Degree Project in Electrical Engineering

Second cycle, 30 credits

Robust STL Planning Through Interpolating Splines

MATTHEW W. LOCK

Robust STL Planning Through Interpolating Splines

MATTHEW W. LOCK

Master's Programme, Embedded Systems, 120 credits
Date: February 25, 2025

Supervisors: Matti Vahs, Jana Tumova
Examiner: Dimos V. Dimarogonas
School of Electrical Engineering and Computer Science
Swedish title: Robust STL-Planering Genom Interpolerande Splines

Abstract

In autonomous systems, robust spatio-temporal planning is essential for navigating complex environments, where precise spatial trajectories and temporal constraints coordination is critical. Signal Temporal Logic (STL) provides a powerful formalism for defining these constraints, with quantitative semantics offering robustness measures vital for safety-critical and disturbance-resilient applications. However, STL-based planning is computationally intensive, posing challenges for real-time applications, particularly in long-horizon missions.

This thesis works towards addressing this limitation by developing a smooth, spline-based STL encoding for an STL fragment that excludes the until operator and supports single-level recursion, including the eventually-always and always-eventually operators, enabling more efficient trajectory planning. The encoding, designed for a Crazyflie 2.1 quadrotor, leverages rest-to-rest and Catmull-Rom splines, decoupling optimization variable count from STL horizon length to enhance computational efficiency in extended missions. Evaluated across benchmarks, the encoding improved performance over previous STL planning methods, with solve times scaling more efficiently with horizon length. For further validation, we applied the encoding in a numerical simulation of a Crazyflie 2.1 performing a complex reach-avoid task, achieving robust trajectory tracking with improved solve times, though still limited for stringent real-time applications.

Keywords

Signal Temporal Logic (STL), Spatio-temporal planning, Spline-based planning, Crazyflie 2.1

Sammanfattning

I autonoma system är robust spatiotemporal planering, det vill säga planering i tid och rum, avgörande för att navigera i komplexa miljöer, där noggrann koordinering av banor och tidsbegränsningar är kritisk. Signal Temporal Logic (STL) är en kraftfull formalism för att definiera dessa begränsningar, med kvantitativ semantik som erbjuder robusthetsmått som är viktiga för säkerhetskritiska och störningståliga applikationer. STL-baserad planering är dock beräkningsintensiv, vilket innebär utmaningar för realtidsapplikationer, särskilt i uppdrag med lång tidshorisont.

Denna avhandling syftar till att ta itu med denna begränsning genom att utveckla en jämn, splinebaserad STL-kodning för ett STL-fragment som utesluter ”until”-operatorn och stöder rekursion på en nivå, inklusive operatörerna ”eventually-always” och ”always-eventually”, vilket möjliggör effektivare banplanering. Kodningen, som är utformad för en Crazyflie 2.1-drönare med fyra rotorer, utnyttjar ”rest-to-rest”- och ”Catmull-Rom”-splines, som frikopplar antalet optimeringsvariabler från STL-horisontlängden för att förbättra beräkningseffektiviteten vid längre uppdrag. Utvärderat över diverse riktmärken förbättrade kodningen prestandan jämfört med tidigare STL-planeringsmetoder, med lösningstider som skalar mer effektivt med horisontlängden. För ytterligare validering tillämpade vi kodningen i en numerisk simulering av en Crazyflie 2.1 som utförde en komplex ”reach-avoid”-uppgift och uppnådde robust banföljning med förbättrade lösningstider, även om den fortfarande är begränsad för mer krävande realtidsapplikationer.

Nyckelord

Signal Temporal Logic (STL), Spatio-temporal planering, Spline-baserad planering, Crazyflie 2.1

Acknowledgments

As I reflect on my journey over the past few years in Sweden, I find it difficult to fully express my gratitude for the people and experiences that have profoundly shaped me. Completing this chapter of my life would not have been possible without the support, encouragement, and friendship of countless individuals. To each of you who has been part of this journey, I extend my heartfelt thanks for the impact you have had.

Kath, your encouragement at the beginning of this journey was a pivotal for me. Your encouragement and your own determination are what inspired me to take this step. I am endlessly grateful knowing you are always in my corner. Miguel, embarking on this adventure in Sweden with you has been nothing short of extraordinary. Knowing you were just down the hall provided a comforting sense of familiarity, and for that, I am deeply grateful. Emelia, you have been an anchor point for both me and Miguel, offering us a a little sense of home. Thank you for welcoming us to Stockholm with your kindness, generosity, and companionship.

I am also grateful to Ecobloom for providing me with the opportunity to sustain myself while forming meaningful friendships. Max, I look forward to continuing our after-work catch-ups. Márk and Zoë, I cannot thank you enough for your emotional support, thoughtful advice, shared dinners, and countless adventures. To my Embedded Systems friends, you kept me grounded during our time in Kista, and for that, I am immensely appreciative.

Everita, you were the first friend I made here, and I could not have asked for a better person to fill that role. Zach, thank you for the late-night work sessions and adventures—it has been an honour to learn from you and share those experiences. To my other KTH friends, thank you for welcoming me into your little family. Filip, your adventurous spirit has been inspiring, and I have been fortunate to join in some of those experiences. Loizos, your support during our walks and talks has been invaluable. Isabella, I am so lucky and grateful for your empathy, encouragement, and genuine friendship—I hope our future trips include South Africa. Beatriz, your authenticity has always been refreshing, and I look forward to catching up soon. Marco Cella! Thank you for always brightening my days with your humour and many shenanigans.

Jonne, our shared train rides to ABB and the past year of friendship have been a joy. To everyone else who has been part of this journey, thank you for your presence and kindness. To my colleagues at SMaRC, I am grateful for the incredible experiences I have had with the team. Ivan, thank you for giving me the opportunity to contribute and for including me in unforgettable trips. Special shout-out to Sriharsha and Nacho for the memorable moments we shared in Svalbard, and Nacho, thank you for quite literally saving my life during that one near fall from the boat.

I owe a special debt of gratitude to my supervisor, Matti, for his patience, guidance, and unwavering support throughout this project. I am equally thankful to Jana for her constructive feedback and co-supervision.

Lastly, to my family: without your sacrifices, unwavering support, and unconditional love, I would not be where I am today. You mean the world to me.

Stockholm, February 2025
Matthew W. Lock

Contents

List of Figures

List of Tables

List of Acronyms and Abbreviations

CCP	convex-concave procedure
DC	difference of convex
IPOPT	Interior Point Optimizer
LTL	Linear Temporal Logic
MILP	Mixed Integer Linear Programming
MIP	Mixed Integer Programming
MIQP	Mixed Integer Quadratic Programming
MLD	Mixed Logic Dynamical
MPC	Model Predictive Control
MTL	Metric Temporal Logic
PNF	Positive Normal Form
PRM	Probabilistic Roadmaps
RHC	Receding Horizon Controller
RRT	Rapidly-exploring Random Trees
RT-RRT*	Real-time RRT*
SQP	Sequential Quadratic Programming
STL	Signal Temporal Logic
TL	Temporal Logic
UAV	unmanned aerial vehicle

Chapter 1

Introduction

1.1 Motivation

With advancements in computational power and sensory capabilities, robotic systems have emerged as versatile platforms with various applications. These applications often require operating in complex environments where precise coordination under arbitrary spatial and temporal requirements is crucial for successful operation. Consider the example of an autonomous drone tasked with delivering a package in a busy urban environment. The drone must navigate a dynamic landscape filled with tall buildings, other drones, and ground vehicles while adhering to strict delivery deadlines. The challenge is not only to avoid collisions and reach the destination efficiently but also to ensure that the timing of each manoeuvre aligns with external constraints, such as avoiding certain areas at specific times due to pedestrian traffic. This scenario exemplifies the need for spatio-temporal planning, where both spatial trajectories and temporal constraints must be simultaneously optimised to ensure safe and efficient operation.

Consequently, motion planning in the context of robotics and autonomous systems has become a critical and continued area of focus in various fields, including autonomous vehicles [1] and unmanned aerial vehicles (UAVs) [2]. By applying methods for robust obstacle avoidance within dynamic limitations and preferences, agents can navigate complex environments while optimizing objectives like energy consumption, travel time, or robustness. While many popular motion planning methods, including Model Predictive Control (MPC) [3],

optimal control [4], Probabilistic Roadmaps (PRM) [5], and Rapidly-exploring Random Trees (RRT) [6] have been thoroughly investigated and successfully applied, most approaches focus on spatial aspects, such as generating collision-free trajectories. They often neglect the temporal dimensions of planning, such as the precise timing of manoeuvres or maintaining specific temporal sequences. This oversight limits the applicability of these methods in scenarios requiring complex spatio-temporal coordination. As a result, incorporating temporal preferences into these methods typically requires careful, case-by-case adjustments, which may not lead to the most desirable or efficient trajectories under given constraints.

One method for directly capturing arbitrary spatio-temporal requirements is Temporal Logic (TL); a family of formal modelling languages that combine logical predicates with temporal operators to define expected system behaviours, such as the timing, frequency, and importance of task execution. In this work, we utilise Signal Temporal Logic (STL) [7] to formally specify desired agent motion in continuous time. STL includes Robust Semantics [8], allowing it to quantify the degree of satisfaction or violation of a specification, providing a numerical measure of adherence in addition to Boolean evaluation. This ability to measure satisfaction levels allows STL to support flexible adjustments to constraints and quantify robustness, both of which are valuable in safety-critical applications or environments requiring resilience against disturbances and tracking inaccuracies. These features have motivated the direct incorporation of STL into trajectory planning [9, 10] and control synthesis [11, 12] for spatio-temporal missions.
Despite the successes of STL in these areas, a significant research gap remains in developing methods that can handle long-horizon, complex STL specifications in real-time, especially as the complexity of the environment or the mission increases. The primary challenge lies in balancing the computational demands with the need for precise and efficient spatio-temporal planning.

1.2 Aims

This thesis aims to take steps towards addressing the issues surrounding the computational complexity of STL planning. To do this, the work aims to develop an efficient and scalable method for spatio-temporal motion planning for the Crazyflie 2.1 quadcopter under STL constraints. Specifically, the objectives are:

1. **To address long-horizon STL planning**: Develop an encoding of STL specifications using interpolating splines that can handle long-duration missions without computational complexity scaling linearly with mission length, formulating a smooth method suitable for real-time applications.

2. **To implement and tailor the encoding for specific spline classes**: Focus on rest-to-rest and Catmull-Rom splines to generate trajectories that satisfy STL constraints by solving a non-linear optimisation problem, leveraging their properties to improve planning efficiency and trajectory smoothness.

3. **To evaluate the method's performance**: Conduct benchmark scenarios to assess and compare the encoding's computational efficiency, accuracy, and robustness.

4. **To demonstrate practical applicability through simulation**: Apply the encoding to a numerical simulation of a Crazyflie 2.1 quadcopter executing a complex reach-avoid mission, evaluating online replanning capabilities and dynamic feasibility of the generated paths.

1.3 Scope

This work focuses on robust trajectory planning for systems with independent position control under potentially complex STL specifications with long-horizon missions. More specifically, we consider a single-agent Crazyflie 2.1 quadrotor. We limit our consideration to affine predicate functions and support specifications with a single temporal recursion. Lastly, we only consider a fragment of the STL grammar, excluding the *until* operator.

Inspired by [10], we develop a smooth spline-based encoding of the STL specifications to work towards real-time planning. Our encoding removes the dependency link between the number of optimisation variables and the STL horizon length. Unlike previous work that regularly samples the generated spline, we incorporate the inherent properties of the Catmull-Rom spline, such as the convex-hull, directly into the STL encoding for spline generation. The encoding is applied to two interpolating spline classes: (a) rest-to-rest and (b) Catmull-Rom splines. Our encoding does not explicitly maximise the generated trajectory's temporal robustness but does capture inherent temporal aspects. These could be used in future to manipulate the overall temporal robustness but is left as a future extension of this work.

We solve the planning problem for a series of benchmark scenarios to highlight the efficacy of our encoding. We also simulate and experiment with the Crazyflie 2.1 * platform, showing that we can successfully generate and track trajectories to satisfy arbitrary STL specifications.

1.4 Contributions

Our main contributions include:

- Developing an STL encoding for rest-to-rest and Catmull-Rom splines tailored for systems where we approximate independent position control is possible. While the encoding can, in principle, be applied to any class of interpolating spline with the necessary guarantees, our implementation and evaluation focus on these specific systems.
- Improving the efficiency of the approach proposed by [10] by eliminating the linear scaling of the computational domain as the mission duration increases.
- Providing a novel method for encoding singularly *eventually-always* and *always-eventually* operators.
- Assessment of the efficacy and efficiency of the proposed encoding through benchmarks and numerical simulation.

*`www.bitcraze.io/products/crazyflie-2-1/`

1.5 STL Based Planning and Synthesis

TL offers a precise and rigorous framework for characterising the expected behaviour of dynamic systems, making it well-suited for spatial-temporal mission planning. Considering the additional benefits of STL, including application for continuous-time systems as well as robustness quantification, planning under STL specifications has garnered significant attention in the context of autonomous systems tasked with arbitrary spatiotemporal requirements.

While synthesis under STL constraints has been shown for finite-state automata [13] and control-barrier functions [14], these approaches are often confined to fragments of the temporal specifications or require substantial offline pre-computation. Considering that we aim to provide a more generalised encoding without any pre-computation, we focus on presenting related works for three distinct categories of STL planning and control synthesis, which broadly cover larger fragments of the temporal specifications and do not require significant offline pre-computation. Namely these are (a) optimisation-based control synthesis, (b) planning with sampling methods, and (c) planning with splines. Additionally, we briefly discuss proposed encodings that admit additional benefits, such as temporal robustness.

1.5.1 Optimisation-based Synthesis

Mixed Integer Programming

It was first shown in [15] that Mixed Integer Linear Programming (MILP) and Mixed Integer Quadratic Programming (MIQP) techniques could be applied to model checking and optimal control problems for Mixed Logic Dynamical (MLD) systems subject to Linear Temporal Logic (LTL) specifications under a finite horizon assumption. This approach improved efficiency and scalability over abstraction-based approaches by circumventing the computationally expensive construction of finite-state abstractions and Büchi automatons. A novel method for directly encoding LTL specifications as mixed-integer linear constraints was then introduced in [16] to encompass infinite-horizon specifications through the incorporation of loop constraints; enforcing that the trajectory is eventually periodic and broadening the scope of modelling to intriguing

problems such as periodic inspection and persistent monitoring. A further contribution of this work was the reduction of the encoding complexity from quadratic to linear.

Exploiting the improvements in the efficiency of Mixed Integer Programming (MIP) techniques, [17] presents an MPC strategy for discrete-time systems subject to STL specifications. STL specifications are automatically encoded as mixed-integer linear constraints on the system variables, admitting a robust-based encoding of the Quantitative semantics, thereby providing a notion of robust satisfaction of the STL specification. This allows for arbitrary relaxation or hardening of the STL constraints to ensure feasible trajectory generation.

Despite this success, MILP problems have been shown to scale exponentially with the number of binary variables [11, 12]. This makes application to real-time systems challenging and has led to significant effort being put towards reducing the associated problem size and improving computational efficiency. Notably, it was demonstrated in [12] that the number of binary variables needed can be reduced from a linear to a logarithmic dependence on the time horizon through disjunctive programming. The authors effectively demonstrate these improvements for several benchmark specifications, outperforming state-of-the-art encodings for long-time horizons and complex specifications. A heuristic approach presented in [18] finds the minimal number of binary variables needed to generate a satisfying trajectory. After relaxing all the binary variable constraints, the authors iteratively solve a MILP and incrementally reintroduce constraints at the necessary *critical times* for *critical predicates*. Termination of the algorithm occurs when either a satisfying trajectory is found, or the MILP returns an infeasible solution; returning a minimally sized satisfying MILP or a non-feasible solution. The authors incorporate this approach into a Receding Horizon Controller (RHC) framework to perform reactive real-time synthesis for a dynamic operating environment. Naturally, the discretisation period of the dynamical system places a fundamental limit on the necessary control rate. The authors, therefore, apply the iterative optimal RHC control actions before a feasible solution has been found, potentially violating the specification.

While the above-mentioned approaches permit arbitrary combinations of temporal formulas, the predicate functions themselves remain restricted to linear functions in all MILP-based methods. Put more simply, all atomic propositions must be composed of half-spaces. Additionally, problems remain computationally intensive for higher dimensional problems, complex specifications and long time horizons, preventing the adoption of any one superior method for real-time application.

Smooth Approximation

One method developed to alleviate the computational burden of MILP approaches is a smooth approximation of the minimisation and maximisation operators needed for the STL encoding. This effectively allows the problem to be incorporated into general non-linear problems and Sequential Quadratic Programming (SQP) problems, which do not suffer from the same issues regarding complexity exponentially scaling with the number of optimisation variables. Consequently, the expressiveness of STL semantics is retained while avoiding the combinatorial nature of MILP.

The first work to do this was [19] that presented an infinitely differentiable approximation of the robust semantics of arbitrary Metric Temporal Logic (MTL) formula through smooth approximations of the minimum and maximum functions of the encoding. This allows for the application of off-the-shelf gradient descent optimisers to maximise the smooth robustness and robustly satisfy the MTL specifications. A large body of work building on this has since developed. For example, the authors in [10] apply this encoding to perform planning for multi-quadrotor missions in real-time for short mission durations, while the authors of [20] develop a smooth optimal controller using an SQP framework known as SCVvx [21] and introduce a novel encoding of the STL specifications known as the "STL subdynamics". The authors in [22] apply smooth approximation techniques to an alternative formulation of the STL encoding to provide "smooth cumulative quantitative semantics". Unlike previous works that only consider robustness at the most critical time instances, the cumulative semantics maximises both the robustness at critical times and the duration for which the formula is satisfied. In all the works mentioned above, it has been shown that

smooth approximations lead to faster solve times and often yield more robust trajectories compared to MIP and heuristics-based approaches. Furthermore, they admit the use of general non-linear and non-affine predicate functions.

Despite all the advancements, several challenges remain an essential area of focus for ongoing work on smooth approximations. More specifically, current methods tend to scale poorly for complex STL specifications or long time horizons, reducing the effectiveness for real-time application. Moreover, when applied to methods such as SQP, solutions may quickly become infeasible or may be limited to local optima due to the iterative approximations in the SQP procedure [23].

Convex-Concave Procedure

Recognising the shortcomings regarding the scalability of MIP encoding and increased infeasibility when solving SQP problems, some works have explored alternative optimisation techniques. Most notably, [23] develops a framework known as *STLCCP* which explicitly incorporates the structure of STL through decomposition of the STL formulas and conversion of the original program into A difference of convex (DC) program. These quadratic programs are subsequently solved sequentially using the convex-concave procedure (CCP). This framework retains all information of the convex part of each iteration, unlike traditional SQP methods, resulting in fewer approximations. Furthermore, the authors introduce a novel robustness measure more appropriate for application to their framework which utilises a new smoothing function known as the *mellowmin* function. Numerical experiments on several benchmarks demonstrate state-of-the-art robustness maximization and computation time performance. Despite these advances, the solve times for complex missions remain in the tens of seconds, rendering the framework impractical for real-time systems.

1.5.2 Sampling-Based Planning

Sampling-based planning methods differ from traditional optimisation techniques in that they leverage random sampling to explore the solution space. Rather than exhaustively searching for an optimal trajectory, sampling methods strategically sample a subset of solutions. This

stochastic exploration more effectively accommodates high-dimensional state spaces and provides a means to address inherent uncertainties in real-world environments. Furthermore, the explorative nature of these techniques provides a mechanism for finding globally optimal solutions.

Consequently, several works have studied how the quantitative semantics of STL can be incorporated into the cost-function of sampling-based methods to allow for improved STL planning capabilities over optimisation techniques both in terms of computational performance and robustness [24, 25, 26, 9]. For example, [24] first proposes a cost function with user-specified parameters integrated into a modified RRT*. Here, the user-specified spatial preferences are defined through a fragment of STL. At the same time, the associated cost function is shown to trade off the efficiency against the spatial robustness of the generated trajectory by asymptotically minimising the cost function. Similarly, [25] proposes a novel cost function that reflects the robustness of a safety-driven fragment of STL. This is incorporated into a state-of-the-art exploration algorithm to improve autonomous exploration performance while adhering to user-defined specifications and circumvents issues related to conservative obstacle inflation.

An alternative approach in [26] applies specification-based heuristics to sampling-based control synthesis for TL specifications to incrementally synthesise a motion control policy. This is achieved by explicitly using bounds on the quantitative robustness as a heuristic guide, biasing the sampling procedure of an RRT* algorithm towards the "Direction of Increasing Satisfaction". Finally, recognising the relevance of real-time application, the authors in [9] extend the Real-time RRT* (RT-RRT*) algorithm to handle STL specifications through the introduction of a compatible cost function to allow for real-time planning with dynamic obstacles. Their implementation supports a fragment of the STL specifications, including the bounded and unbounded *eventually* and *always* operators. Still, it limits the *until* operator to a time-bounded specification.

1.5.3 Planning with Splines

As previously noted, considerable research efforts have been devoted to addressing the reduction of computational load inherent in STL planning. A novel approach by [10] demonstrates that applying splines to STL planning effectively reduces the number of decision variables needed for optimisation-based methods. Their work showcases a hierarchical control scheme with claimed real-time capability for short-horizon plans by utilising a smoothed robustness measure [19] to perform high-level waypoint generation under STL constraints. By establishing a relationship between spatially relocatable waypoints and a continuous spline trajectory, the authors generate dynamically feasible trajectories with maximal robustness that satisfy the STL specification for short-horizon trajectories. The number of waypoints generated is mission-specific, depending on the horizon length and geometric complexity, requiring apriori knowledge. The authors demonstrate their work on various missions with six waypoints evenly sampled at 1 Hz and do not provide any results showcasing the efficacy for long-duration missions. Furthermore, the application of this approach is constrained to systems exhibiting dynamics analogous to those observed in multi-rotor drones. The authors extend their work in [27] by co-designing a trajectory planner to satisfy the STL specification with a robustness margin large enough to accommodate the tracking error bound.

