory," *Advances in Mathematical Physics*, vol. 2017, pp. 1–11, January 2017.

[107] K. A. Ackerman, I. M. Gregory, and N. Hovakimyan, "Flight control methods for multirotor UAS," in *International Conference on Unmanned Aircraft Systems (ICUAS)*, Washington, DC, USA, June 2019, pp. 353–361.

[108] H. K. Khalil, *Nonlinear Systems*. Englewood Cliffs, NJ, USA: Prentice Hall, 2002.

[109] A. Loría and E. Panteley, "Uniform exponential stability of linear time-varying systems: revisited," *Systems & Control Letters*, vol. 47, no. 1, pp. 13–24, September 2002.

[110] E. Panteley and A. Loría, "Uniform exponential stability for families of linear time-varying systems," in *IEEE Conference on Decision and Control*, Sydney, Australia, December 2000, pp. 1948–1953.

[111] D. S. Bernstein, *Matrix Mathematics*, 2nd ed. Princeton, NJ, USA: Priceton University Press, 2009.

[112] K. Fan and A. J. Hoffman, "Some metric inequalities in the space of matrices," *Proceedings of the American Mathematical Society*, vol. 6, no. 1, pp. 111–116, February 1955.

[113] E. D. Sontag and Y. Wang, "On characterizations of the input-to-state stability property," *Systems & Control Letters*, vol. 24, no. 5, p. 351–359, 1995.

[114] M. Greenberg, *Advanced Engineering Mathematics*, 2nd ed. Pearson Education, 2016.