1.5.4 Temporally Robust Planning

Despite the successful consideration of robust semantics to the STL synthesis and planning problems, robustness measures have until recently been limited to the notion of spatial robustness. The concept of temporal robustness was first introduced by [8] who developed left and right temporal robustness. More plainly, it refers to the amount of time a signal can be shifted forward or backward while adhering to temporal specifications. Through the formulation of a MILP encoding, [28] was the first to provide formal methods for maximising time robustness of STL specifications.

The authors in [29] further refine temporal robustness into *synchronous* and *asynchronous* temporal robustness, whereby individual predicates are either shifted synchronously or asynchronously by an upper

bound limit. The authors present a control synthesis scheme for maximising either left or right temporal robustness through a MILP formulation and successfully demonstrate control synthesis for a multi-agent drone scenario. Their investigations have indicated that while asynchronous robustness provides more utility in enabling arbitrary asynchronous temporal perturbations of the predicates, optimising is more computationally expensive. Left and right temporal robustness is then combined by [30] into a single temporal robustness measure quantifying the maximal time perturbations in either direction that will satisfy the STL specification. Recognising the computational burden of optimising temporal robustness, the authors in [31] propose an efficient synthesis method for generating temporally robust trajectories with specified temporal robustness. Unlike [29], which focuses on asynchronous robustness down to the individual predicate level, the authors instead consider the asynchronous robustness of decoupled "sub-trajectories". Insightfully, they avoid combinatorial exploring all possible sub-trajectories by exploiting redundancies between them and introducing the notion of "instant-shift pairs".

Interestingly, the authors in [32] propose a continuous-time framework leveraging Bézier curve parametrization to maximize asynchronous temporal robustness for multi-agent STL planning. In their approach, trajectories are represented using Bézier curves, which allow for independent parameterization of spatial curvature and time progression. This flexibility enables the generation of smooth, continuous trajectories that inherently satisfy kinematic constraints while maximizing the permissible time shifts agents can handle without violating STL specifications. They formulate the problem as a MILP problem to generate continuous-time trajectories and achieve improved performance over previous works.

The notion of combined temporal and spatial robustness has been explored in [33]. After generating an offline trajectory that is maximally spatially and temporally robust through MILP, an event-triggered MPC scheme is then employed to ensure online satisfaction of an STL specification despite disturbances. The limiting factor in this approach relates to the computational intractability of the MILP for offline planning with solve times greater than one hour.

1.6 Sustainability, Ethics and Society

This project focuses on developing and implementing an efficient trajectory planning method using STL. By optimising the generated trajectories to minimise unnecessary movements and ensure precise spatio-temporal execution, the work contributes to reducing energy consumption in robotic systems, such as drones or autonomous vehicles. This directly aligns with important environmental sustainability goals by lowering the carbon footprint associated with energy-intensive operations, particularly in scenarios such as delivery services or persistent surveillance tasks.

From an economic perspective, the proposed STL-based approach can enhance the scalability and efficiency of trajectory planning, potentially reducing development time and costs for companies deploying autonomous systems. This could make advanced planning technologies more accessible and feasible for widespread adoption in various industries.

Regarding ethics, the work does not pose any inherent risks or threats to society, as it primarily focuses on improving the operational efficiency of autonomous systems. However, it is worth noting that the responsible application of such systems must be considered, for example, in contexts where autonomous technologies raise concerns about surveillance.

1.7 Outline

Section 2 establishes the theoretical background, covering STL, Crazyflie dynamics, motion primitives, and Catmull-Rom splines. Building on this, Section 3 explores the implementation of rest-to-rest spline planning under STL constraints, including problem formulation and encoding strategy. Section 4 extends the approach to Catmull-Rom splines to remove the stop-and-go nature of the rest-to-rest primitives. In Section 5, benchmark tests and numerical simulations capture the encoding's computational efficiency, spatial robustness, and trajectory tracking performance. A discussion follows this in Section 6, which analyses the results and compares the relevant benchmarks to existing methods. The thesis concludes in Section 7 with a summary of contributions and proposed directions for future research.

Chapter 2

Background

This chapter provides a foundational overview necessary for understanding the methodologies discussed later. We start by introducing STL, then provide the utilised dynamic model of the Crazyflie quadrotor, followed by an overview of the minimal-jerk motion primitives and the Catmull-Rom spline needed to develop the proposed spline-based STL encoding.

2.1 Signal Temporal Logic

STL is a temporal logic variant introduced in [7] to monitor and specify the desired behaviour of continuous-time signals. This is achieved by integrating predicates over real-valued, continuous-time functions and temporal operators which can be precisely bounded within specific time intervals. This allows for detailed temporal reasoning about system behaviours.

Let $\boldsymbol{x} : \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ be a continuous-time signal. At the most granular level, a property of the signal can be described by a Boolean predicate π^μ over the signal, taking on a binary truth value evaluated through a continuously differentiable, real-valued predicated function $\mu : \mathbb{R}^n \to \mathbb{R}$ so that for $\boldsymbol{x} \in \mathbb{R}^n$:

$$\pi^\mu := \begin{cases} \top & \text{if } \mu(\boldsymbol{x}) > 0, \\ \bot & \text{if } \mu(\boldsymbol{x}) < 0. \end{cases} \tag{2.1}$$

Definition 2.1.1: STL Grammar [20]

Let ψ be defined by

$$\psi ::= \Diamond_{[a,b]}\ \varphi \mid \Box_{[a,b]}\ \varphi \mid \Box_{[a,b]}\Diamond_{[c,d]}\varphi \mid \Diamond_{[a,b]}\Box_{[c,d]}\varphi, \tag{2.2}$$

where $\varphi ::= \pi^{\mu} \mid \neg\ \pi^{\mu}$ with $\neg$ denoting logical negation, $\Diamond_{[a,b]}$ and $\Box_{[a,b]}$ represent the bounded temporal operators **eventually** and **always**, each subject to a given time frame $[a, b]$ with $a, b \in \mathbb{R}$ and $0 \leq a \leq b \leq \infty$.

Then STL formulas are recursively defined from the grammar:

$$\varphi ::= \ \psi_1 \wedge \psi_2 \mid \psi_1 \vee \psi_2, \tag{2.3}$$

where ψ_1 are ψ_2 are defined by (2.2), and $\wedge$, and $\vee$ denote logical operators **conjunction** (AND) and **disjunction** (OR) respectively.

The **eventually** operator indicates that φ holds at least once within the given interval. In contrast, the **always** operator requires ψ to hold at all times in the given interval. The **until** operator, denoted $\varphi_1\ \mathcal{U}_{[a,b]}\ \varphi_2$, is not explored in this work and requires φ_2 to hold at some time within the interval, and for φ_1 to hold at all times on the interval before φ_1 holds. Using the grammar defined in Definition 2.1.1, STL formulas can be recursively formulated from a set of predicates. Importantly, we only consider formulas in Positive Normal Form (PNF) wherein negation directly precedes predicate functions. Further, in this work, we only consider affine predicate functions of the form $\mu(\boldsymbol{x}) = \boldsymbol{a}^T\boldsymbol{x} + \boldsymbol{b}$ with a maximum of two temporal recursions.

Definition 2.1.2: STL Time-Bound [17]

The time-bound N of a bounded-time formula φ is the maximum over the sums of all nested upper bounds on the temporal operators and provides a conservative maximum trajectory length needed to determine the satisfaction of the formula.

An STL formula is categorised as bounded, or *bounded-time* when it contains no unbounded temporal operators. In such a case, the satisfaction of the STL formula can be established in a finite amount of time. Given a bounded STL formula, the time-bound N can be calculated according to Definition 2.1.2. For example, given the STL formula $\square_{[0,10]}\Diamond_{[1,6]}\ \varphi$, then $N \geq 10 + 6 = 16$.

2.1.1 Boolean Semantics

Informally, Definition 2.1.3 demonstrates that Boolean semantics yield a binary assertion regarding the satisfaction criteria wherein the satisfaction relation $(\boldsymbol{x}, t) \vDash \phi$ indicates that the signal $\boldsymbol{x}$ satisfies ϕ at time t. This simplifies the evaluation of an STL specification, yielding a straightforward yes or no problem of whether the specification is satisfied. This binary outcome, however, also abstracts away the nuances of how robustly the system meets or deviates from the specification.

Definition 2.1.3: Boolean STL Semantics [20]

Formally, the satisfaction of signal $\boldsymbol{x}$ at time t with respect to an STL formula φ is defined as:

$$
\begin{aligned}
(\boldsymbol{x}, t) &\vDash \pi^{\mu} && \Leftrightarrow && \mu(\boldsymbol{x}_t) > 0 \\
(\boldsymbol{x}, t) &\vDash \neg\ \psi && \Leftrightarrow && \neg\ (\boldsymbol{x}, t) \vDash \psi \\
(\boldsymbol{x}, t) &\vDash \varphi \wedge \psi && \Leftrightarrow && (\boldsymbol{x}, t) \vDash \varphi \wedge (\boldsymbol{x}, t) \vDash \psi \\
(\boldsymbol{x}, t) &\vDash \varphi \vee \psi && \Leftrightarrow && (\boldsymbol{x}, t) \vDash \varphi \vee (\boldsymbol{x}, t) \vDash \psi \\
(\boldsymbol{x}, t) &\vDash \Diamond_{[a,b]}\ \varphi && \Leftrightarrow && \exists\ t' \in [t + a, t + b], (\boldsymbol{x}, t') \vDash \varphi \\
(\boldsymbol{x}, t) &\vDash \square_{[a,b]}\ \varphi && \Leftrightarrow && \forall\ t' \in [t + a, t + b], (\boldsymbol{x}, t') \vDash \varphi.
\end{aligned}
$$

Definition 2.1.4: Robust/Quantitative STL Semantics [20]

The robustness measure of signal $\boldsymbol{x}$ at time t is evaluated through a real-valued function ρ^{φ}, recursively defined as:

$$
\begin{aligned}
\rho^{\pi^{\mu}}(\boldsymbol{x}, t) &= \mu(\boldsymbol{x}_t) \\
\rho^{\neg\ \psi}(\boldsymbol{x}, t) &= -\rho^{\psi}(\boldsymbol{x}, t) \\
\rho^{\varphi_1 \wedge \varphi_2}(\boldsymbol{x}, t) &= \min\left(\rho^{\varphi_1}(\boldsymbol{x}, t), \rho^{\varphi_2}(\boldsymbol{x}, t)\right) \\
\rho^{\varphi_1 \vee \varphi_2}(\boldsymbol{x}, t) &= \max\left(\rho^{\varphi_1}(\boldsymbol{x}, t), \rho^{\varphi_2}(\boldsymbol{x}, t)\right) \\
\rho^{\Diamond_{[a,b]}\psi}(\boldsymbol{x}, t) &= \max_{t' \in [t+a, t+b]} \rho^{\psi}(\boldsymbol{x}, t') \\
\rho^{\Box_{[a,b]}\psi}(\boldsymbol{x}, t) &= \min_{t' \in [t+a, t+b]} \rho^{\psi}(\boldsymbol{x}, t').
\end{aligned}
$$

2.1.2 Quantitative Semantics

Quantitative semantics [8] defined in Definition 2.1.4, also known as *robust* semantics, provide a real measure of the extent of satisfaction or violation of STL formulas. This is achieved through evaluation of a real-valued function $\rho^{\varphi} : \mathbb{R}^n \to \mathbb{R}$ such that $(\boldsymbol{x}, t) \vDash \varphi \equiv \rho^{\varphi}(\boldsymbol{x}, t) > 0$. Simply put, the sign of ρ^{φ} denotes boolean satisfaction while the magnitude denotes the satisfaction margin. This is particularly useful to provide a concise measure of the system's robustness against uncertainties.

Example 2.1.1: Spatial Robustness of a Scalar Signal

Consider a dynamical system with a scalar state variable x and control input u. Given $\varphi := x > 5$ and $\mu(x_t, u_t) = x - 5$, a robustness measure of 3 indicates that the system can tolerate disturbances of up to three units while satisfying φ. Conversely, a value of -2 suggests that the system violates the condition by two units. It should be noted that in the case that $\rho^{\varphi}(x, t) = 0$, the outcome is inconclusive.

2.1.3 Smooth Semantics

An alternative infinitely differentiable approximation of the max and min operators, known as the *log-sum exponential* functions, can be used in place of the *min* and *max* operators. Applying this approximation to the

robust STL semantics leads to the notion of *smooth semantics* [19] whose advantages are discussed in section 1.5.1.

Definition 2.1.5: Smooth Max Min Operators [19]

The smooth approximations of the m-ary maximum and minimum operators for $k \geq 1$ are given by :

$$\widetilde{\max}(a_1, ..., a_m) := \frac{1}{k} \ln \left(e^{ka_1} + ... + e^{ka_m}\right) \tag{2.4}$$

$$\widetilde{\min}(a_1, ..., a_m) := -\widetilde{\max}(-a_1, ..., -a_m) \tag{2.5}$$

As discussed in [19], it can be shown that for any set $\{a_i\}_{i=1}^m$, the lower and upper bounds of the approximation errors are given by

$$0 \leq \widetilde{\max}(a_1, ..., a_m) - \max(a_1, ..., a_m) \leq \frac{\ln(m)}{k} \tag{2.6}$$

$$0 \leq \widetilde{\min}(a_1, ..., a_m) - \min(a_1, ..., a_m) \leq \frac{\ln(m)}{k}, \tag{2.7}$$

where the maximum error occurs where all values a_i are equal, independent of the magnitude of a_i. It is easily shown that the error tends to zero as $k \to \infty$. Subsequently, we can introduce the notion of smooth semantics.

Definition 2.1.6: Smooth STL Semantics [19]

Following Definition 2.1.4, the smooth robustness measure $\tilde{\rho}^{\varphi}$ is obtained by substituting all min and max operators in the robust semantics with the smooth $\widetilde{\min}$ and $\widetilde{\max}$ operators. The approximation error in (2.6) and (2.7) leads to the smooth robustness error $\epsilon = |\tilde{\rho}^{\varphi} - \rho^{\varphi}|$. Boolean satisfaction of an STL formula can be defined in terms of the smooth robustness measure $\tilde{\rho}^{\varphi}$ such that $(\boldsymbol{x}, t) \vDash \varphi \equiv \tilde{\rho}^{\varphi}(\boldsymbol{x}, t) > \epsilon$.

2.1.4 Temporal Robustness

Another dimension of STL robustness is *temporal robustness* [8], [29], which seeks to quantify the robustness of a signal against time shifts in the predicates of the STL formula. This helps to provide guarantees for systems subject to timing uncertainties. This section considers both the notions of *synchronous* and *asynchronous* left and right temporal robustness, referring to the synchronous and asynchronous nature of the time shifts of the predicates. We briefly state the results presented in [29] and refer the reader to the original works for full treatments.

Definition 2.1.7 first defines the characteristic function $\mathcal{X}_\varphi(\mathbf{x}, t)$ to characterise pointwise satisfaction and violation of the STL formula.

Definition 2.1.7: Characteristic Function [29]

The characteristic function $\mathcal{X}_\varphi(\mathbf{x}, t) : X^{\mathbb{T}} \times T \to \{\pm 1\}$ of an STL formula φ relative to a signal $\mathbf{x}$ at time t is recursively defined as

$$
\begin{aligned}
\mathcal{X}_p(\mathbf{x}, t) &:= \operatorname{sign}(\mu(\boldsymbol{x}(t))) \\
\mathcal{X}_{\neg\varphi}(\mathbf{x}, t) &:= -\mathcal{X}_\varphi(\mathbf{x}, t) \\
\mathcal{X}_{\varphi_1 \wedge \varphi_2}(\mathbf{x}, t) &:= \mathcal{X}_{\varphi_1}(\mathbf{x}, t) \sqcap \mathcal{X}_{\varphi_2}(\mathbf{x}, t) \\
\mathcal{X}_{\varphi_1 \vee \varphi_2}(\mathbf{x}, t) &:= \mathcal{X}_{\varphi_1}(\mathbf{x}, t) \sqcup \mathcal{X}_{\varphi_2}(\mathbf{x}, t) \\
\mathcal{X}_{\Diamond_{[a,b]}\varphi}(\mathbf{x}, t) &:= \bigsqcup_{t' \in [t+a, t+b]} \mathcal{X}_\varphi(\mathbf{x}, t') \\
\mathcal{X}_{\Box_{[a,b]}\varphi}(\mathbf{x}, t) &:= \bigsqcap_{t' \in [t+a, t+b]} \mathcal{X}_\varphi(\mathbf{x}, t')
\end{aligned}
$$

where $\sqcup$ is the supremum operator and $\sqcap$ the infimum operator, $\mathbb{T} := \mathbb{R}$ denotes the time domain of the signal, and $X^{\mathbb{T}}$ the set of all signals $\boldsymbol{x} : \mathbb{T} \to \mathbb{R}^n$. The signal $\boldsymbol{x}$ satisfies φ at time t when $\mathcal{X}_p(\mathbf{x}, t) = 1$, while $\mathcal{X}_p(\mathbf{x}, t) = -1$ indicates that $\boldsymbol{x}$ does not satisfy φ at time t.

Informed by Definition 2.1.7, the satisfaction $\mathcal{X}_\varphi(\mathbf{x}, t)$ of the signal $\boldsymbol{x}$ over the formula φ is recursively defined through the characteristic functions $\mathcal{X}_{p_i}(\mathbf{x}, t)$ of each predicate $p_i \in \mathcal{AP}$ contained in φ. Consequently, it is shown in [29] that satisfaction of φ subject to time perturbations in $\boldsymbol{x}$ is equivalent to shifting the predicates via the characteristic functions.

Synchronous Temporal Robustness

Regarding predicate shifts, synchronous temporal robustness refers to synchronous time shifts of all predicates $p_i \in \mathcal{AP}$ contained in φ. Although less general than asynchronous temporal robustness, the synchronous temporal robustness of a signal is shown to be the upper bound of its asynchronous temporal robustness [29]. In its essence, it quantifies the maximal amount of time by which we can shift a signal $\boldsymbol{x}$, or each predicate in φ synchronously, such that φ remains satisfied at t.

With respect to an STL formula φ, left and right synchronous temporal robustness $\eta_\varphi^\pm(\mathbf{x}, t) : X^\mathbb{T} \times \mathbb{T} \to \bar{\mathbb{T}}$ of a signal $\mathbf{x}$ at time t is defined as

$$\eta_\varphi^+ := \mathcal{X}_\varphi(\mathbf{x}, t) \cdot \sup\{\tau \geq 0 : \forall t' \in [t, t+\tau], \mathcal{X}_\varphi(\mathbf{x}, t') = \mathcal{X}_\varphi(\mathbf{x}, t)\} \quad (2.8)$$

$$\eta_\varphi^- := \mathcal{X}_\varphi(\mathbf{x}, t) \cdot \sup\{\tau \geq 0 : \forall t' \in [t-\tau, t], \mathcal{X}_\varphi(\mathbf{x}, t') = \mathcal{X}_\varphi(\mathbf{x}, t)\} \quad (2.9)$$

where $\bar{\mathbb{T}} = \mathbb{T} \cup \{\pm\infty\}$. In shorthand for both left and right synchronous temporal robustness, it is written

$$\eta_\varphi^\pm := \mathcal{X}_\varphi(\mathbf{x}, t) \cdot \sup\{\tau \geq 0 : \forall t' \in t \pm [0, \tau], \mathcal{X}_\varphi(\mathbf{x}, t') = \mathcal{X}_\varphi(\mathbf{x}, t)\} \quad (2.10)$$

It follows that for a signal $\boldsymbol{x} \in X^\mathbb{T}$, $\tau \in \mathbb{T}_{\geq 0}$, and a set of predicates $\mathcal{AP}$, that the signal $\boldsymbol{x}^{+\tau}$ is synchronously τ-*early* (left shifted) if

$$\forall p_i \in \mathcal{AP}, \forall t \in \mathbb{T}, \qquad \mathcal{X}_{p_i}(\boldsymbol{x}^{+\tau}, t) = \mathcal{X}_{p_i}(\boldsymbol{x}, t+\tau). \quad (2.11)$$

Similarly, the signal $\boldsymbol{x}^{-\tau}$ is synchronously τ-*late* (right shifted) if

$$\forall p_i \in \mathcal{AP}, \forall t \in \mathbb{T}, \qquad \mathcal{X}_{p_i}(\boldsymbol{x}^{-\tau}, t) = \mathcal{X}_{p_i}(\boldsymbol{x}, t-\tau). \quad (2.12)$$

Theorem 2.1.1: Synchronous Temporal Robustness

Given an STL formula φ built upon the predicate set $\mathcal{AP}$, and a signal $\boldsymbol{x} : \mathbb{R} \to X$ it holds for any $t \in \mathbb{R}$ and $r \in \bar{R}_{\geq 0}$ that

$$|\eta^\pm(\boldsymbol{x}, t)| = r \Leftrightarrow \forall \tau \in [0, r), \mathcal{X}_\varphi(\boldsymbol{x}^{\pm\tau}, t) = \mathcal{X}_\varphi(\boldsymbol{x}, t) \text{ and} \quad (2.13)$$

$$r < \infty \Rightarrow \forall \epsilon > 0, \exists \tau \in [r, r+\epsilon), \mathcal{X}_\varphi(\boldsymbol{x}^{\pm\tau}, t) \neq \mathcal{X}_\varphi(\boldsymbol{x}, t). \quad (2.14)$$

Asynchronous Temporal Robustness

In the more general case, the left and right asynchronous temporal robustness quantifies the maximal amount of time by which any individual predicate $p_i \in \mathcal{AP}$ contained in φ can be shifted asynchronously left and right, respectively.

With respect to an STL formula φ, left and right asynchronous temporal robustness $\theta^{\pm}_{\varphi}(\mathbf{x}, t) : X^{\mathbb{T}} \times \mathbb{T} \to \bar{\mathbb{T}}$ of a signal $\mathbf{x}$ at time t is recursively defined starting from a predicate p as

$$\theta^{+}_{p} := \mathcal{X}_p(\mathbf{x}, t) \cdot \sup\{\tau \geq 0 : \forall t' \in [t, t+\tau], \mathcal{X}_p(\mathbf{x}, t') = \mathcal{X}_p(\mathbf{x}, t)\} \quad (2.15)$$

$$\theta^{-}_{p} := \mathcal{X}_p(\mathbf{x}, t) \cdot \sup\{\tau \geq 0 : \forall t' \in [t-\tau, t], \mathcal{X}_p(\mathbf{x}, t') = \mathcal{X}_p(\mathbf{x}, t)\} \quad (2.16)$$

Applying the recursive operator rules similar to those found in Definition 2.1.7 then leads to

$$\theta^{\pm}_{\neg\varphi}(\mathbf{x}, t) := -\theta^{\pm}_{\varphi}(\mathbf{x}, t) \quad (2.17)$$

$$\theta^{\pm}_{\varphi_1 \wedge \varphi_2}(\mathbf{x}, t) := \theta^{\pm}_{\varphi_1}(\mathbf{x}, t) \sqcap \theta^{\pm}_{\varphi_2}(\mathbf{x}, t) \quad (2.18)$$

$$\theta^{\pm}_{\varphi_1 \mathcal{U}_{[a,b]} \varphi_2}(\mathbf{x}, t) := \bigsqcup_{t' \in [t+a, t+b]} \left(\theta^{\pm}_{\varphi_2}(\mathbf{x}, t) \bigsqcap_{t'' \in [t, t')} \theta^{\pm}_{\varphi_1}(\mathbf{x}, t'') \right). \quad (2.19)$$

For a signal $\boldsymbol{x} \in X^{\mathbb{T}}$, a set of predicates $\mathcal{AP} := (p_1, ..., p_L)$, a set of shifts $\bar{\tau} := (\tau_1, ..., \tau_L)$ with $\tau_1, ...\tau_L \in \mathbb{T}_{\geq 0}$, the signal $\boldsymbol{x}^{+\bar{\tau}}$ is asynchronously τ-*early* (left shifted) if

$$\forall p_i \in \mathcal{AP}, \forall t \in \mathbb{T}, \qquad \mathcal{X}_{p_i}(\boldsymbol{x}^{+\bar{\tau}}, t) = \mathcal{X}_{p_i}(\boldsymbol{x}, t + \tau_i). \quad (2.20)$$

Similarly, the signal $\boldsymbol{x}^{-\bar{\tau}}$ is asynchronously τ-*late* (right shifted) if

$$\forall p_i \in \mathcal{AP}, \forall t \in \mathbb{T}, \qquad \mathcal{X}_{p_i}(\boldsymbol{x}^{-\bar{\tau}}, t) = \mathcal{X}_{p_i}(\boldsymbol{x}, t - \tau_i). \quad (2.21)$$

In contrast to the case of synchronous robustness, we note here that the characteristic function $\mathcal{X}_{p_i}(\boldsymbol{x}, t + \tau_i)$ of each predicate p_i can be individually shifted through τ_i. The authors in [29] present the conditions under which time shifts of the predicates correspond with time shifts in the underlying comments of the signal.

Theorem 2.1.2: Asynchronous Temporal Robustness

Given an STL formula φ over the predicate set $\mathcal{AP} := \{p_1, ..., p_L\}$, and a signal $\boldsymbol{x} : \mathbb{R} \to X$ it holds for any $t \in \mathbb{R}$ and $r \in \bar{R}_{\geq 0}$ that

$$|\theta^{\pm}(\boldsymbol{x}, t)| = r \Leftrightarrow \forall \tau_1, ..., \tau_L \in [0, r), \mathcal{X}_{\varphi}(\boldsymbol{x}^{\pm\bar{\tau}}, t) = \mathcal{X}_{\varphi}(\boldsymbol{x}, t), \quad (2.22)$$

where $\bar{\tau} := (\tau_1, ...\tau_L)$.

Combined Left and Right Temporal Robustness

For both the synchronous and asynchronous temporal robustness presented above, it is essential to note the directionality of the temporal perturbation. More plainly, the given definition for temporal robustness does not hold when considering cases where some predicates are shifted left while others are shifted right. Although not explicitly dealt with in this work, we note the current works into *combined left and right* robustness to quantify the amount of permissible time perturbation of predicates in both directions. The curious reader is referred to [30] for more information.

2.2 Crazyflie Dynamics

In modelling the Crazyflie quadrotor, we adopt the model established in [34]. The world and body fixed frames are denoted as $\mathcal{F}_W$ and $\mathcal{F}_B$, respectively, and are illustrated in Figure 2.1. $\mathcal{F}_B$ is represented by axes x_B, y_B, z_B, whereby x_B is the preferred forward orientation and z_B is perpendicular to the plane containing the propellers pointing upward. Z-X-Y Euler angles define the roll ϕ, pitch θ, and yaw ψ between $\mathcal{F}_W$ and $\mathcal{F}_B$. The rotation matrix $\boldsymbol{R}_B^W$ between $\mathcal{F}_W$ and $\mathcal{F}_B$ is defined as

$$\boldsymbol{R}_B^W = \begin{bmatrix} c_\psi c_\theta - s_\phi s_\psi s_\theta & -c_\phi s_\psi & c_\psi s_\theta + c_\theta s_\phi s_\psi \\ c_\theta s_\psi + c_\psi s_\phi s_\theta & c_\phi s_\psi & s_\psi s_\theta - c_\psi c_\theta s_\phi \\ -c_\phi s_\theta & s_\phi & c_\phi c_\theta \end{bmatrix} \quad (2.23)$$

where s_ϕ and c_ϕ denote $\sin(\phi)$ and $\cos(\phi)$ respectively, and similarly for θ and ψ.

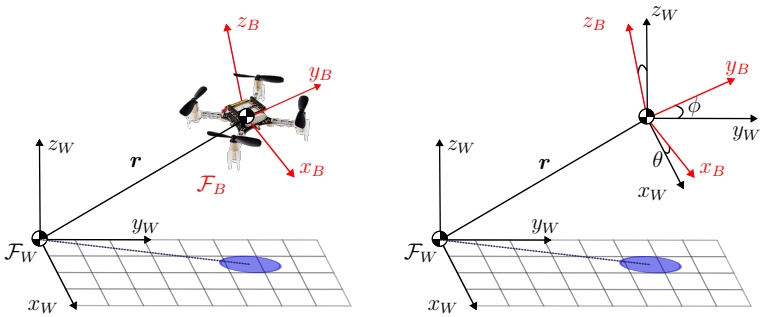


Figure 2.1: Crazyflie quadcopter shown in the world coordinate frame $\mathcal{F}_W$ and body-fixed coordinate frame $\mathcal{F}_B$, along with the Euler angles.

It can consequently be shown that the system dynamics are given by

$$\begin{cases} \dot{\boldsymbol{r}} = \boldsymbol{v} & (2.24) \\ \dot{\boldsymbol{v}} = \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \frac{1}{m} \boldsymbol{R}_B^W \begin{bmatrix} 0 \\ 0 \\ \sum_{i=1}^{4} F_i \end{bmatrix} & (2.25) \\ \dot{\boldsymbol{R}}_B^W = \boldsymbol{R}_B^W [\boldsymbol{\Omega}]_\times & (2.26) \\ \boldsymbol{J}\dot{\boldsymbol{\Omega}} = \boldsymbol{M} - [\boldsymbol{\Omega}]_\times \times \boldsymbol{J}[\boldsymbol{\Omega} & (2.27) \end{cases}$$

where $\boldsymbol{r}$ is the position of the quadcopter's centre-of-mass in $\mathcal{F}_W$, F_i is the force generated by propeller i, $\boldsymbol{\Omega}$ and $\boldsymbol{M}$ are the angular velocity and torque of the quadrotor in $\mathcal{F}_B$, while m and $\boldsymbol{J}$ denote the mass and inertial matrix. Applying the small angle approximation, the dynamics are simplified according to [35] to give

$$\begin{aligned} \ddot{r}_x^{des} &= g(\theta^{des} \cos(\psi) + \phi^{des} \sin(\psi)), \\ \ddot{r}_y^{des} &= g(\theta^{des} \sin(\psi) - \phi^{des} \cos(\psi)), \\ \ddot{r}_z^{des} &= \frac{F^{des}}{m} - g. \end{aligned} \tag{2.28}$$

where F^{des} is the desired thrust as the sum of all the individual propeller forces, g is the gravitational acceleration, θ^{des} and ϕ^{des} represent the desired pitch and roll angles. The desired yaw is assumed to be zero. By inverting (2.28), θ^{des}, ϕ^{des} and F^{des} can be represented in terms of the

desired acceleration in each axis. The resulting dynamics are given by

$$\begin{aligned} \phi^{des} &= \frac{1}{g}(\ddot{r}_x^{des}\sin(\psi) - \ddot{r}_y^{des}\cos(\psi)), \\ \theta^{des} &= \frac{1}{g}(\ddot{r}_x^{des}\cos(\psi) + \ddot{r}_y^{des}\sin(\psi)), \\ F^{des} &= m(\ddot{r}_3^{des} + g). \end{aligned} \tag{2.29}$$

Finally, by noticing that the desired accelerations in each axis of (2.29) can be used as direct control input $\boldsymbol{u} = [\ddot{r}_x^{des}\ \ddot{r}_y^{des}\ \ddot{r}_z^{des}]^T$, it is remarked that the system can be modelled as a second-order integrator with

$$\begin{bmatrix} \dot{\boldsymbol{r}} \\ \ddot{\boldsymbol{r}} \end{bmatrix} = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{r} \\ \dot{\boldsymbol{r}} \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \boldsymbol{u}. \tag{2.30}$$

2.3 Motion Primitives

In the search for a computationally efficient method for dynamically feasible trajectory generation for quadcopters, [36] proposes a set of motion primitives derived from solving independent minimum jerk optimal control problems along three orthogonal axes. The resulting trajectory, represented by a spline $\boldsymbol{s}(t) : [0, T_f] \to \mathbb{R}^3$, takes the form :

$$\begin{bmatrix} p^*(t) \\ v^*(t) \\ a^*(t) \end{bmatrix} = \begin{bmatrix} \frac{\alpha}{120}t^5 + \frac{\beta}{24}t^4 + \frac{\gamma}{6}t^3 + a_0t^2 + v_0t + p_0 \\ \frac{\alpha}{24}t^4 + \frac{\beta}{6}t^3 + \frac{\gamma}{2}t^2 + a_0t + v_0t \\ \frac{\alpha}{6}t^3 + \frac{\beta}{2}t^2 + \gamma t + a_0t \end{bmatrix}, \tag{2.31}$$

where $p(t), v(t), a(t)$ represent position, velocity, and acceleration along a given axis. α, β, and γ are scalar linear functions of the initial condition s_0 and desired final condition s_f, depending on the desired motion characteristics. The reader is referred to [36] for a comprehensive exploration of these motion characteristics.

2.3.1 Rest-to-Rest Primitives

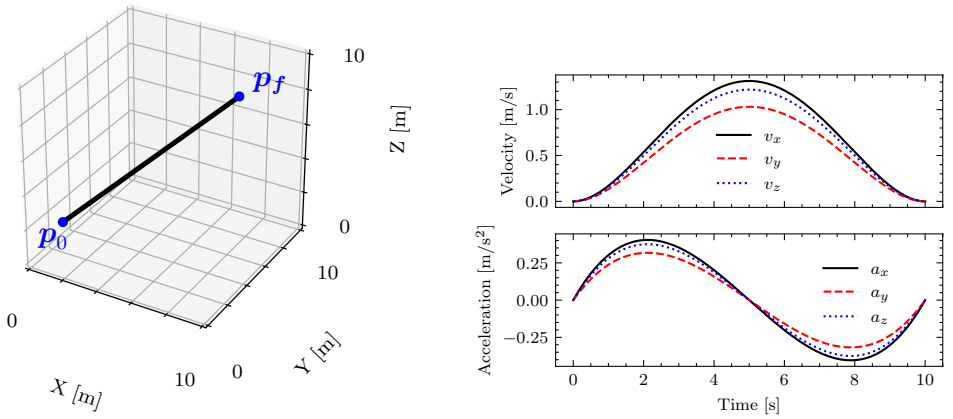
We consider fully defined end transitional states with a desired end position, velocity, and acceleration given by $s(T_f) = (p_f, v_f, a_f)$ along each axis. In this case, the scalar constants α, β, and γ are given by

$$\begin{bmatrix}\alpha\\ \beta\\ \gamma\end{bmatrix} = \frac{1}{T_f^5}\begin{bmatrix}720 & -360T_f & 60T_f^2\\ -360T_f & 168T_f^2 & -24T_f^3\\ 60T_f^2 & -24T_f^3 & 3T_f^4\end{bmatrix}\begin{bmatrix}\Delta p\\ \Delta v\\ \Delta a\end{bmatrix}, \tag{2.32}$$

with $\Delta p = p_f - p_0, \Delta p = v_f - v_0$, and $\Delta a = a_f - a_0$. More specifically, we are interested in the case where $v_0 = v_f = a_0 = a_f = 0$ such that the system under consideration starts at rest from $s_0 = (p_0, 0, 0)$ and comes to a complete stop at $t = T_f$ with $s(T_f) = (p_f, 0, 0)$. Consequently, the expressions of the scalar constants are given by

$$\begin{bmatrix}\alpha\\ \beta\\ \gamma\end{bmatrix} = \frac{1}{T_f^5}\begin{bmatrix}720\Delta p\\ -360T_f\Delta p\\ 60T_f^2\Delta p\end{bmatrix}. \tag{2.33}$$

An example of the rest-to-rest motion is illustrated in Figure 2.2 with $p_0 = (1, 1.5, 2), p_f = (8, 7, 8.5)$, and $T_f = 10$. The generated velocity and acceleration profiles are given in Figure 2.2(b).



(a) Position in $\mathbb{R}^3$ (b) Velocity and acceleration profile

Figure 2.2: Illustration of (a) position and (b) the associated velocity and acceleration profiles for a rest-to-rest motion primitive trajectory with $p_0 = (1, 1.5, 2), p_f = (8, 7, 8.5)$, and $T_f = 10$.

2.3.2 Feasibility Guarantees

The particular closed-form expression of these rest-to-rest motion primitives admits several feasibility guarantees. For the sake of brevity, applicable results of these guarantees are briefly stated here with a complete treatment found in the original work [36].

Proposition 2.3.1: Position Feasibility

s demonstrated in [36], rest-to-rest primitives follow a straight line segment between the initial and final positions. Consequently, for a convex set $\mathcal{A}$, if $p_0, p_f \in \mathcal{A}$, then $p(t') \in \mathcal{A}$ for all $0 \leq t' \leq T_f$. In other words, any rest-to-rest primitive that starts and ends within $\mathcal{A}$ remains entirely within $\mathcal{A}$ throughout its trajectory. This result follows directly from the definition of a convex set.

Proposition 2.3.2: Velocity Feasibility

For rest-to-rest primitives of duration T_f, the maximum velocity along a particular axis occurs at time $t = \frac{T_f}{2}$ with a value of

$$\max_{t\in[0,T_f]} v(t) = \frac{15}{8}\frac{\Delta p}{T_f}. \tag{2.34}$$

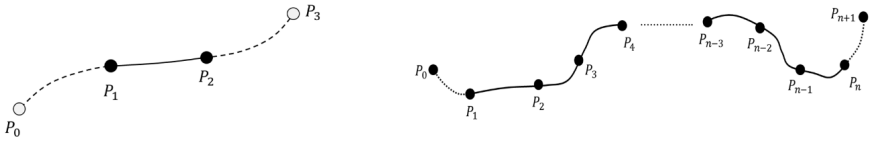
Proposition 2.3.3: Input Feasibility

hen considering the acceleration profile of rest-to-rest trajectories along a single axis, it can be shown that the extrema $a_1 = a_2$ lie at times $t = (\frac{1}{2} \pm \frac{\sqrt{3}}{6})T_f$, giving a maximum acceleration value of

$$\max_{t\in[0,T_f]} a(t) = |a_1| = |a_2| = \left|\pm\frac{10\sqrt{3}\Delta p}{3T_f^2}\right|. \tag{2.35}$$

2.4 Catmull-Rom Spline

First introduced in [37], Catmull-Rom splines are a family of piecewise cubic interpolating splines with smooth interpolation between control points, otherwise known as *knots*. Catmull-Rom splines are commonly used in computer graphics for modelling and animation [38], and, although not widely used in robotics, have some recent applications in robotic manipulator trajectory planning [39], and RRT* path planning for nonholonomic mobile robots [40].



(a) Catmull-Rom spline segment (b) N segment Catmull-Rom spline

Figure 2.3: Catmul-Rom (a) segment and (b) multi-segment curve.

As depicted in Figure 2.3(a), a single segment of the Catmull-Rom spline is defined by four control points $P_0, ..., P_3$, with the curve passing through P_1 and P_2, and virtual control points, P_0 and P_3. Segments are cascaded to generate a curve passing through N control points, as in Figure 2.3(b), requiring $N+2$ total control points. This can be reduced to N points by constraining the two virtual control points to mirror their second adjacent control points about the adjacent control point. In Figure 2.3(b), this would mean constraining control points control points P_0 and P_{n+1} to be the mirror of P_2 and P_{n-1} about P_1 and P_n.

Importantly, the Catmull-Rom spline provides $\boldsymbol{C}^1$ continuity, meaning that no discontinuities exist in the first derivative (velocity) profile, and local support, ensuring that the shape of each line segment is only affected by neighbouring control points. While alternative splines with similar properties have been successfully applied in the context of motion planning [41], [42], the Catmull-Rom spline avoids the tedious task of defining two virtual control points per segment required by the Bezier spline or selecting a collinear arrangement of consecutive tangents required by the Hermite spline.

2.4.1 Uniform Parametrisation

Consider a single Catmull-Rom segment where the position at time t is denoted $P(t)$. In the uniform formulation where $0 \leq t_k \leq 1$, position and velocity constraints are given by

$$P(0) = P_1, P(1) = P_2 \tag{2.36}$$

$$\dot{P}(0) = \alpha(P_2 - P_0), \dot{P}(1) = \alpha(P_3 - P_1) \tag{2.37}$$

where $\dot{P}(0)$ and $\dot{P}(1)$ are determined from the vector difference between neighbouring control points. The scalar parameter $0 \leq \alpha \leq 1$ is known as the "tension" and affects the curvature of the generated curve.

Following this, the cubic polynomial expression for the position as a function of time can be given in matrix form

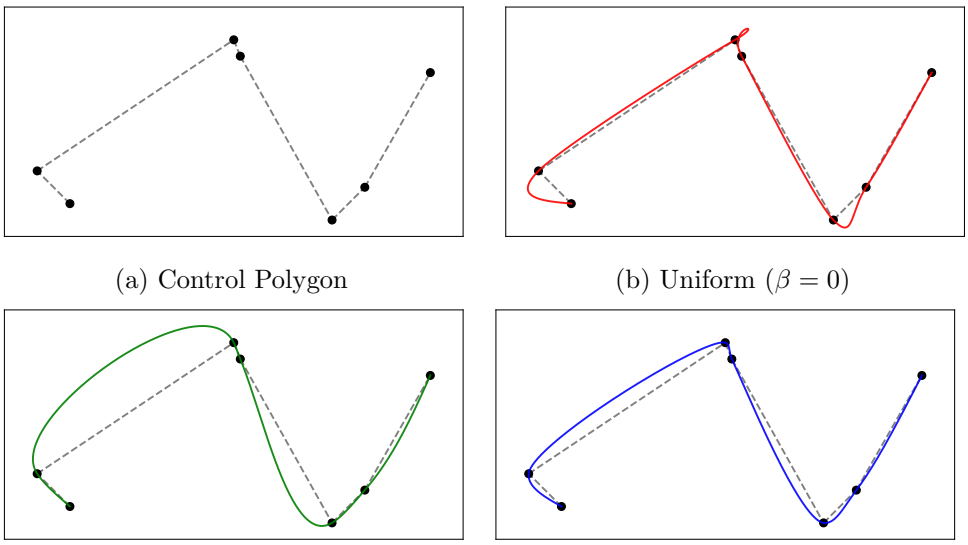
$$P(t) = \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\alpha & 0 & \alpha & 0 \\ 2\alpha & \alpha - 3 & 3 - 2\alpha & -\alpha \\ -\alpha & 2 - \alpha & \alpha - 2 & \alpha \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix}, \tag{2.38}$$

forming a special case of a Cardinal spline [37] where explicitly setting $\alpha = 1/2$ results in the uniform Catmull-Rom spline. The velocity profile $\dot{P}(t)$ and acceleration profile $\ddot{P}(t)$ for $0 \leq t \leq 1$ is trivially found through differentiating $P(t)$ with respect to time.
For a cascaded sequence of segments, the time t_i at which each control point is reached is defined by (2.39) with $t_0 = 0$.

$$t_{i+1} = t_i + 1 \tag{2.39}$$

2.4.2 Non-Uniform Parametrisation

While uniform parametrisation provides a powerful and simple method for path generation, it offers little flexibility. More importantly, uniform parametrisation can lead to cusp and self-intersections within line segments, as shown in Figure 2.4(b). This is considered undesirable as self-intersections may induce to large accelerations generated by a small radius of curvature, leading to dynamically infeasible paths.



(a) Control Polygon (b) Uniform ($\beta = 0$)

(c) Chordal ($\beta = 1$) (d) Centripetal ($\beta = 0.5$)

Figure 2.4: Catmull-Rom splines generated with the same set of (a) control points using different parameterizations: (b) uniform, (c) chordal, and (d) centripetal. Adapted from [43].

An alternative non-uniform parametrisation is produced through the geometric relationship between control points

$$t_{i+1} = t_i + ||P_{t_{i+1}} - P_{t_i}||^{\beta}, \ (0 \leq \beta \leq 1). \tag{2.40}$$

such that the time to traverse between control points is proportional to the distance between them. Varying the scalar value β gives parametrisations, with different behaviours as depicted in Figure 2.4, including the chordal parametrisation with $\beta = 1$ and the centripetal parametrisation with $\beta = 0.5$. Setting $\beta = 0$ removes the geometric encoding and collapses into the uniform parametrisation.

The expression $P(t)$ governing position at time t between control points P_1 and P_2 [44] is given by

$$P(t) = \frac{t_2 - t}{t_2 - t_1} B_1(t) + \frac{t - t_1}{t_2 - t_0} B_2(t), \ (t_1 \leq t \leq t_2) \tag{2.41}$$

where the parameters $B_1(t)$ and $B_2(t)$ are a function of time determined by positions $P_0, ..., P_3$ and times $t_0, ..., t_3$ expressed as

$$B_1(t) = \frac{t_2 - t}{t_2 - t_0} A_1(t) + \frac{t - t_0}{t_2 - t_0} A_2(t) \tag{2.42}$$

$$B_2(t) = \frac{t_3 - t}{t_3 - t_1} A_2(t) + \frac{t - t_1}{t_3 - t_1} A_3(t) \tag{2.43}$$

$$A_1(t) = \frac{t_1 - t}{t_1 - t_0} P_o + \frac{t - t_0}{t_1 - t_0} P_1 \tag{2.44}$$

$$A_2(t) = \frac{t_2 - t}{t_2 - t_1} P_1 + \frac{t - t_1}{t_2 - t_1} P_2 \tag{2.45}$$

$$A_3(t) = \frac{t_3 - t}{t_3 - t_2} P_2 + \frac{t - t_2}{t_3 - t_2} P_3. \tag{2.46}$$

Centripetal Parametrisation

In this work, we adopt the centripetal parametrisation of the Catmull-Rom spline characterised by (2.40) and (2.41) with $\beta = 1/2$.

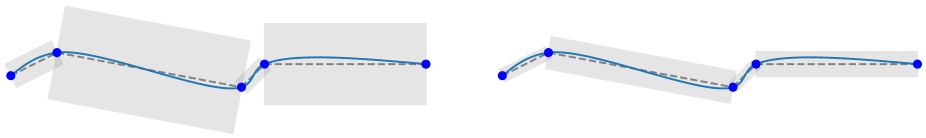
Notably, [38] shows that the centripetal parameterisation is the only parameterisation not producing cusps or self-intersections, relieving any concerns of large accelerations. Moreover, the author shows that for the centripetal parameterisation, the distance h between a curve segment and the line segment containing P_1 and P_2 is bounded by a fraction of the edge length between P_1 and P_2, dependent on the ratio of the distance between P_1 to P_2, and P_1 to P_0.

$$h = ||P_2 - P_1|| \frac{r^{\frac{1}{2}}}{4(1 + r^{\frac{1}{2}})}, \text{ and } r = \frac{||P_1 - P_0||}{||P_2 - P_1||} \tag{2.47}$$

This can further be reduced to a fraction independent of r whereby the distance h between a curve segment and the line segment containing P_1 and P_2 is entirely upper bounded by a fraction of the edge length between P_1 and P_2, given by

$$h = \frac{1}{4} ||P_2 - P_1||. \tag{2.48}$$

Consequently, the bound of a spline segment in a centripetal Catmull-Rom Spline is given by Theorem 2.4.1. Bounding volumes of the same control polygon is illustrated in Figure 2.5 for both (2.47) and (2.48).



(a) Corresponding line segment (b) Neigbouring line segments

Figure 2.5: Bounding volumes of centripetal Catmull-Rom computed using (a) corresponding line segment lengths as defined in (2.48) and (b) neighbouring line segments as defined in (2.47). Notice how using neighbouring line segments results in tighter bounds. Adapted from [43].

Theorem 2.4.1: Bound of Centripetal Catmull-Rom Spline [43]

For a Catmull-Rom curve with centripetal parameterization, each spline segment can be entirely bounded by bounding boxes extruded in the perpendicular direction of its line segment with a distance determined by (2.47) or (2.48).

Chapter 3

Rest-to-Rest Planning

In this chapter, we propose an STL encoding to generate a sequence of dynamically feasible rest-to-rest motion primitives that satisfy STL specifications in continuous time. The proposed method draws inspiration from [10], which utilises regular sampling of continuous-time motion primitives to provide continuous-time guarantees over a set of discrete trajectory samples. Instead, we directly account for continuous time satisfaction in our encoding, considering the position, velocity, and acceleration of guarantees offered by rest-to-rest motion primitives.

3.1 Problem Formulation

Consider a sequence of $N \geq 2$ positional waypoints $(\boldsymbol{x}_0, ..., \boldsymbol{x}_{N-1}) \in \mathbb{R}^3$ and a corresponding sequence of $N-1$ rest-to-rest motion primitives connecting the waypoints. We define $\hat{\sigma}(t) := (\sigma_i)_{i=1}^{N-1}$ as a continuous-time trajectory composed of sequential rest-to-rest motion primitives $\sigma_i(t) : [0, T_{f,i}] \to \mathbb{R}^3$, defined by (2.31) and (2.33), such that

$$\hat{\sigma}(t) = \begin{cases} \sigma_1(t), \ 0 \leq t \leq T_{f,1} \\ \sigma_2(t), \ T_{f,1} \leq t \leq T_{f,1} + T_{f,2} \\ \vdots \\ \sigma_{N-1}(t), \ \sum_{i=0}^{N-2} T_{f,i} \leq t \leq \sum_{i=0}^{N-2} T_{f,i} + T_{f,N-1} \end{cases} . \tag{3.1}$$

Keep in mind that each motion primitive comprises three independent rest-to-rest trajectories for each axis denoted $\sigma_i^{(j)}(t)$ for $j = 1, 2, 3$. The sequence of rest-to-rest primitives is dynamically feasible if the velocity and acceleration profiles of each primitive remain bounded with $\underline{v} \leq \dot{\sigma}^{(j)}(t) \leq \bar{v}$ and $\underline{a} \leq \ddot{\sigma}^{(j)}(t) \leq \bar{a}$. Lastly, the cumulative trajectory length T of $\hat{\sigma}(t)$ is given by $T = \sum_{j=1}^{N-1} T_{f,j}$.

Now that we have established $\hat{\sigma}(t)$, we state the generalised problems of generating a sequence of rest-to-rest motion primitives that satisfy an STL formula φ in Problem 3.1.1 and Problem 3.1.2.

Problem 3.1.1: Boolean Rest-to-Rest STL Planning

Given a bounded STL specification φ and initial state $\boldsymbol{x}_0 \in \mathbb{R}^3$, generate a dynamically feasible trajectory $\hat{\sigma}(t)$ composing a set of rest-to-rest motion primitives such that $\hat{\sigma}(t) \vDash \varphi$. To achieve this, we use smooth robust semantics and solve the following problem:

$$\tilde{p}^{\varphi}(\hat{\sigma}(t)) \geq \epsilon \tag{3.2a}$$
$$\text{s.t } \hat{\sigma}(0) = \boldsymbol{x}_0, \tag{3.2b}$$
$$\underline{v} \leq \dot{\hat{\sigma}}^{(j)}(t) \leq \bar{v} \text{ for } j = 1, 2, 3, \tag{3.2c}$$
$$\underline{a} \leq \ddot{\hat{\sigma}}^{(j)}(t) \leq \bar{a} \text{ for } j = 1, 2, 3, \tag{3.2d}$$

Problem 3.1.2: Robust Rest-to-Rest STL Planning

Given a bounded STL specification φ and initial state $\boldsymbol{x}_0 \in \mathbb{R}^3$, generate a dynamically feasible trajectory $\hat{\sigma}(t)$ composing a set of rest-to-rest motion primitives that maximises satisfaction of the STL smooth robust semantics through the following problem :

$$\max_{\boldsymbol{x}_0,\ldots,\boldsymbol{x}_{N-1},T_{f,1}\ldots,T_{f,N-1}} \tilde{p}^{\varphi}(\hat{\sigma}(t)) \tag{3.3a}$$
$$\text{s.t } \hat{\sigma}(t) = \boldsymbol{x}_0, \tag{3.3b}$$
$$\underline{v} \leq \dot{\hat{\sigma}}^{(j)}(t) \leq \bar{v} \text{ for } j = 1, 2, 3, \tag{3.3c}$$
$$\underline{a} \leq \ddot{\hat{\sigma}}^{(j)}(t) \leq \bar{a} \text{ for } j = 1, 2, 3, \tag{3.3d}$$
$$\tilde{p}^{\varphi}(\hat{\sigma}(t)) \geq \epsilon. \tag{3.3e}$$

Constraints (3.2c - 3.2d) and (3.3c-3.3d) define the conditions for dynamic feasibility through placing upper and lower bounds on the velocity and acceleration in each axis. Constraints (3.2a) and (3.3e) provide lower bounds for the approximation error of the smooth semantics [10], [19] to ensure Boolean satisfaction.

3.2 STL Encoding

One challenge associated with solving Problem 3.1.1 and Problem 3.1.2 is the fact that we can only evaluate robustness at discrete time samples. To circumvent this issue, [10] ensures that the continuous-time trajectory is sampled regularly to satisfy a suitable stricter version φ_s of φ. To generate a sequence of rest-to-rest trajectories $\hat{\sigma}(t)$ that satisfies φ in continuous time, we instead rely on a specific strategy for placement of the rest-to-rest primitive final positions $p_{f,i}$ (waypoints) and selection of trajectory lengths $T_{f,i}$.

With this in mind, we introduce the notion of control points $\boldsymbol{q} \in \mathbb{R}^3$, analogous to the terminology used in the context of splines where the ordered sequence of control points $\mathcal{Q} = (\boldsymbol{q}_0, ..., \boldsymbol{q}_{N-1})$ represents the positions visited in order by the sequence of rest-to-rest primitives. Associated with each control point $\boldsymbol{q}_i$ is the time t_i at which the waypoint is reached with $\hat{\sigma}(t_i) = \boldsymbol{q}_i$. Putting this together, we say that the trajectory $\hat{\sigma}(t)$ is parameterised by $\mathbf{z} = ((\boldsymbol{q}_0, t_0), ..., (\boldsymbol{q}_{N-1}, t_{N-1}))$.

3.2.1 Boolean Semantics

Starting with Boolean satisfaction, we define an STL encoding in terms of $\mathbf{z}$ for both Boolean satisfaction and spatial robustness in continuous time. Before introducing the encoding, for the sake of notation, we introduce $\mathcal{SP}$ and $\mathcal{TP}$ as the ordered sets of *spatial* and *temporal pairs*

$$\mathcal{SP} = \{(\boldsymbol{q}_0, \boldsymbol{q}_1), ..., (\boldsymbol{q}_{N-1}, \boldsymbol{q}_N)\} \text{ where } |\mathcal{SP}| = N - 1 \text{ and,} \tag{3.4}$$

$$\mathcal{TP} = \{(t_0, t_1), ..., (t_{N-1}, t_N)\} \text{ where } |\mathcal{TP}| = N - 1. \tag{3.5}$$

Eventually $\Diamond_{[a,b]}\ \psi$

Noticing the satisfaction criteria of the eventually operator $\Diamond_{[a,b]}\ \psi$ given in Definition 2.1.3, it stands to reason that the existence of control point $\boldsymbol{q}_i$ satisfying ψ such that $a \leq t_i \leq b$ would satisfy the specification. Therefore, we define a condition for Boolean satisfaction of $\Diamond_{[a,b]}\psi$ as

$$\exists \boldsymbol{q}_i \in \mathcal{Q} \text{ s.t } \boldsymbol{q}_i \vDash \psi, a \leq t_i \leq b \Rightarrow \hat{\sigma}(t) \vDash \Diamond_{[a,b]}\psi. \tag{3.6}$$

Notice that the criteria imposed on t_i can be captured by the conjunction of two predicate functions

$$t_i \vDash \varphi^{[a,b]} \Leftrightarrow (\mu^{a\leq t}(t_i) \geq 0) \wedge (\mu^{t\leq b}(t_i) \geq 0), \tag{3.7}$$

$$\text{where } \mu^{a\leq t}(t) = t - a,\ \mu^{t\leq b}(t) = b - t. \tag{3.8}$$

We can therefore further condense the Boolean satisfaction of $\Diamond_{[a,b]}\ \psi$ to

$$\exists\ \boldsymbol{q}_i \in \mathcal{Q}, (\boldsymbol{q}_i \vDash \psi \wedge t_i \vDash \varphi^{[a,b]}) \Rightarrow \hat{\sigma}(t) \vDash \Diamond_{[a,b]}\ \psi. \tag{3.9}$$

Always $\Box_{[a,b]}\ \psi$

Consider the always operator $\Box_{[a,b]}\ \psi$ where ψ defines some position constraint within a convex space $\mathcal{A}$. Given the position feasibility guarantee of rest-to-rest motion primitives previously discussed in Section 2.3.2, it easily follows that for some ordered sequence of control points $\mathcal{SP}' = \{(\boldsymbol{q}_k, \boldsymbol{q}_{k+1}), ..., (\boldsymbol{q}_{M-1}, \boldsymbol{q}_M)\} \subseteq \mathcal{SP}$ where $\forall \boldsymbol{q}_i \in \mathcal{SP}', \boldsymbol{q}_i \vDash \psi$ and $t_k \leq a$, $t_M \geq b$, that $\hat{\sigma}(t) \vDash \Box_{[a,b]}\psi$. More plainly, a sequence of rest-to-rest trajectories spanning over $[a, b]$ will always remain within $\mathcal{A}$ during $a \leq t \leq b$ provided that all control points are constrained within $\mathcal{A}$. To ensure the existence of such an ordered sequence, we write that

$$\begin{aligned}
&\Big(\exists(\boldsymbol{q}_i, \boldsymbol{q}_{i+1}) \in \mathcal{SP} \text{ s.t } (\boldsymbol{q}_i \vDash \psi) \wedge (\boldsymbol{q}_{i+1} \vDash \psi) \wedge (t_i \leq a) \wedge (t_{i+1} \geq a)\Big) \wedge \\
&\quad \Big(\exists(\boldsymbol{q}_j, \boldsymbol{q}_{j+1}) \in \mathcal{SP} \text{ s.t } (\boldsymbol{q}_j \vDash \psi) \wedge (\boldsymbol{q}_{j+1} \vDash \psi) \wedge (t_j \leq b) \wedge (t_{j+1} \geq b)\Big) \wedge \\
&\quad\quad \Big(\forall \boldsymbol{q_k} \in \{\boldsymbol{q}_l \in \mathcal{Q} \mid a \leq t_l \leq b\}, \boldsymbol{q_k} \vDash \psi\Big) \Rightarrow \hat{\sigma}(t) \vDash \Box_{[a,b]}\psi.
\end{aligned} \tag{3.10}$$

When the always operator spans the entire STL bound with $[a, b] = [0, T]$,

Boolean satisfaction is simply given by

$$\forall \boldsymbol{q}_i \in \mathcal{Q}, \boldsymbol{q}_i \vDash \psi \Rightarrow \hat{\sigma}(t) \vDash \square_{[0,T]}\psi. \tag{3.11}$$

Eventually-Always $\lozenge_{[a,b]}\square_{[0,\tau]}\psi$

Having developed an encoding for the **eventually** and **always** operators, we can proceed to develop an encoding for the recursively defined **eventually-always** operator denoted $\lozenge_{[a,b]}\square_{[0,\tau]}\ \psi$.

Through the application of Definition 2.1.3, the boolean satisfaction of $\lozenge_{[a,b]}\square_{[0,\tau]}\ \psi$ for an arbitrary signal $\boldsymbol{x}$ is given by

$$\begin{aligned}(\boldsymbol{x}, 0) \vDash \lozenge_{[a,b]}\square_{[0,\tau]}\psi \ \Leftrightarrow\ & \exists\ t'_k \in [a, b], \\ & \forall t''_k \in [t'_k, t'_k + \tau], (\boldsymbol{x}, t''_k) \vDash \psi.\end{aligned} \tag{3.12}$$

This can be written as

$$(\boldsymbol{x}, 0) \vDash \lozenge_{[a,b]}\square_{[0,\tau]}\psi \ \Leftrightarrow \forall t''_k \in [t'_k, t'_k + \tau], (\boldsymbol{x}, t''_k) \vDash \psi, \tag{3.13a}$$

$$a \leq t'_k \leq b. \tag{3.13b}$$

Noticing the equivalence of (3.13a) to Boolean satisfaction of the always operator in Definition 2.1.3, the criteria for Boolean satisfaction of an eventually-always operator in a rest-to-rest sequence can be written as

$$\begin{aligned}&\Big(\exists(\boldsymbol{q}_i, \boldsymbol{q}_{i+1}) \in \mathcal{SP} \text{ s.t. } (\boldsymbol{q}_i \vDash \psi) \wedge (\boldsymbol{q}_{i+1} \vDash \psi) \wedge \\ &\qquad\qquad (t_i \leq t'_k) \wedge (t_{i+1} \geq t'_k)\Big) \wedge \\ &\Big(\exists(\boldsymbol{q}_j, \boldsymbol{q}_{j+1}) \in \mathcal{SP} \text{ s.t. } (\boldsymbol{q}_j \vDash \psi) \wedge (\boldsymbol{q}_{j+1} \vDash \psi) \wedge \\ &\qquad\qquad (t_j \leq t'_k + \tau) \wedge (t_{j+1} \geq t'_k + \tau)\Big) \wedge \\ &\quad\Big(\forall \boldsymbol{q_k} \in \{\boldsymbol{q}_l \in \mathcal{Q} \mid t'_k \leq t_l \leq t'_k + \tau\}, \boldsymbol{q_k} \vDash \psi\Big) \wedge \\ &\qquad\Big(a \leq t'_k \leq b\Big) \Rightarrow \hat{\sigma}(t) \vDash \lozenge_{[a,b]}\square_{[0,\tau]}\psi.\end{aligned} \tag{3.14a}$$

For a given STL specification, the number of eventually-always operators that appear in the specification is denoted $N^{\lozenge\square}$.

Always-Eventually $\square_{[a,b]}\lozenge_{[0,\tau]}\psi$

The last temporal operator for which we develop an encoding is the always-eventually operator $\square\lozenge\varphi$ used for persistence specifications.

According to Definition 2.1.3, the criteria for Boolean satisfaction of a signal $\boldsymbol{x}$ according for $\square\lozenge\psi$ is given by

$$\begin{aligned}(\boldsymbol{x},0) \vDash \square_{[a,b]}\lozenge_{[0,\tau]}\psi \;\Leftrightarrow \forall\; t'_k \in [a,b],\\ \exists t''_k \in [t'_k, t'_k+\tau], (\boldsymbol{x}, t''_k) \vDash \psi.\end{aligned} \tag{3.15}$$

The continuous condition placed on t'_k and the discrete existence condition of t''_k makes this a particularly challenging operator to encode. We rather relax the operator and provide an over-approximation that is sufficient, but not necessary, to satisfy $\square\lozenge\psi$. For example, consider that $(\boldsymbol{x},0) \vDash \square_{[a,b+\tau]}\ \psi \Rightarrow (\boldsymbol{x},0) \vDash \square_{[a,b]}\lozenge_{[0,\tau]}\psi$. In other words, the always operator satisfies the always-eventually requirement but drastically reduces the solution space. We suggest a less restrictive over-approximation by regularly applying the eventually operator over equally sized partitions of the time interval $[a,b]$.

For the always-eventually operator $\square_{[a,b]}\lozenge_{[0,\tau]}\ \varphi$ where $b-a$ is the duration of the always operator, let $N' := 2\frac{b-a}{\tau}$ be the number of equally sized partitions of $[a,b]$. For simplicity, we only consider the case $\frac{b-a}{\tau} \in \mathbb{Z}^+$. Thus, the set of partitions $\mathcal{P} \in \mathbb{R}^{2N'}$ is given by

$$\mathcal{P} = \left\{[a+0\frac{\tau}{2}, a+1\frac{\tau}{2}], [a+1\frac{\tau}{2}, a+2\frac{\tau}{2}], .., [a+(N'-1)\frac{\tau}{2}, b]\right\}. \tag{3.16}$$

The extended set $\mathcal{P}' \in \mathbb{R}^{2N'+1}$ includes an additional partition such that

$$\mathcal{P}' = \left\{[a+0\frac{\tau}{2}, a+1\frac{\tau}{2}], .., [a+(N'-1)\frac{\tau}{2}, b], [b, b+\frac{\tau}{2}]\right\}. \tag{3.17}$$

Then $\forall t'_k \in [a,b], \exists (a',b') \in \mathcal{P}'$ s.t $t'_k \leq a'$ and $t'_k+\tau \geq b'$. In other words, for every point t'_k there exists a partition entirely spanned by $[t'_k, t'_k+\tau]$. If there exists a point in each partition that satisfies ψ, we can write that

$$\begin{aligned}\forall (a_i, b_i) \in \mathcal{P}, \exists t' \in [a_i, b_i] \text{ s.t } (\boldsymbol{x}, t') \vDash \psi \Rightarrow \forall\ t'_k \in [a, b], \\ \exists t''_k \in [t'_k, t'_k + \tau], (\boldsymbol{x}, t''_k) \vDash \psi.\end{aligned} \tag{3.18}$$

Subsequently, Boolean satisfaction of $\square_{[a,b]}\Diamond_{[0,\tau]}\psi$ is defined by

$$\bigwedge_{j=1}^{2N'+1} \left(\exists \boldsymbol{q}_i \in \mathcal{Q} \text{ s.t } (\boldsymbol{q}_i \vDash \psi) \wedge (t_i \vDash \varphi^{[a+(j-1)\frac{\tau}{2}, a+j\frac{\tau}{2}]+\tau'}) \right) \Rightarrow \hat{\sigma}(t) \vDash \square_{[a,b]}\Diamond_{[0,\tau]}\psi. \tag{3.19}$$

The number of always-eventually operators that appear in a given STL specification is denoted $N^{\square\Diamond}$.

3.2.2 Robust Semantics

Following the boolean satisfaction introduced above, we present the robust encoding of each operator to evaluate the level of satisfaction or violation of the sequence of rest-to-rest trajectories according to the STL specification.

Eventually $\Diamond_{[a,b]}\ \psi$

Using the encoding for smooth STL semantics of logical operators introduced in Definition 2.1.6, we define the smooth robust semantics for the eventually operator of a rest-to-rest sequence $\tilde{\rho}^{\Diamond_{[a,b]}\psi}(\boldsymbol{z})$ as

$$\tilde{\rho}^{\Diamond_{[a,b]}\psi}(\boldsymbol{z}) = \tilde{\bigsqcup_{\boldsymbol{q}_i \in \mathcal{Q}}} \left(\mu^{\psi}(\boldsymbol{q}_i) \tilde{\sqcap} \varphi^{[a,b]}(t_i) \right), \tag{3.20}$$

where the infimum operator is replaced by the smooth minimum function in Definition 2.1.5. Given the induced approximation error of ϵ, we have that $\tilde{\rho}^{\square_{[a,b]}\psi}(\boldsymbol{z}) \geq \epsilon \Rightarrow \hat{\sigma}(t) \vDash \square_{[a,b]}\psi$.

Always $\square_{[a,b]}\ \psi$

Following the logic of Section 3.2.1, smooth robust semantics for the always operator of a rest-to-rest sequence $\tilde{\rho}^{\square_{[a,b]}\psi}(\boldsymbol{z})$ is given by

$$\tilde{\rho}^{\square_{[a,b]}\psi}(\boldsymbol{z}) = \varphi_1 \ \tilde{\sqcap}\ \varphi_2 \ \tilde{\sqcap} \varphi_3 \tag{3.21}$$

$$
\begin{aligned}
\varphi_1 &= \tilde{\bigsqcup_{(\boldsymbol{q}_i,\boldsymbol{q}_{i+1})\in\mathcal{SP}}} \left(\mu^{\psi}(\boldsymbol{q}_i) \,\tilde{\sqcap}\, \mu^{\psi}(\boldsymbol{q}_{i+1}) \,\tilde{\sqcap}\, \varphi^{t\leq a}(t_i) \,\tilde{\sqcap}\, \varphi^{t\geq a}(t_{i+1})\right), \\
\varphi_2 &= \tilde{\bigsqcup_{(\boldsymbol{q}_j,\boldsymbol{q}_{j+1})\in\mathcal{SP}}} \left(\mu^{\psi}(\boldsymbol{q}_j) \,\tilde{\sqcap}\, \mu^{\psi}(\boldsymbol{q}_{j+1}) \,\tilde{\sqcap}\, \varphi^{t\leq b}(t_j) \,\tilde{\sqcap}\, \varphi^{t\geq b}(t_{j+1})\right), \\
\varphi_3 &= \tilde{\prod_{\boldsymbol{q}_k\in\mathcal{Q}}} \left(\mu^{\psi}(\boldsymbol{q}_k) \,\tilde{\sqcap}\, \varphi^{[a,b]}(t_k)\right) \,\tilde{\sqcup}\, \mu^{t\leq a}(t_k) \,\tilde{\sqcup}\, \mu^{t\geq b}(t_k).
\end{aligned}
\tag{3.22}
$$

The additional supremum operators in φ_3 have been introduced to discard any control points which do not lie on the time frame $[a, b]$.

It is important to note that the temporal aspects have now been directly included in the smooth encoding. Therefore, the smooth rest-to-rest encoding no longer strictly represents the *spatial* robustness, but rather a combined *spatial-temporal* robustness. In order to offset this effect, we multiply any temporal predicate by some large scalar value M so that the *spatial-temporal* robustness cannot be upper-bounded by any temporal predicate. This way, the new *spatial-temporal* robustness can be used as a proxy for the true spatial robustness measure.

A Note on Safety

In safety-critical applications, it is typical to specify safety through a negation of the always operator [12] such as $\varphi = \square_{[0,T]}\neg\phi$. Here, the predicate ϕ may for example denote a convex obstacle $\mathcal{O}$ where

$$
\phi = x \geq 2 \wedge x \leq 4 \wedge y \geq 2 \wedge y \leq 4. \tag{3.23}
$$

Unfortunately, the negation of the predicate results in a *safe set* which is no longer convex upon immediate, and for which rest-to-rest primitives does not provide any guarantees. Following the PNF formulation and apply De Morgan's laws, the negated condition $\neg\phi$ is given by

$$
\begin{aligned}
\neg\phi &= \neg(x \geq 2) \vee \neg(x \leq 4) \vee \neg(y \geq 2) \vee \neg(y \leq 4) \\
\neg\phi &= (x \leq 2) \vee (x \geq 4) \vee (y \leq 2) \vee (y \geq 4)
\end{aligned}
\tag{3.24}
$$

As a result, the negated predicate can be expressed as disjunction of convex sets $\mathcal{R}_1 = (x \leq 2)$, $\mathcal{R}_2 = (x \geq 4)$, $\mathcal{R}_2 = (y \leq 2)$ and $\mathcal{R}_4 = (y \geq 4)$ such that

$$\neg\phi = \mathcal{R}_1 \vee \mathcal{R}_2 \vee \mathcal{R}_3 \vee \mathcal{R}_4 \tag{3.25}$$

Given that we now have a set of convex regions, we can provide an encoding that ensures that we never enter the obstacle $\mathcal{O}$ provided that the always operator is defined over the entire mission duration.

More specifically, we require that each spatial pair must be contained within one of the convex regions, explicitly written as

$$\begin{aligned}\tilde{\rho}^{\Box_{[0,T]}\neg\phi}(\boldsymbol{z}) = \tilde{\prod_{(\boldsymbol{q}_i,\boldsymbol{q}_{i+1})\in\mathcal{SP}}} & \Big(\mu^{\mathcal{R}_1}(\boldsymbol{q}_i) \,\tilde{\sqcap}\, \mu^{\mathcal{R}_1}(\boldsymbol{q}_{i+1})\Big) \,\tilde{\sqcup} \\ & \Big(\mu^{\mathcal{R}_2}(\boldsymbol{q}_i) \,\tilde{\sqcap}\, \mu^{\mathcal{R}_2}(\boldsymbol{q}_{i+1})\Big) \,\tilde{\sqcup} \\ & \Big(\mu^{\mathcal{R}_3}(\boldsymbol{q}_i) \,\tilde{\sqcap}\, \mu^{\mathcal{R}_3}(\boldsymbol{q}_{i+1})\Big) \,\tilde{\sqcup} \\ & \Big(\mu^{\mathcal{R}_4}(\boldsymbol{q}_i) \,\tilde{\sqcap}\, \mu^{\mathcal{R}_4}(\boldsymbol{q}_{i+1})\Big).\end{aligned} \tag{3.26}$$

Eventually-Always $\Diamond_{[a,b]}\Box_{[0,\tau]}\psi$

Remembering the equivalence highlighted in the Boolean satisfaction of the **eventually-always** operator, we propose a similar encoding $\tilde{\rho}^{\Diamond_{[a,b]}\Box_{[0,\tau]}\psi}(\boldsymbol{z})$ that is written as

$$\tilde{\rho}^{\Diamond_{[a,b]}\Box_{[0,\tau]}\psi}(\boldsymbol{z}) = \phi_1 \,\tilde{\sqcap}\, \phi_2 \,\tilde{\sqcap}\, \phi_3 \tag{3.27}$$

$$\begin{aligned}\phi_1 &= \tilde{\bigsqcup_{(\boldsymbol{q}_i,\boldsymbol{q}_{i+1})\in\mathcal{SP}}} \Big(\mu^{\psi}(\boldsymbol{q}_i) \,\tilde{\sqcap}\, \mu^{\psi}(\boldsymbol{q}_{i+1}) \,\tilde{\sqcap}\, \varphi^{t\leq t'_k}(t_i) \,\tilde{\sqcap}\, \varphi^{t'_k\leq t}(t_{i+1})\Big) \\ \phi_2 &= \tilde{\bigsqcup_{(\boldsymbol{q}_j,\boldsymbol{q}_{j+1})\in\mathcal{SP}}} \Big(\mu^{\psi}(\boldsymbol{q}_j) \,\tilde{\sqcap}\, \mu^{\psi}(\boldsymbol{q}_{j+1}) \,\tilde{\sqcap}\, \varphi^{t\leq t'_k+\tau}(t_j) \,\tilde{\sqcap}\, \varphi^{t'_k+\tau\leq t}(t_{j+1})\Big) \\ \phi_3 &= \tilde{\prod_{\boldsymbol{q}_k\in\mathcal{Q}}} \Big(\mu^{\psi}(\boldsymbol{q}_k) \,\tilde{\sqcap}\, \varphi^{[t'_k,t'_k+\tau]}(t_k)\Big) \,\tilde{\sqcup}\, \mu^{t\leq t'_k}(t_k) \,\tilde{\sqcup}\, \mu^{t\geq t'_k+\tau}(t_k).\end{aligned} \tag{3.28}$$

such that $t_k + a \leq t'_k \leq t_k + b$. Going forward, we introduce an auxiliary variable τ' representing t'_k.

Always-Eventually $\square_{[a,b]}\Diamond_{[0,\tau]}\psi$

The robust encoding of the **always-eventually** operator $\tilde{\rho}^{\square_{[a,b]}\Diamond_{[0,\tau]}\psi}(\boldsymbol{z})$ remains similar to (3.19) with the addition of the auxiliary variable τ'' such that $0 \leq \tau'' \leq \frac{\tau}{2}$, leading to the encoding

$$\begin{aligned}\tilde{\rho}^{\square_{[a,b]}\Diamond_{[0,\tau]}\psi}(\boldsymbol{z}) &= \tilde{\bigsqcup_{j=1,\dots,j=2N'+1}} \Big(\psi_1 \,\tilde{\sqcap}\, \psi_2\Big), \\ \psi_1 &= \tilde{\bigsqcup_{\boldsymbol{q}_i \in \mathcal{Q}}} \mu^{\psi}(\boldsymbol{q}_i), \\ \psi_2 &= \varphi^{[a+(j-1)\frac{\tau}{2}, a+j\frac{\tau}{2}]+\tau''}(t_i).\end{aligned} \tag{3.29}$$

3.2.3 Planning with Optimisation

With the robust encoding developed, we proceed to solve the planning problem using optimization techniques. We append the auxiliary states to obtain $\tilde{\boldsymbol{z}} = [\boldsymbol{z}, \tau'_1, ..., \tau'_{N^{\Diamond\square}}, \tau''_1, ..., \tau''_{N^{\square\Diamond}}] \in \mathbb{R}^{4N+N^{\Diamond\square}+N^{\square\Diamond}}$. We aim to maximise the trajectory's spatial robustness while satisfying the STL specifications and dynamic constraints. This leads to the following optimization problem, which we solve using Interior Point Optimizer (IPOPT).

Given an STL specification φ time bounded by T and initial state $\boldsymbol{x}_0 \in \mathbb{R}^3$, maximisation of spatial robustness when following a sequence of $N-1$ rest-to-rest motion primitives is given by

$$\max_{\tilde{\boldsymbol{z}}} \ \tilde{\rho}^{\varphi}(\tilde{\boldsymbol{z}}) \tag{3.30a}$$

$$\text{s.t.} \ \left|\frac{15}{8}\frac{\Delta p_i^{(j)}}{T_{f,i}}\right| \leq \bar{v} \ \forall i = 1, .., N-1, j \in \{1,2,3\} \tag{3.30b}$$

$$\left|\frac{10\sqrt{3}\Delta p^{(j)}}{3{T_{f,i}}^2}\right| \leq \bar{a} \ \forall i = 1, .., N-1, j \in \{1,2,3\} \tag{3.30c}$$

$$0.1 \leq T_{f,i} \leq T \ \forall i = 1, ..., N-1, \tag{3.30d}$$

$$a_i \leq t'_i \leq b_i \ \forall i = 1, ..., N^{\Diamond\square}, \tag{3.30e}$$

$$0 \leq \tau'_i \leq \tau_1 \ \forall i = 1, ..., N^{\square\Diamond}, \tag{3.30f}$$

$$\boldsymbol{z}_0 = (\boldsymbol{x}_0^{(1)}, \boldsymbol{x}_0^{(2)}, \boldsymbol{x}_0^{(3)}, 0) \tag{3.30g}$$

$$\tilde{\rho}^{\varphi}(\tilde{\boldsymbol{z}}) \geq \epsilon \tag{3.30h}$$

where $\Delta p_i^{(j)} = \boldsymbol{x}_i^{(j)} - \boldsymbol{x}_{i-1}^{(j)}$ is the distance between the i^{th} and $i^{th} - 1$ control points in the j^{th} axis, and $T_{f,i} = t_i - t_{i-1}$ is the trajectory length of the i^{th} rest-to-rest primitive. The constraints (3.30b) and (3.30c) ensure dynamic feasibility as described in Section 2.3.2, while (3.30d) has been introduced to place reasonable limits on the length of each motion primitive and avoid division by zero errors. Constraints (4.17c-4.17d) provide the limits on the auxiliary variables, while (4.17e) places the initial waypoint and (4.17f) accounts for the approximation error.

Notably, for a given number of control points N, solve times for this problem formulation should theoretically be independent of the time bound T due to the introduction of motion primitive duration as an optimisation variable. This is true provided that the mission duration is sufficiently long to allow for dynamic feasibility and presents an improvement over the work in [10] where the number of optimisation variables linearly increases with the T. Instead, our formulation's number of control points N will depend geometrically on the environment and the complexity of the STL requirement, independent of T.

In the case of Boolean satisfaction, we drop the maximisation and find a solution where $\tilde{\rho}^{\varphi}(\boldsymbol{z}) \geq \epsilon$.

Chapter 4

Catmull-Rom Planning

This section extends the STL encoding developed in Chapter 3 and applies it to the Catmull-Rom spline. This addresses the inefficiencies associated with rest-to-rest primitives, particularly in terms of coming to a stop at each control point. Before doing so, we develop the prerequisite guarantees provided by the Catmull-Rom spline.

4.0.1 Feasibility Guarantees

Unlike rest-to-rest trajectories, the Catmull-Rom spline poses additional challenges in attaining feasibility guarantees. Due to its bounded nature, we focus solely on centripetal parameterisation.

Position Feasibility

By Theorem 2.4.1, the centripetal Catmull-Rom spline remains bounded through the extrusion of bounding boxes perpendicular to line segments between control points. Proposition 4.0.1 trivially follows.

Proposition 4.0.1: Position Feasibility of Catmull-Rom

For a centripetal Catmull-Rom spline, a spline segment with bounding volume $\mathcal{B}$, produced by Theorem 2.4.1, will remain in some convex region $\mathcal{A}$ if $\mathcal{B} \subseteq \mathcal{A}$.

This condition for position feasibility necessitates that for each line segment connecting control points in $\mathbb{R}^n$, we calculate $2(n-1)$ perpendicular lines and two points along each perpendicular line at a distance h from the corresponding control point. This is cumbersome

when considering these conditions as constraints in an optimisation problem. Instead, we over-approximate the bounding volume $\mathcal{B}$, avoiding the computation of tangent lines.

Consider an n-dimensional hypercube with a side length of $2h$ centred on some control point P_i. The set of vertices $\mathcal{V}_i = \{v_1, ..., v_{2^n-1}\}$ of the hypercube is given by

$$\mathcal{V}_i = \left\{ \boldsymbol{v} = (v^{(1)}, ..., v^{(n)}) \mid v^{(j)} = P_i^{(j)} \pm h \text{ for } j = 1, ..., n \right\}. \tag{4.1}$$

where $v_i \in \mathbb{R}^n$, $|\mathcal{V}_i| = \mathbb{R}^{2^n}$, and $P_i^{(j)}$ is the vertex coordinate in the j^{th} axis of control point i. Figure 4.1 shows an illustration of the two-dimensional hypercubes of a centripetal Catmull-Rom segment from P_1 to P_2 where $\mathcal{V}_1 = \{v_1, ..., v_4\}$ and $\mathcal{V}_2 = \{v_5, ..., v_8\}$.

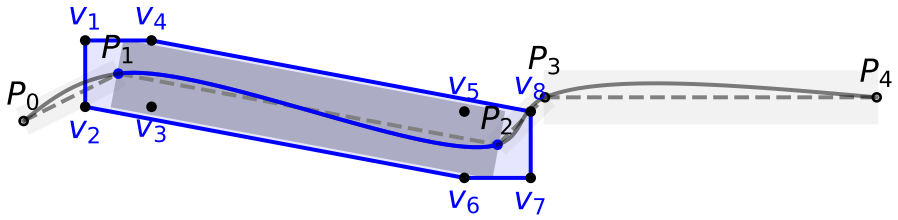


Figure 4.1: Convex hull (shaded blue) of vertex points $e_1, ..., e_8$ produced by placing hypercubes of side-length (2.47) or (2.48) at points P_1 and P_2 along a centripetal Catmull-Rom spline.

It can be shown that for the convex hull of the vertices of the set of vertices $\{\mathcal{V}_i, \mathcal{V}_{i+1}\}$, denoted $\text{conv}(\mathcal{V}_i, \mathcal{V}_{i+1})$, that $\mathcal{B} \subseteq \text{conv}(\mathcal{V}_i, \mathcal{V}_{i+1})$. Consequently, the criteria for position feasibility of a centripetal Catmull-Rom spline segment is given by Proposition 4.0.2.

Proposition 4.0.2: Over-Approximate Feasibility of Catmull-Rom

For a centripetal Catmull-Rom spline segment connecting P_i and P_{i+1}, the spline segment will remain within some convex region $\mathcal{A}$ if the convex hull $\text{conv}(\mathcal{V}_{i1}, \mathcal{V}_{i+1})$ of the hypercube vertices of (4.1) is contained in $\mathcal{A}$.

Temporal Scaling

When generating paths for quadcopters, it is crucial to adhere to maximum velocity and acceleration constraints to ensure the dynamic feasibility of the generated paths. Unfortunately, no analytical solution has been found for determining limits a priori from the control points of a Catmull-Rom spline. To address this, we employ a time-scaling method inspired by [39] and [45] whereby the time to navigate between control points is scaled by a factor γ such that $\tilde{P}(\gamma t_i) = P_i$, with a new time relation given by

$$t_{i+1} = t_i + \gamma ||P_{t_{i+1}-P_{t_i}}||^{\beta}, \ (0 \leq \beta \leq 1). \tag{4.2}$$

As γ increases, the generated path becomes more likely to be dynamically feasible as the system becomes more likely to respect the actuation limits. When generating a path $\tilde{P}(t)$ using (4.2), the derivatives are given by

$$\dot{\tilde{P}}(t) = \frac{1}{\gamma} P(\gamma t), \text{ and } \ddot{\tilde{P}}(t) = \frac{1}{\gamma^2} P(\gamma t). \tag{4.3}$$

As $\gamma \to \infty$, all derivatives tend to zero, leading to no traversal of the generated path. In [39], the authors post-analyse the generated path and evaluate the maximum observed velocity and acceleration, setting γ to

$$\gamma = \max \left(\frac{v_{max}}{\max(\dot{P}(t))}, \frac{a_{max}}{\max(\ddot{P}(t))} \right), \tag{4.4}$$

where v_{max} and a_{max} are the maximum permissible velocity and acceleration. This exact approach of post-processing the path is inappropriate in our work due to the temporal constraints imposed on each control point. We rely on a heuristic approach, setting γ before generating the path.

4.1 Problem Formulation

Consider a sequence of $N \geq 2$ positional waypoints $(\boldsymbol{x}_0, ..., \boldsymbol{x}_{N-1})$ where $\boldsymbol{x}_i \in \mathbb{R}^3$. Let $\hat{P}(t) : [0, T] \to \mathbb{R}^3$ be the centripetal Catmull-Rom spline connecting the waypoints such that each spline segment is defined by (2.41) and (4.2) with $\beta = 0.5$. Any mention of the Catmull-Rom spline hereafter will directly refer to the centripetal parameterisation.

Similar to Problems 3.1.1 and 3.1.2, we now state the problem of generating a Catmull-Rom spline for which traversal of the generated path will satisfy an STL formula φ.

Problem 4.1.1: Boolean Catmull-Rom STL Planning

Given a time bounded STL specification φ and initial state $\boldsymbol{x}_0 \in \mathbb{R}^3$, generate a centripetal Catmull-Rom trajectory $\hat{P}(t)$ such that $\hat{P}(t) \vDash \varphi$. To achieve this, we use smooth robust semantics and solve the following problem:

$$\tilde{p}^{\varphi}(\hat{P}(t)) \geq \epsilon \tag{4.5a}$$

$$\text{s.t } \hat{P}(t) = \boldsymbol{x}_0, \tag{4.5b}$$

Problem 4.1.2: Robust Catmull-Rom STL Planning

Given a bounded STL specification φ and initial state $\boldsymbol{x}_0 \in \mathbb{R}^3$, generate a centripetal Catmull-Rom trajectory $\hat{P}(t)$ that maximises satisfaction of the STL smooth robust semantics through the following problem:

$$\max_{\boldsymbol{x}_0,\ldots,\boldsymbol{x}_{N-1}} \tilde{p}^{\varphi}(\hat{P}(t)) \tag{4.6a}$$

$$\text{s.t } \hat{P}(t) = \boldsymbol{x}_0, \tag{4.6b}$$

$$\tilde{p}^{\varphi}(\hat{P}(t)) \geq \epsilon. \tag{4.6c}$$

4.2 STL Encoding

In contrast to the rest-to-rest trajectory, we no longer treat the associated time points directly as optimisation variables; instead, we incorporate them as constraints determined by (4.2). An obvious downside to this approach is the coupling between trajectory duration and trajectory length. This could be addressed by introducing a variable time-scale for each segment but would result in a loss of C^1 continuity. We instead use a single time-scale determined a prior and write the optimisation variables as $\mathbf{z} = (\boldsymbol{q}_0, \ldots \boldsymbol{q}_{N-1})$.

4.2.1 Boolean Semantics

As before, we start by developing the Boolean satisfaction for STL in terms of the control points of the Catmull-Rom spline. We use the same notation for the spatial and temporal pairs as in (3.4) and (3.5).

Eventually $\Diamond_{[a,b]}\ \psi$

Evident from (3.9) is that Boolean satisfaction is given directly by the spatial and temporal positioning of control points. As the Catmull-Rom spline interpolates directly through the control points, it can be trivially shown that Boolean satisfaction of the **eventually** operator for the rest-to-rest and Catmull-Rom splines are equivalent.

Always $\Box_{[a,b]}\ \psi$

As discussed, ensuring the position feasibility of the Catmull-Rom spline requires establishing additional criteria on the segment boundaries. Consequently, direct application of the encoding for the always operator, as employed in the rest-to-rest case, is not feasible.

As stated in Lemma 4.0.2, a spline segment is guaranteed to remain within a given convex region $\mathcal{A}$ if the convex hull formed by its corresponding hypercube's vertices is wholly contained within $\mathcal{A}$. This condition ensures that all the vertices of the hypercube lie within the boundaries of $\mathcal{A}$, thereby affirming the entirety of the hypercube and, consequently, the spline segment resides within $\mathcal{A}$. We, therefore, adapt the Boolean semantics of the rest-to-rest trajectory, giving

$$\begin{aligned}&\Big(\exists(\boldsymbol{q}_i,\boldsymbol{q}_{i+1})\in\mathcal{SP}\text{ s.t }(\mathcal{V}_i\vDash\psi)\wedge(\mathcal{V}_{i+1}\vDash\psi)\wedge(t_i\leq a)\wedge(t_{i+1}\geq a)\Big)\wedge\\&\quad\Big(\exists(\boldsymbol{q}_j,\boldsymbol{q}_{j+1})\in\mathcal{SP}\text{ s.t }(\mathcal{V}_j\vDash\psi)\wedge(\mathcal{V}_{j+1}\vDash\psi)\wedge(t_j\leq b)\wedge(t_{j+1}\geq b)\Big)\wedge\\&\quad\Big(\forall\boldsymbol{q_k}\in\{\boldsymbol{q}_l\in\mathcal{Q}\mid a\leq t_l\leq b\},\mathcal{V}_k\vDash\psi\Big)\Rightarrow\hat{P}(t)\vDash\Box_{[a,b]}\psi.\end{aligned}\tag{4.7}$$

where $\mathcal{V}_i\vDash\psi$ indicates that all the vertices $\mathcal{V}_i$ in (4.1) satisfy ψ. Similar to the rest-to-rest case, Boolean satisfaction of an always operator spanning the entire STL bound is given by

$$\forall \boldsymbol{q}_i \in \mathcal{Q}, \mathcal{V}_i \vDash \psi \Rightarrow \hat{P}(t) \vDash \Box_{[0,T]}\psi. \tag{4.8}$$

Eventually-Always $\Diamond_{[a,b]}\Box_{[0,\tau]}\psi$

It was previously shown that the encoding of the eventually-always operator can be over-approximated by the always operator with the additional criterion. Therefore, the Boolean satisfaction of the eventually-always operator for the Catmull-Rom spline is given by

$$\begin{aligned}
&\Big(\exists(\boldsymbol{q}_i, \boldsymbol{q}_{i+1}) \in \mathcal{SP} \text{ s.t } (\mathcal{V}_i \vDash \psi) \wedge (\mathcal{V}_{i+1} \vDash \psi) \wedge (t_i \leq t'_k) \wedge (t_{i+1} \geq t'_k \leq)\Big) \wedge \\
&\Big(\exists(\boldsymbol{q}_j, \boldsymbol{q}_{j+1}) \in \mathcal{SP} \text{ s.t } (\mathcal{V}_j \vDash \psi) \wedge (\mathcal{V}_{j+1} \vDash \psi) \wedge (t_j \leq t'_k + \tau) \wedge (t_{j+1} \geq t'_k + \tau)\Big) \wedge \\
&\Big(\forall \boldsymbol{q_k} \in \{\boldsymbol{q}_l \in \mathcal{Q} \mid t'_k \leq t_l \leq t'_k + \tau\}, \mathcal{V}_k \vDash \psi\Big) \wedge \\
&\Big(a \leq t'_k \leq b\Big) \Rightarrow \hat{P}(t) \vDash \Diamond_{[a,b]}\Box_{[0,\tau]}\psi.
\end{aligned} \tag{4.9}$$

Always-Eventually $\Box_{[a,b]}\Diamond_{[0,\tau]}\psi$

Lastly, we note that the Boolean satisfaction of the always-eventually operator can be established by analysis of the control points without consideration of the hypercube vertices. Therefore, we can conclude that the always-eventually Boolean encoding for the Catmull-Rom spline is equivalent to the rest-to-rest encoding developed in (3.19).

4.2.2 Robust Semantics

Through application of the same principles followed in Section 3.2.2, we now state the robust encoding of each STL temporal operator for Catmull-Rom trajectories.

Eventually $\Diamond_{[a,b]}\ \psi$

$$\tilde{\rho}^{\Diamond_{[a,b]}\psi}(\boldsymbol{z}) = \underset{\boldsymbol{q}_i \in \mathcal{Q}}{\tilde{\bigsqcup}} \left(\mu^{\psi}(\boldsymbol{q}_i) \tilde{\sqcap} \varphi^{[a,b]}(t_i)\right), \tag{4.10}$$

Always $\square_{[a,b]}\ \psi$

$$\tilde{\rho}^{\square_{[a,b]}\psi}(\boldsymbol{z}) = \varphi_1 \,\tilde{\sqcap}\, \varphi_2 \,\tilde{\sqcap} \varphi_3 \tag{4.11}$$

$$\begin{aligned}
\varphi_1 &= \tilde{\bigsqcup_{(\mathcal{V}_\rangle, \mathcal{V}_{\rangle+\infty}) \in \mathcal{SP}}} \left(\mu^\psi(\mathcal{V}_i) \,\tilde{\sqcap}\, \mu^\psi(\mathcal{V}_{i+1}) \,\tilde{\sqcap}\, \varphi^{t \leq a}(t_i) \,\tilde{\sqcap}\, \varphi^{t \geq a}(t_{i+1}) \right), \\
\varphi_2 &= \tilde{\bigsqcup_{(\mathcal{V}_|, \mathcal{V}_{|+\infty}) \in \mathcal{SP}}} \left(\mu^\psi(\mathcal{V}_j) \,\tilde{\sqcap}\, \mu^\psi(\mathcal{V}_{j+1}) \,\tilde{\sqcap}\, \varphi^{t \leq b}(t_j) \,\tilde{\sqcap}\, \varphi^{t \geq b}(t_{j+1}) \right), \\
\varphi_3 &= \tilde{\prod_{\mathcal{V}_\| \in \mathcal{Q}}} \left(\mu^\psi(\mathcal{V}_k) \,\tilde{\sqcap}\, \varphi^{[a,b]}(t_k) \right) \,\tilde{\sqcup}\, \mu^{t \leq a}(t_k) \,\tilde{\sqcup}\, \mu^{t \geq b}(t_k).
\end{aligned} \tag{4.12}$$

Here, $\mu^\psi(\mathcal{V}_{i+1})$ indicates that all vertices meet the specification such that

$$\mu^\psi(\mathcal{V}_{i+1}) = \mu^\psi(v_1) \,\tilde{\sqcap}\, ... \,\tilde{\sqcap}\, \mu^\psi(v_{2^{n-1}}). \tag{4.13}$$

Eventually-Always $\square_{[0,\tau]}\ \psi$

Similarly, we have that

$$\tilde{\rho}^{\Diamond_{[a,b]}\square_{[0,\tau]}\psi}(\boldsymbol{z}) = \phi_1 \,\tilde{\sqcap}\, \phi_2 \,\tilde{\sqcap}\, \phi_3 \tag{4.14}$$

$$\begin{aligned}
\phi_1 &= \tilde{\bigsqcup_{(\mathcal{V}_\rangle, \mathcal{V}_{\rangle+\infty}) \in \mathcal{SP}}} \left(\mu^\psi(\mathcal{V}_i) \,\tilde{\sqcap}\, \mu^\psi(\mathcal{V}_{i+1}) \,\tilde{\sqcap}\, \varphi^{t \leq t'_k}(t_i) \,\tilde{\sqcap}\, \varphi^{t'_k \leq t}(t_{i+1}) \right) \\
\phi_2 &= \tilde{\bigsqcup_{(\mathcal{V}_|, \mathcal{V}_{|+\infty}) \in \mathcal{SP}}} \left(\mu^\psi(\mathcal{V}_j) \,\tilde{\sqcap}\, \mu^\psi(\mathcal{V}_{j+1}) \,\tilde{\sqcap}\, \varphi^{t \leq t'_k + \tau}(t_j) \,\tilde{\sqcap}\, \varphi^{t'_k + \tau \leq t}(t_{j+1}) \right) \\
\phi_3 &= \tilde{\prod_{\mathcal{V}_\| \in \mathcal{Q}}} \left(\mu^\psi(\mathcal{V}_k) \,\tilde{\sqcap}\, \varphi^{[t'_k, t'_k + \tau]}(t_k) \right) \,\tilde{\sqcup}\, \mu^{t \leq t'_k}(t_k) \,\tilde{\sqcup}\, \mu^{t \geq t'_k + \tau}(t_k).
\end{aligned} \tag{4.15}$$

Always-Eventually $\square_{[a,b]}\Diamond_{[0,\tau]}\psi$

Lastly, we have the equivalent encoding

$$\begin{aligned} \tilde{\rho}^{\square_{[a,b]}\Diamond_{[0,\tau]}\psi}(\boldsymbol{z}) &= \tilde{\bigsqcup_{j=1,\dots,j=2N'+1}} \Big(\psi_1 \,\tilde{\sqcap}\, \psi_2\Big), \\ \psi_1 = \tilde{\bigsqcup_{\boldsymbol{q}_i \in \mathcal{Q}}} \mu^{\psi}(\boldsymbol{q}_i), \text{ and } \psi_2 &= \varphi^{[a+(j-1)\frac{\tau}{2}, a+j\frac{\tau}{2}]+\tau''}(t_i). \end{aligned} \tag{4.16}$$

4.2.3 Planning with Optimisation

Similarly to Section 3.2.3, we proceed to solve the planning problem using optimization techniques. Given the developed STL encoding, we solve the following optimization problem using IPOPT.

Given an STL specification φ time bounded by T and initial state $\boldsymbol{x}_0 \in \mathbb{R}^3$, maximisation of spatial robustness when following Catmull-Rom spline comprised of N control points is given by

$$\max_{\tilde{\boldsymbol{z}}} \ \tilde{\rho}^{\varphi}(\tilde{\boldsymbol{z}}) \tag{4.17a}$$

$$\text{s.t. } t_{i+1} = t_i + \gamma_i ||\boldsymbol{x}_{t_{i+1}} - \boldsymbol{x}_{t_i}||^{1.5} \tag{4.17b}$$

$$a_i \leq t_i' \leq b_i \ \forall i = 1, ..., N^{\Diamond\square}, \tag{4.17c}$$

$$0 \leq \tau_i' \leq \tau_1 \ \forall i = 1, ..., N^{\square\Diamond}, \tag{4.17d}$$

$$\boldsymbol{z}_0 = (\boldsymbol{x}_0^{(1)}, \boldsymbol{x}_0^{(2)}, \boldsymbol{x}_0^{(3)}) \tag{4.17e}$$

$$\tilde{\rho}^{\varphi}(\tilde{\boldsymbol{z}}) \geq \epsilon \tag{4.17f}$$

The constraints (4.17c-4.17d) provide the limits on the auxiliary variables, while (4.17e) places the initial waypoint and (4.17f) accounts for the approximation error. In the case of Boolean satisfaction, we again drop the maximisation in (4.17) and find a solution with the given constraints where $\tilde{\rho}^{\varphi}(\boldsymbol{z}) \geq \epsilon$. The range for the time-scale factors γ_i is heuristically chosen before solving.

Chapter 5

Experimental Results

In this chapter, we develop and present the results for a series of benchmarks to evaluate the proposed STL encoding techniques. These benchmarks include simple and complex tasks designed to challenge the encoding's robustness and flexibility. Following the benchmark results, we present numerical simulations conducted with the Crazyflie 2.1 quadcopter to demonstrate the encoding's capability to generate feasible trajectories.

5.1 Benchmark Scenarios

We consider several benchmark specifications inspired by [12] and [10] involving a quadrotor operating at a constant altitude. The state $\boldsymbol{x} = [p_x, p_y, \dot{p}_x, \dot{p}_y]^T \in X$ and input $\boldsymbol{u} = [\ddot{p}_x, \ddot{p}_y]^T \in U$ are constrained to ensure dynamic feasibility with a maximum velocity and acceleration of 0.5 m/s and 0.25 m/s^2 in each axis

$$X = \{\boldsymbol{x} \mid 0 \leq p_x \leq 3, 0 \leq p_y \leq 3\, |\dot{p}_x| < 0.5, |\dot{p}_y| < 0.5\}, \tag{5.1}$$

$$U = \{\boldsymbol{u} \mid |\ddot{p}_x| \leq 0.25, |\ddot{p}_y| \leq 0.25\}. \tag{5.2}$$

We utilise the Python CasADi interface [46] to formulate the optimisation problem using IPOPT [47]. All experiments are run on an Asus Zenbook laptop with an Intel(R) i7-10750H CPU running at 2.60GHz, and with 16 GB of Random Access Memory.

We use a value of $k = 50$ for the smoothing factor. This value is chosen to balance the approximation error (2.6, 2.7) when considering each scenario's geometric limitations against the issue of computational overflow associated with the log-sum approximation [48].

Two-Target Reach-Avoid [12]

In the first benchmark scenario shown, we impose a persistence condition requiring that the quadcopter eventually visit one of two alternate targets T_1 or T_2 for five seconds. In addition, the quadcopter must eventually reach the goal G while avoiding the obstacle denoted $\mathcal{O}$ for the mission's duration. The specification φ_1 is written

$$\varphi_1 = \Diamond_{[0,T-5]}(\Box_{[0,5]} T_1 \vee \Box_{[0,5]} T_2) \wedge \Box_{[0,T]} \neg \mathcal{O} \wedge \Diamond_{[0,T]} G. \tag{5.3}$$

Many-Target [12]

Whilst simultaneously avoiding the obstacle $\mathcal{O}$, the quadcopter must visit at least one target T_i in each group of targets T^j. There are five groups of targets, each consisting of two targets. The specification is written as

$$\varphi_2 = \bigwedge_{j=1}^{5} \left(\bigvee_{i=1}^{2} \Diamond_{[0,T]} T_i^j \right) \wedge \Box_{[0,T]}(\neg \mathcal{O}), \tag{5.4}$$

Timed Package Delivery

The quadcopter must visit two delivery regions $\mathcal{D}_1$ and $\mathcal{D}_2$ within the first 20 seconds. Following this, it must visit the base region $\mathcal{B}$ between 20 and 40 seconds. Additionally, the quadcopter must avoid the obstacle $\mathcal{O}$ for the mission's duration. The specification is written as

$$\varphi_3 = \Diamond_{[0,20]} \mathcal{D}_1 \wedge \Diamond_{[0,20]} \mathcal{D}_2 \wedge \Diamond_{[20,40]} \mathcal{B} \wedge \Box_{[0,40]}(\neg \mathcal{O}). \tag{5.5}$$

Persistent Surveillance

In this scenario, we define two regions of interest $\mathcal{R}_1$ and $\mathcal{R}_2$ where the objective of the quadcopter is to ensure repeated observation of each area every 20 seconds. The mission specification is written as

$$\varphi_4 = \Box_{[20,40]} \Diamond_{[0,20]} \mathcal{R}_1 \wedge \Box_{[20,40]} \Diamond_{[0,20]} \mathcal{R}_2. \tag{5.6}$$

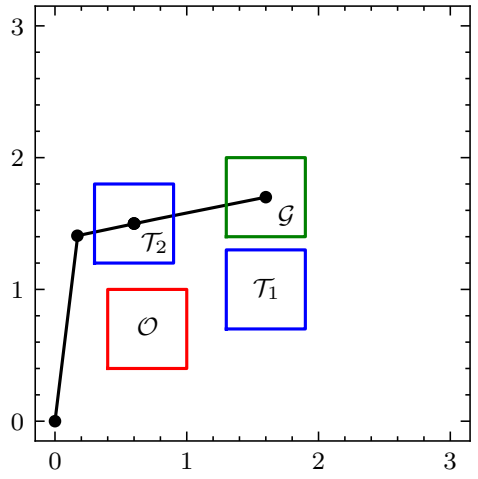
5.1.1 Rest-to-Rest Planning

In Table 5.1, we tabulate the results of each benchmark scenario using rest-to-rest splines. For each benchmark scenario we record the smooth robustness $\tilde{p}$ according to (3.20 - 3.29), robustness p according to Definition 2.1.4, and relaxed robustness p^* such that p does not consider the always-safe operator. We also record the time it takes to solve the optimisation problem and generate the rest-to-rest trajectories for both boolean and robust modes. When in robust mode, the always-safe operator is kept as a constraint in all relevant scenarios but not included in the overall maximisation of the non-linear solver. Finally, when presenting the time-to-solve, we show the average results of over five consecutive simulation runs to minimise the effects of cache warming.

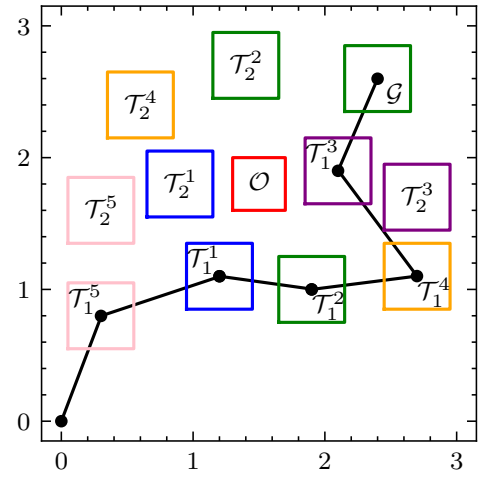
Benchmark	T	k	N	$\tilde{p}$	p	p^*	Sol^b	Sol^r
Two-Target	25	50	6	0.028	0.124	0.300	0.21	0.43
	50			0.028	0.126	0.284	0.23	0.48
Many-Target	25	50	9	-	-	-	-	-
	50			0.210	0.250	0.250	0.74	0.71
Timed Delivery	40	50	10	0.206	0.230	0.230	0.78	1.07
Surveillance	60	50	10	0.222	0.250	-	1.01	1.36

Table 5.1: Benchmark scenarios results showing mission duration T in seconds, the number of control points N, smooth robustness $\tilde{p}$ according to (3.20 - 3.29), robustness p according to Definition 2.1.4, relaxed robustness p^* such that p does not consider the always-safe operator, and the boolean Sol^b and robust Sol^r mode solve times in seconds for rest-to-rest trajectories.

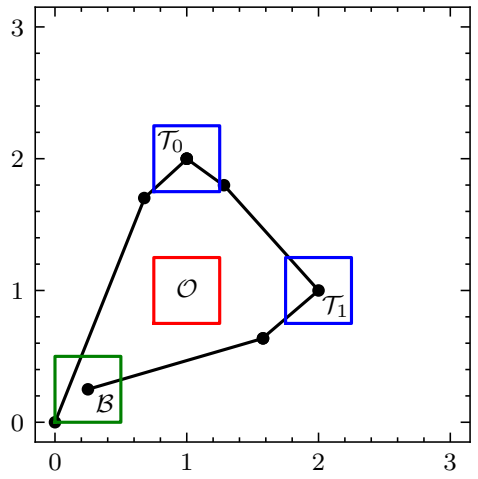
Figure 5.1 shows the traversed trajectories for each scenario when optimised for robustness. The figure indicates that spatial robustness is optimised as control points are placed at the centre of the goal areas. Notice that $p \leq 0.3$ due to the side length 0.6 m of each goal region. At the same time, it can be observed in Figure 5.2 that the generated velocity and acceleration profiles of the trajectories are considered dynamically feasible by virtue of the fact that they respect the maximum velocity and acceleration bounds.



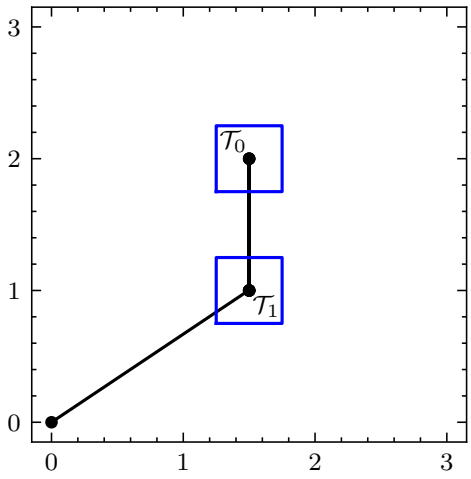
(a) Two-Target Reach-Avoid benchmark with parameters $N = 6, T = 50, k = 50$.



(b) Many Target benchmark with parameters $N = 9, T = 50, k = 50$.

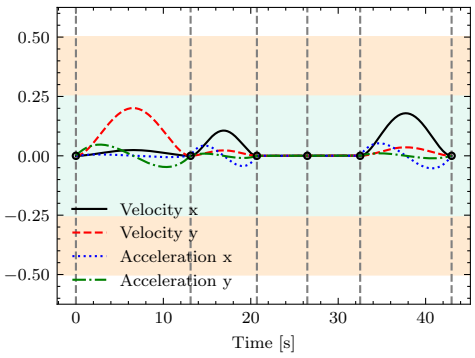


(c) Timed Package Delivery benchmark with parameters $N = 10, T = 40, k = 50$.

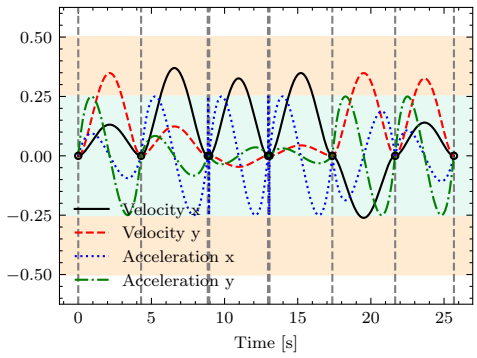


(d) Persistent Surveillance benchmark with parameters $N = 10, T = 60, k = 50$.

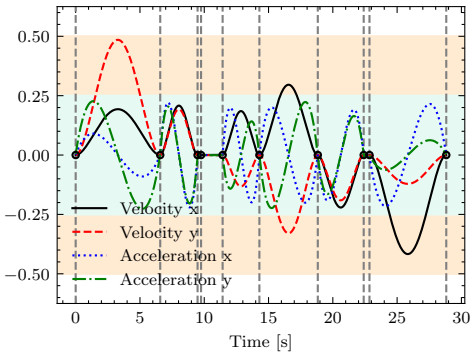
Figure 5.1: Trajectories of rest-to-rest STL benchmark scenarios including (a) Two-Target Reach-Avoid, (b) Many Target, (c) Timed Package Delivery, and (d) Persistent Surveillance.



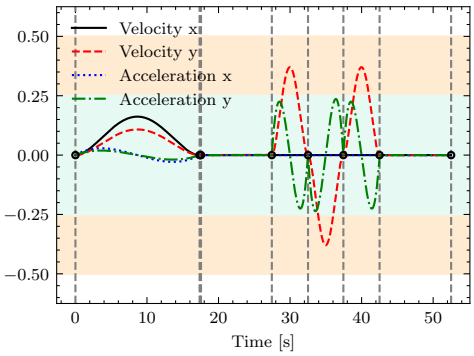
(a) Two-Target Reach-Avoid benchmark with parameters $N = 6, T = 50, k = 50$.



(b) Many Target benchmark with parameters $N = 9, T = 50, k = 50$.



(c) Timed Package Delivery benchmark with parameters $N = 10, T = 40, k = 50$.



(d) Persistent Surveillance benchmark with parameters $N = 10, T = 60, k = 50$.

Figure 5.2: Velocity and acceleration profiles of rest-to-rest STL benchmark scenarios: (a) Two-Target Reach-Avoid, (b) Many Target, (c) Timed Package Delivery, and (d) Persistent Surveillance.

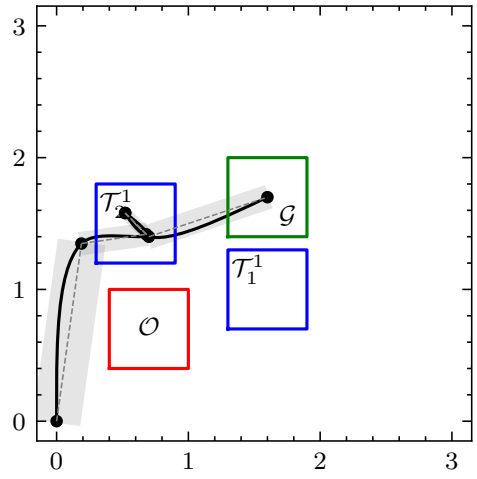
5.1.2 Catmull-Rom Planning

Similarly, Table 5.2 tabulates the results of the Catmull-Rom spline for each benchmark scenario. The always-safe is again not included in the maximisation, and solve times are again presented as an average of over five simulation runs for each benchmark scenario.

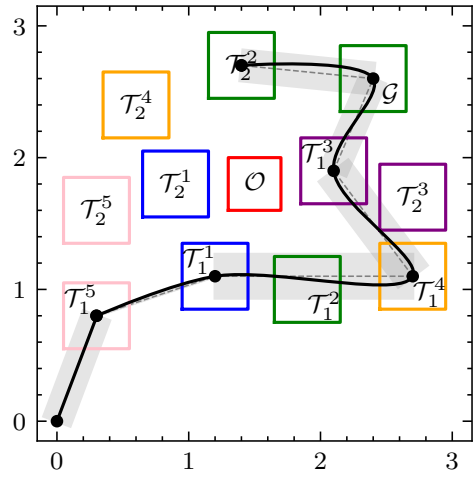
Benchmark	T	k	N	$\tilde{p}$	p	p^*	Sol^b	Sol^r
Two-Target	25	50	7	0.089	0.213	0.213	3.75	4.85
	50			0.052	0.206	0.206	4.30	2.20
Many-Target	25	50	9	0.190	0.250	0.250	2.02	1.62
	50			0.190	0.250	0.250	1.09	1.97
Timed Delivery	40	50	8	0.200	0.223	0.223	2.17	1.24
Surveillance	60	50	10	0.190	0.235	-	-	8.44

Table 5.2: Benchmark scenarios results showing mission duration T in seconds, the number of control points N, smooth robustness $\tilde{p}$ according to (3.20 - 3.29), robustness p according to Definition 2.1.4, relaxed robustness p^* such that p does not consider the always-safe operator, and the boolean Sol^b and robust Sol^r mode solve times in seconds for Catmull-Rom trajectories.

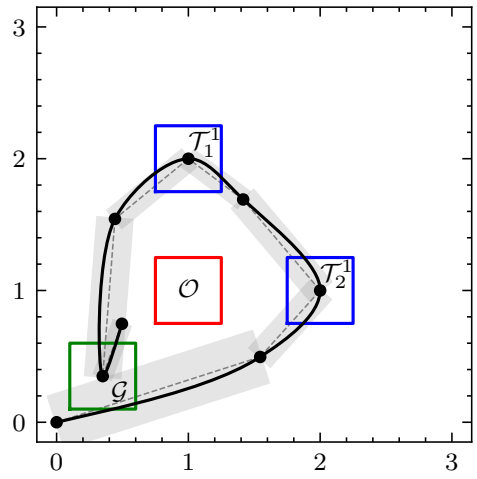
Figure 5.3 shows the traversed trajectories for each scenario when optimised for robustness, indicating that spatial robustness is optimised as control points are placed at the centre of the goal areas. Figure 5.4 shows the trajectories' generated velocity and acceleration profiles where we observe momentary violations of the maximum acceleration constraints. Considering the conservative nature of these limits, they are not expected to cause any significant issues for trajectory tracking. In practice, the temporal scaling of the Catmull-Rom splines would need to be adjusted according to each controller's tracking ability.



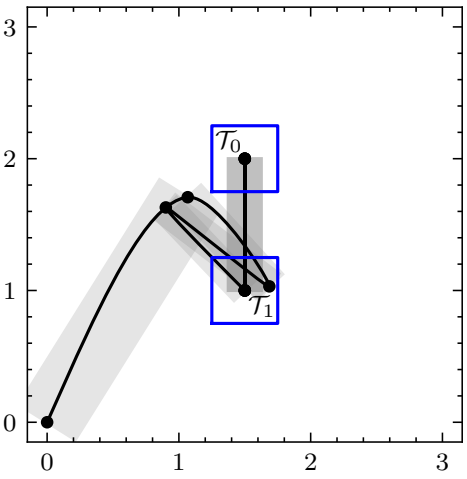
(a) Two-Target Reach-Avoid benchmark with parameters $N = 7, T = 50, k = 50$.



(b) Many Target benchmark with parameters $N = 9, T = 50, k = 50$.

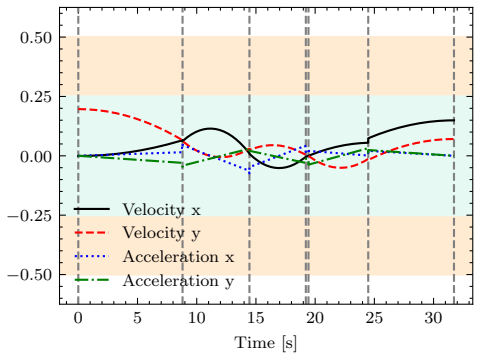


(c) Timed Package Delivery benchmark with parameters $N = 8, T = 40, k = 50$.

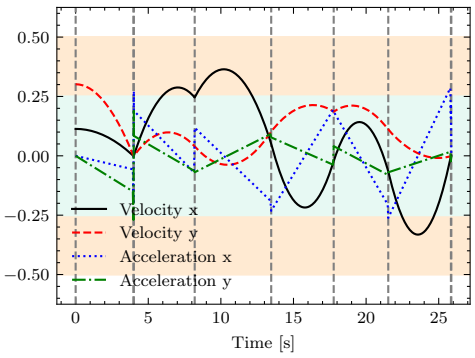


(d) Persistent Surveillance benchmark with parameters $N = 10, T = 60, k = 50$.

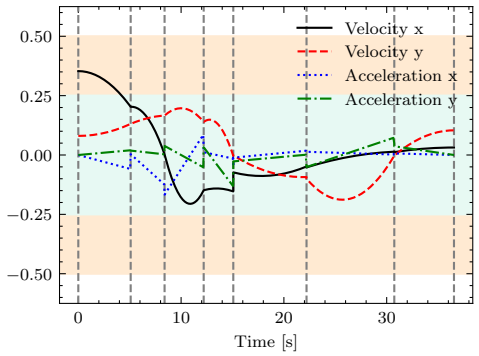
Figure 5.3: Trajectories of centripetal Catmull-Rom STL benchmark scenarios including (a) Two-Target Reach-Avoid, (b) Many Target, (c) Timed Package Delivery, and (d) Persistent Surveillance. The grey shaded areas represent the trajectory bound of each segment calculated according to (2.47).



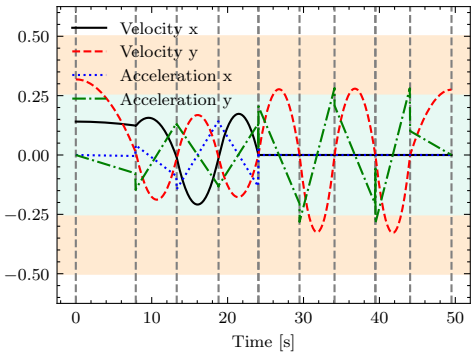
(a) Two-Target Reach-Avoid benchmark with parameters $N = 6, T = 50, k = 50$.



(b) Many Target benchmark with parameters $N = 9, T = 50, k = 50$.



(c) Timed Package Delivery benchmark with parameters $N = 8, T = 40, k = 50$.



(d) Persistent Surveillance benchmark with parameters $N = 10, T = 60, k = 50$.

Figure 5.4: Velocity and acceleration profiles of centripetal Catmull-Rom STL benchmark scenarios: (a) Two-Target Reach-Avoid, (b) Many Target, (c) Timed Package Delivery, and (d) Persistent Surveillance.

5.2 Numerical Crazyflie Simulation

We now evaluate our encoding using a numerical simulation of the Crazyflie 2.0 quadcopter. This evaluation is focused on the Catmull-Rom encoding and is motivated by three key objectives. First, we aim to demonstrate that the generated trajectory, while not explicitly dynamically feasible, is still trackable by the quadcopter. Second, we show that spatial robustness is maximized, ensuring that the STL specification is met despite imperfect tracking. Finally, we assess our method's feasibility for real-time systems with closed-loop control.

5.2.1 Experimental Setup

Figure 5.5 shows that we use a numerical simulation of the Crazyflie 2.0 quadcopter to track a generated trajectory. The trajectory is generated by the Catmull-Rom planner, which takes some STL requirement φ as input and generates a Catmull-Rom spline to be tracked by the quadcopter to satisfy the requirement. This trajectory is then used as input to an MPC controller which generates the references streamed to the onboard controllers of the Crazyflie to track the trajectory.

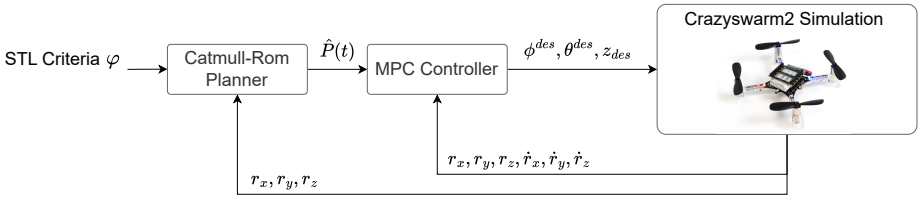


Figure 5.5: Experimental setup for the numerical Crazyfly 2.0 simulation.

In our experimentation, we utilised the Many Target benchmark scenario outlined in Section 5.1 and performed simulations for two scenarios.

- **Two-dimensional Many Target**: This scenario has the drone operating at a constant altitude of 0.5 m with centre points for all targets at the same altitude.
- **Three-dimensional Many Target**: In this scenario, targets are positioned at different heights, and the drone operates within an altitude range of 0.5 to 1.0 m. This increases the dimensionality compared to previous benchmark scenarios.

Catmull-Rom Planner

To develop the Catmull-Rom planner, we apply the same methods used in Section 5.1 for solving the benchmark cases. Specifically, we utilize the Python CasADi interface to formulate an optimization problem based from (4.17). The optimization problem is set up with objectives and constraints tailored to ensure that the generated trajectory meets the STL requirements.

The trajectory is initially generated with the starting position of the drone constrained to $(r_x, r_y, r_z) = (0, 0, 0.5)$ at $t = 0$. Replanning is performed in a shrinking horizon fashion such that an additional control point at the horizon's end is constrained to be within 0.01 units of the previous unconstrained control point. This effectively reduces the number of control points that are free for optimisation. Further, we remove STL criteria as soon as a target pair or goal point is reached. The process is continuous, with the previous solution used as a warm start for the solver, which helps reduce computational time and improve convergence efficiency.

MPC Controller

To track the reference trajectory generated by the Catmull-Rom, we develop a Quadratic Program MPC [49] controller* to calculate the reference accelerations to be streamed to the Crazyflie. Using CasADi, we develop a controller for the unit mass double integrator system according to the dynamics presented in (2.28).

The system state $\boldsymbol{x} = [p_x, p_y, p_z, \dot{p}_x, \dot{p}_y, \dot{p}_z]^T \in X$ and input signal state $\boldsymbol{u} = [\ddot{p}_x, \ddot{p}_y, \ddot{p}_z]^T \in U$ are constrained to ensure dynamic feasibility with a maximum velocity and acceleration such that

$$X = \{\boldsymbol{x} \mid 0 \leq p_x, p_y \leq 3, 0.5 \leq p_z \leq 1, |\dot{p}_x|, |\dot{p}_y|, |\dot{p}_z| < 0.5\}, \tag{5.7}$$

$$U = \{\boldsymbol{u} \mid |\ddot{p}_x| \leq 0.25, |\ddot{p}_y| \leq 0.25, |\ddot{p}_z| \leq 0.25\}. \tag{5.8}$$

The reference accelerations are converted into reference roll and pitch angles based on the equations provided in (2.29).

*https://github.com/mwlock/crazyflie_mpc

Instead of streaming the desired thrust directly, we use one of the Crazyflie's onboard controllers, which takes pitch, roll, and altitude as inputs. The controller operates at a frequency of 50 Hz, with a discretization time of 20 ms and a time horizon of 2 seconds. During experimentation, we used the altitude reference at 0.4 s from the predicted 2 second trajectory as a trivial method to account for communication delay.

Crazyswarm2 Simulation

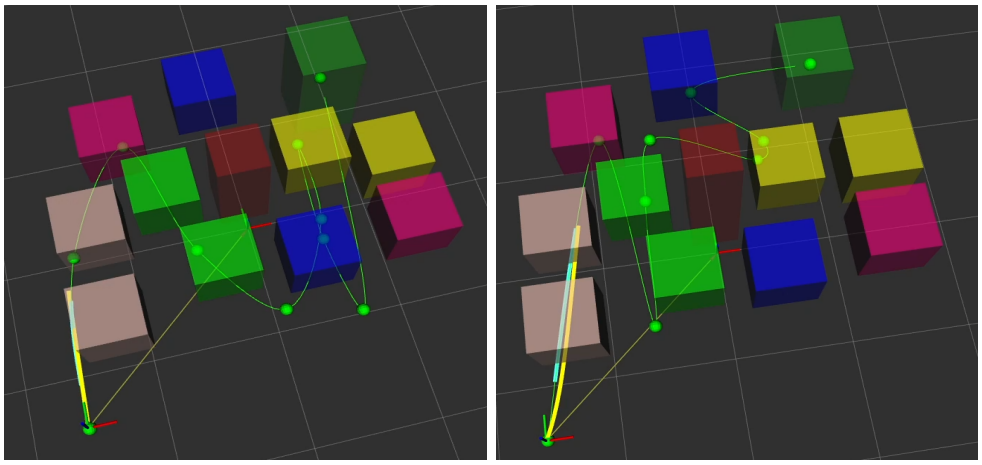
To simulate the Crazflie system dynamics, we employ the Crazyswarm2* package. This package provides an interface for commanding the Crazyflie 2.0 with the Crazyradio and a rigid body dynamics model of the Crazyflie 2.0 for numerical simulation. Interfacing with the physical and simulated drones occurs over the same ROS 2 topics, easing the transition from simulation to hardware.

5.2.2 Results

Figure 5.6 shows the RViz visualization of the initial plan determined by the Catmull-Rom planner at the start of each mission, while Figures 5.7(a) and 5.7(b) show the full path followed for the duration of each respective mission.

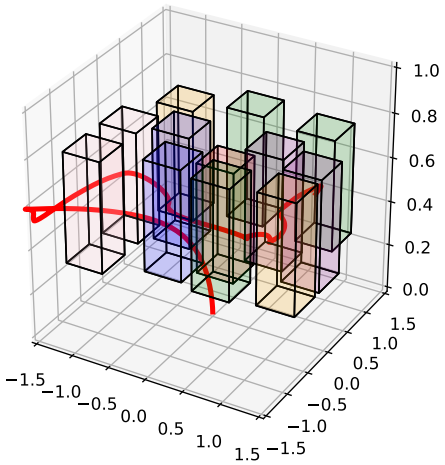
In the case of the three-dimensional mission, we only pass an updated Catmull-rom spline to the MPC controller if the solve time for the planner is below two seconds. This is done to prevent erroneous planning. It was further found that three-dimensional mission constraints had to be relaxed to produce a viable solution. More specifically, requiring that target pair five be visited was removed as a conditional constraint but was kept in the maximisation problem. As a result, we observe that the initial plan does not explicitly place a control point in either target in pair five. Nevertheless, the plan does seek to increase the chance of visiting one of the targets in the pair.

*`https://github.com/IMRCLab/crazyswarm2`

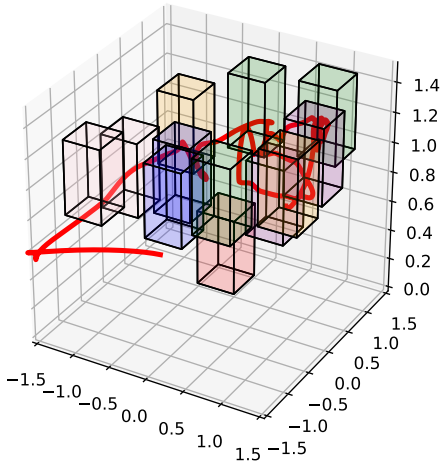


(a) Two-dimensional Many Target. (b) Three-dimensional Many Target.

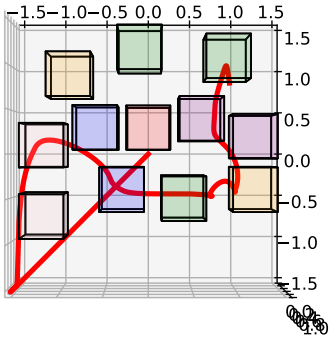
Figure 5.6: Inital plath planned at $t = 0$ for the (a) two-dimensional many target and (b) three-dimensional many target numerical Crazyflie simulations. The obstacle region is shown as the only red region. Regions of the same colour constitute a target pair. The green and yellow paths show the generated Catmull-Rom spline and MPC prediction.



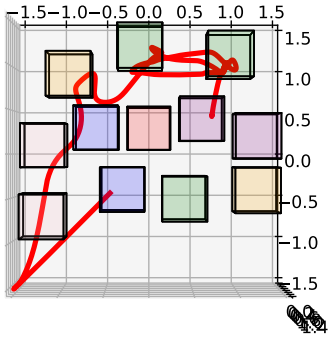
(a) Two-dimensional many target.



(b) Three-dimensional many target.

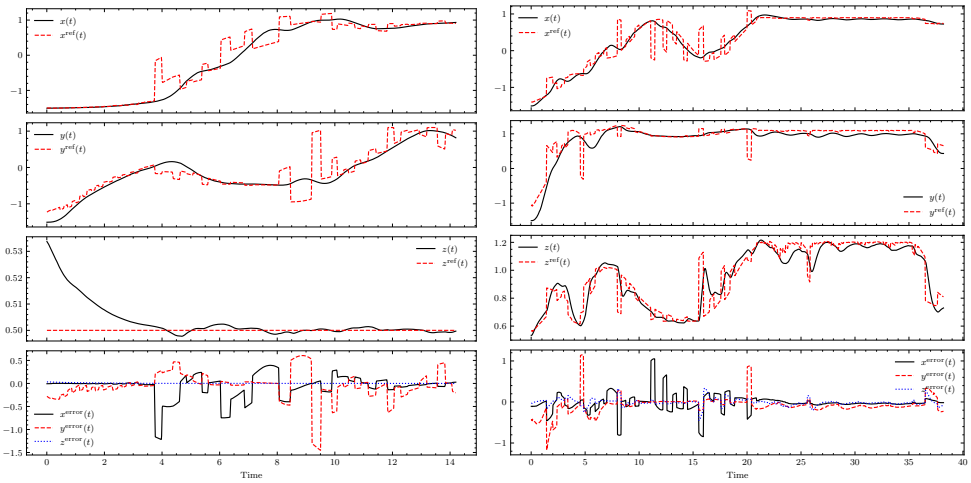


(c) Two-dimensional many target.



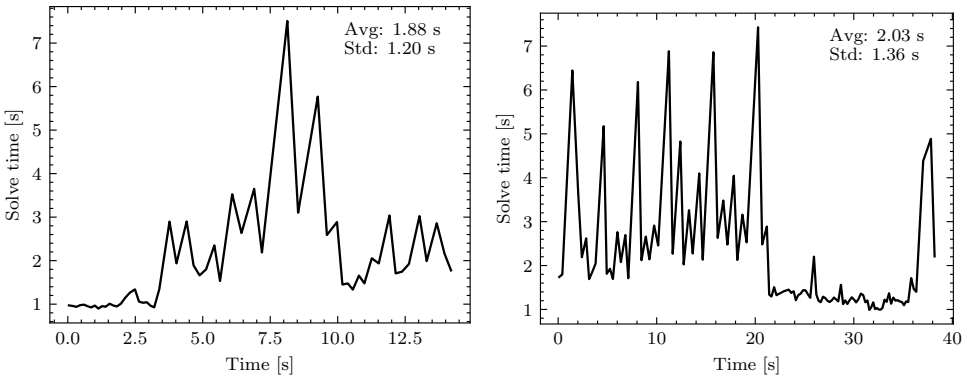
(d) Three-dimensional many target.

Figure 5.7: Path followed by the Crazyflie quadcopter for the (a) two-dimensional Many Target and (b) three-dimensional Many Targets missions. The initial movement from the centre point to the position (-1.5,-1.5, 0.5) is considered a part of the mission. Regions of the same colour constitute a target pair, while the red region represents the obstacle. The blue trace shows the path followed by the quadcopter, while (c) and (d) provide the respective top-down views.



(a) Two-dimensional Many Target. (b) Three-dimensional Many Target.

Figure 5.8: Reference trajectories for each axis continuously generated by the Catmull-Rom planner plotted against the actual trajectory followed by the quadcopter for the (a) two-dimensional and (b) three-dimensional numerical simulations.



(a) Two-dimensional Many Target. (b) Three-dimensional Many Target.

Figure 5.9: Solve times during replanning of the (a) Two-dimensional and (b) Three-dimensional Many Target numerical simulation missions.

Chapter 6

Discussion

This chapter interprets the performance and implications of the developed STL encoding for rest-to-rest and Catmull-Rom splines across the various benchmark scenarios introduced in Chapter 5. We then discuss the numerical simulation results and the encoding's ability to meet online planning requirements. Lastly, we reflect on the encoding compared to the existing spline-based STL planning.

6.1 Benchmarks

For each benchmark scenario, we discuss and compare the efficacy, efficiency, and behavioural aspects of the rest-to-rest and Catmull-Rom splines for the proposed encoding. Results for the rest-to-rest and Catmull-Rom splines are taken from Table 5.1 and Table 5.2 respectively.

6.1.1 Two-Target Reach-Avoid

Rest-to-rest

Considering first the res-to-rest trajectories, we observe a relatively low average robustness of $p = 0.124$ for the 25 and 50-second two-target reach-avoid benchmarks. Closer inspection of Figure 5.2(a) reveals that the always-safe constraint is the limiting factor wherein the second control point is relatively close to the obstacle's x-axis. Conversely, the relaxed robustness p^*, which excludes the always-safe operator in the spatial-robustness calculation, shows a robustness of $p^* = 0.30$ and $p^* = 0.284$ for the 0.124 for the 25 and 50-second missions, respectively.

This is near the maximum possible attainable robustness value for the missions and exemplifies the ability of the encoding to successfully generate a path which satisfies STL requirements.

As was initially expected, binary mode solving is more efficient than robustness maximisation, with robust mode taking almost double the time to find a solution. More interestingly, we observe a negligible difference in solve times between the 25- and 50-second mission durations. This indicates that our encoding works as intended and scales effectively with increasing mission length.

Catmull-Rom

Table 5.2 illustrates the performance of Catmull-Rom spline-based trajectories, contrasting the performance of the rest-to-rest trajectories. In comparison, the robustness measure p^* of the Catmull-Rom Two-Target Reach-Avoid benchmarks decreases slightly compared to their rest-to-rest counterparts. Looking at the path generated in Figure 5.3(a), this decrease is attributed to the two diagonally offset control points in the target area. This occurs due to the nature of the encoding, where the time between control points is positively correlated to the distance between them. Therefore, to remain in the target area for five seconds, the encoding does not place control points directly in the centre of the goal area. This is unlike the rest-to-rest trajectory, where two temporally consecutive control points can be placed at the centre point. Further, we notice a substantial difference between the smooth robustness measure $\tilde{z}$ and the robustness measure z. This is due to the encoding calculating robustness at the vertices of the convex hull of the relevant spline segment, nearer the target area borders. Regardless, the encoding still effectively produces a spatially robust trajectory.

Regarding efficiency, the Catmull-Rom splines exhibit significantly higher computational demands than the rest-to-rest solve times despite having an overall lower number of optimisation variables. This is likely due to the increased number of constraints or the geometric dependence between control points and the time to traverse, leading to an over-constraint optimisation problem and limiting the feasible region.

6.1.2 Many-Target

Rest-to-rest

The second benchmark scenario is chosen to obtain a more complex logical structure [12], requiring the agent to visit one target from each pair of targets, expressed as a disjunction in STL. The rest-to-rest trajectory requires at least one control point for each target area. This, in turn, increases the trajectory duration due to the stop-and-go nature of the motion. Consequently, no valid solution was found for the rest-to-rest Many-Target benchmark with a missing duration of 25 seconds. Increasing the time horizon to 50 seconds, however, results in a valid solution. Solve times are understandably greater than the Two-Target Reach Avoid due to the increased geometric complexity, mission complexity, and the number of optimisation variables. Additionally, we note a minor reduction in the spatial-robustness p^* that is likely due to limitation in the feasible region as described earlier.

Catmull-Rom

Unlike the rest-to-rest trajectory, the Catmull-Rom spline successfully finds a solution for the 25-second duration. This is indicative of the Catmull-Rom splines' capability to manage the geometric complexity of navigating multiple targets within shorter time constraints, as no time is wasted coming to a complete stop at each control point. The spatial-robustness p^* remains consistent across the 25 and 50-second duration missions, suggesting that the Catmull-Rom spline maintains stability in performance regardless of the mission length.

As expected, the Catmull-Rom splines show an increase in solve times over the rest-to-rest approach; for the 50-second duration, boolean mode solve times are 1.09 seconds compared to 0.74 seconds with the rest-to-rest method, and for the robust mode, they are 1.97 seconds compared to 0.71 seconds. Both increases are again attributed to the additional constraints needed to check the vertices of the over-approximated convex hull required in the always-safe constraint encoding.

Interestingly, we observe a position bound of almost zero between in what appears to be segment two in Figure 5.3(b). Upon further investigation, we find that the small position bound, determined by (2.47,) is due to control points two and three overlapping and resulting in a segment with near zero length. This creates a somewhat "redundant segment" that reduces the position bound of the following segment. A similar redundant segment is observed for the last segment of this trajectory.

6.1.3 Timed Package Delivery

In the third scenario, we assess the encoding's ability to respect timing constraints and tighter temporal constraints, requiring that goal areas be visited in a particular order.

Rest-to-rest

The results for the rest-to-rest approach in the Timed Package Delivery scenario indicate that the proposed encoding successfully adheres to the stringent timing constraints. Despite the increased temporal demands, the encoding consistently ensured that each goal area was visited in the required order within the designated time windows. Solve times for the binary mode were similar to those of the Many-Target scenario, with a slight increase observed for the robust mode while achieving a similar robustness value.

Catmull-Rom

For the Catmull-Rom approach, the results demonstrate that the encoding efficiently manages the smooth transitions between waypoints while adhering to strict timing constraints. Looking at Figures 5.2 and 5.4, we observe that the continuous motion enabled by the Catmull-Rom spline does work to reduce the overall travel time between goal areas for this benchmark. Furthermore, there is again an impact on solve times with the Catmull-Rom spline having longer solver solve times.

Notably, the robust mode's solve time is shorter than the boolean mode's. The nature of the optimisation problem in each mode might explain this counter-intuitive result. In boolean mode, the objective function is fixed, and the solver operates within a restricted and static search space. This lack of flexibility provides little direction for the IPOPT solver, which

may lead to slower convergence as it struggles to navigate efficiently. In contrast, the robust mode optimises provides the solver with more informative gradient directions from the objective function. This can reduce the complexity of the optimisation, as the solver can explore the search space more efficiently. Thus, under certain conditions and despite the additional complexity of robust computations, the robust mode's adaptive nature can enhance solution robustness and provide more efficient guidance for the solver.

6.1.4 Persistent Surveillance

Lastly, we consider the Persistent Surveillance benchmark to demonstrate the efficacy of the proposed encoding in handling the relatively complex always-eventually operator. To the author's knowledge, this is the first work to over-approximate the always-eventually operator in this manner.

Rest-to-rest

Looking at Figures 5.1 and 5.2(d), it is apparent that the agent navigates between the two target areas, repeatedly visiting each area per the STL specification. We confirm the encoding ability to approximate the nested operator by looking at the robustness p values. Due to the nature of the approximation comprising several eventually operators placed within specific time regions, we expect to see similar robustness values when compared to previous benchmarks comprising eventually operators. This holds when comparing p to the Timed Package Delivery and Many-Target Reach avoid benchmarks. Finally, we remark that for the rest-to-rest trajectory, solve-time for the benchmark scenario is not significantly higher than other benchmark scenarios despite having the strictest timing constraints.

Catmull-Rom

Compared to the rest-to-rest benchmark, the Catmull-Rom trajectory substantially increases the solve time for the Persistent Surveillance benchmark from 1.36 to 8.4 seconds. This increase is more significant than the previously observed increases and is attributed to the much more restricted solution space and the potential over-constraint nature of the optimisation problem.

6.2 Previous Works

We now review and compare our results with previous works in the field, focusing on benchmark scenarios and solve times. The comparison includes our implementations, "Rest-to-rest" and "Catmull-Rom," as well as state-of-the-art methods such as Fly-by-logic [10] and STLCCP [23].

Benchmark	T	**Solve Time (Seconds)**			
		Rest-to-rest	Catmull-Rom	Fly-by-logic[10]	STLCCP [23]
Reach Avoid	4	*	*	0.32	NA
Two-Target	25	0.43	4.85	*	*
	50	0.48	2.20	*	26.76
	25	-	1.62	*	*
Many Target	50	0.71	1.97	*	21.45

Table 6.1: Comparison of solve times across the Reach Avoid, Two-Target, and Many Target benchmark scenarios for our implementations (Rest-to-rest and Catmull-Rom) alongside Fly-by-logic [10] and STLCCP [23] implementations. The Reach Avoid scenario was only conducted in the Fly-by-logic paper. * indicates that no reasonable comparative simulation is available. T denotes mission duration.

Reach Avoid Scenario: The Reach Avoid scenario involves a straightforward task with a single goal and target. Although only presented in the Fly-by-logic paper, it is showcased here for critical comparison. The solve time reported for this scenario is 0.32 seconds for a mission duration of 4 seconds. This time is favorable for real-time applications. However, the Fly-by-logic paper does not provide results for more complex missions or longer durations. We previously speculated that the method's performance would degrade with longer missions due to the linear relationship between mission duration and the number of optimization variables, potentially limiting its applicability in more complex or prolonged scenarios.

Two-Target Scenario: In the Two-Target scenario, our implementations (Rest-to-rest and Catmull-Rom) demonstrate competitive performance. For a mission duration of 25 seconds, the Rest-to-rest method achieves a solve time of 0.43 seconds, while Catmull-Rom takes 4.85 seconds. As the mission duration increases to 50 seconds, Rest-to-rest performs faster

with a solve time of 0.48 seconds compared to 2.20 seconds for Catmull-Rom. Although Fly-by-logic results are not available for this scenario, the efficiency of our methods shows promise in handling various mission durations.

Two-Target and Many Target Scenarios: For the Two-Target scenario, our methods again perform well. The Rest-to-rest method yields a solve time of 0.43 seconds for a 25-second mission duration and 0.48 seconds for a 50-second mission duration. Catmull-Rom shows increased computational load but similar efficiency with solve times of 4.85 and 2.20 seconds for 25-second and 50-second mission durations. STLCCP, as the state-of-the-art method, reports a solve time of 26.76.45 seconds for the 50-second duration, indicating that our methods outperform STLCCP. Similarly, both of our approaches significantly outperform STLCCP, with the Catmull-Rom having a higher computational load than the rest-to-rest trajectories.

Nevertheless, it is essential to remember the foundational difference between our implementation and that of STLCCP, which is that STLCCP performs control synthesis on a dynamics model, while our methods focus on solving a geometric problem with dynamics limitations considered. With the observed results and the fact that STLCCP provides state-of-the-art performance, our work suggests that direct control synthesis (in its current state) is not ideal for long-horizon missions, reinforcing the effectiveness of our geometric-based approach.

6.3 Numerical Crazyflie Simulation

The numerical simulations of Crazyflie 2.1 provide valuable insights into the real-world applicability of the Catmull-Rom planning approach. The results confirm that the Catmull-Rom planner could generate trajectories that are robust and trackable by Crazyflie's onboard controllers despite the lack of explicit dynamic feasibility in the initial planning phase.

6.3.1 Tracking Performance and Robustness

The simulation results, particularly those illustrated in Figures 5.7(a) and 5.7(b), demonstrate that the Crazyflie was able to successfully follow the planned trajectories and achieve the mission objectives (navigating complex environments with an obstacle and multiple targets and maintaining the required spatial separation from no-fly zone) for both the two-dimensional and three-dimensional settings. Due to replanning, the quadcopter does not pass through the centre points of all goals. This happens because the constraints and objective function are updated upon reaching each objective.

The trajectory shown for the two-dimensional simulation is relatively similar, with the only meaningful difference offered over previous benchmark scenarios being replannning. However, we underscore some challenges regarding the three-dimensional scenario.

- The need to relax certain constraints in the three-dimensional scenario suggests that while the Catmull-Rom planner is highly effective, it may require adjustments to handle more complex or tighter operational constraints.
- Figures 5.7(b) and 5.7(d) show some erratic behaviour due to replanning where the initial position of the generated Catmull-Rom spline does match the current position of the quadcopter. These deviations can easily be seen in Figure 5.8, where the reference trajectory changes dramatically for short periods.
- An almost random walk-looking path is observed near the final goal pair for the three-dimensional simulation. This occurred as constant replanning with low-velocity reference trajectories prevented the quadcopter from progressing significantly towards its goal, eventually triggering the generation of a completely different path. This results in the quadcopter moving haphazardly near the end of the mission.

6.3.2 Implications for Real-Time Systems

Beyond tracking and successfully planning, the simulation highlights some challenges associated with using Catmull-Rom splines in real-time systems. As seen in Figure 5.9, the higher computational load, as seen

in the increased solve times, could limit the planner's responsiveness in fast-changing environments. With solve times occasionally exceeding seven seconds, the Catmull-Rom spline is ineffective in its current form for online planning through the proposed encoding. This is particularly relevant in the three-dimensional scenario, where longer solve times could result in missed updates or the need for more conservative adjustments.

Despite these challenges, the results suggest that with appropriate tunings—such as adjusting the splines' temporal scaling and optimising the planner's computational aspects; the Catmull-Rom approach could be a viable option for real-time trajectory planning in small UAVs like the Crazyflie for long-horizon missions with simple mission constraints.

Chapter 7

Conclusions and Future Work

In this section, we present our conclusions and consider the possible future research directions for this work.

7.1 Conclusions

This thesis introduced a novel encoding for STL-based trajectory planning utilising rest-to-rest and Catmull-Rom splines. Our goal was to address the challenges associated with real-time planning under complex spatial-temporal constraints and long-horizon missions. To do so, we proposed a smooth STL encoding for these splines, offering an alternative to existing methods that suffer from high computational demands or limited scalability. We made several key findings through benchmark testing and numerical simulations.

Novel Encoding: We showed that our proposed STL encoding adapts well to two classes of interpolating splines, offering flexibility across various applications and scenarios. The formulation illustrates its potential to be tailored to various spline configurations without compromising the integrity of the STL specifications. Furthermore, this encoding not only supports complex STL constraints, but also simplifies the robustness evaluation of a continuous trajectory by only determining overall robustness at particular trajectory points. Lastly, we also introduced and verified a new approximation for handling single-nested temporal operators.

Computational Efficiency: The spline-based approach has shown significant reductions in computational load compared to state-of-the-art works like STLCCP [23] and other methods involving MILP. For instance, in scenarios like the Two-Target Reach-Avoid and Many-Target benchmarks, our approach demonstrated quicker solve times and scaled better with increased mission duration. We note that the overall approach of trajectory planning, rather than control synthesis, is more practical for long-duration complex missions. Still, this is a trade-off as our encoding's efficiency comes with a loss of global optimality, particularly in complex scenarios with extensive constraints.

Additionally, we noted that the addition of constraints in the optimisation problem was sometimes found to enhance the solver's efficiency. By providing a more structured feasible region, we suspect the constraints guide the optimiser more effectively towards feasible solutions. However, this also implies that overly restrictive constraints could potentially limit the solution space, impacting the overall robustness of the trajectory and leading to unpredictable solve times in real-world applications.

Real-World Applicability: Numerical simulations with the Crazyflie 2.1 drone highlighted the possible application of our methods to real systems. The Catmull-Rom planner, in particular, demonstrated its capability to generate feasible, robust trajectories that the drone's onboard controllers could successfully track. Still, the simulations also revealed challenges for real-time applications due to increased computational demands, ultimately hindering replanning capabilities for dynamic obstacles or changing mission parameters.

7.2 Future Work

In advancing the field of STL-based planning, our work has opened several avenues for deeper investigation and enhancement. While the proposed encoding has demonstrated significant steps towards managing complex spatio-temporal constraints with improved computational efficiency, there are areas within our research that necessitate further exploration to capitalise on the advancements made. Some potential areas of focus are described below.

First, we suggest evaluating alternative spline forms that could better balance robustness, computational efficiency, and ease of implementation. Specifically, we are looking for splines that offer more flexibility compared to the rest-to-rest motions but also do not come with the limitations of the geometric time dependence between control points, as seen in the Catmull-Rom spline.

Secondly, we believe that benefits could be seen from developing a hierarchical planning framework that integrates our spline-based STL planning approach at a high level with a more reactive, low-level planner. This framework would allow for the high-level planner to generate feasible trajectories under global or even simplified mission constraints, while the low-level planner could dynamically adjust these trajectories in realtime to react to unforeseen obstacles or changes in the environment. In such a framework, the low-level planner might act to enforce STL robustness within a shorter planning horizon.

References

[1] B. Paden, M. Cap, S. Z. Yong, D. Yershov, and E. Frazzoli, "A Survey of Motion Planning and Control Techniques for Self-driving Urban Vehicles," Apr. 2016, arXiv:1604.07446 [cs]. [Online]. Available: http://arxiv.org/abs/1604.07446 [Page 1.]

[2] L. Quan, L. Han, B. Zhou, S. Shen, and F. Gao, "Survey of UAV motion planning," *IET Cyber-Systems and Robotics*, vol. 2, no. 1, pp. 14–21, 2020. doi: 10.1049/iet-csr.2020.0004 _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1049/iet-csr.2020.0004. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1049/iet-csr.2020.0004 [Page 1.]

[3] C. Liu, S. Lee, S. Varnhagen, and H. E. Tseng, "Path planning for autonomous vehicles using model predictive control," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. Los Angeles, CA, USA: IEEE, Jun. 2017. doi: 10.1109/IVS.2017.7995716. ISBN 978-1-5090-4804-5 pp. 174–179. [Online]. Available: http://ieeexplore.ieee.org/document/7995716/ [Page 1.]

[4] K. Bergman, *Exploiting Direct Optimal Control for Motion Planning in Unstructured Environments*, ser. Linköping Studies in Science and Technology. Dissertations. Linköping: Linköping University Electronic Press, May 2021, vol. 2133. ISBN 978-91-7929-677-3. [Online]. Available: http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-174175 [Page 2.]

[5] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, Aug. 1996. doi: 10.1109/70.508439 Conference Name: IEEE Transactions on

Robotics and Automation. [Online]. Available: https://ieeexplore.ieee.org/document/508439 [Page 2.]

[6] L. Ma, J. Xue, K. Kawabata, J. Zhu, C. Ma, and N. Zheng, "A fast RRT algorithm for motion planning of autonomous road vehicles," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, Oct. 2014. doi: 10.1109/ITSC.2014.6957824 pp. 1033–1038, iSSN: 2153-0017. [Online]. Available: https://ieeexplore.ieee.org/document/6957824 [Page 2.]

[7] O. Maler and D. Nickovic, "Monitoring Temporal Properties of Continuous Signals," in *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*, ser. Lecture Notes in Computer Science, Y. Lakhnech and S. Yovine, Eds. Berlin, Heidelberg: Springer, 2004. doi: 10.1007/978-3-540-30206-3_12. ISBN 978-3-540-30206-3 pp. 152–166. [Pages 2 and 13.]

[8] A. Donzé and O. Maler, "Robust Satisfaction of Temporal Logic over Real-Valued Signals," in *Formal Modeling and Analysis of Timed Systems*, D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, K. Chatterjee, and T. A. Henzinger, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, vol. 6246, pp. 92–106. ISBN 978-3-642-15296-2 978-3-642-15297-9 Series Title: Lecture Notes in Computer Science. [Online]. Available: http://link.springer.com/10.1007/978-3-642-15297-9_9 [Pages 2, 10, 16, and 18.]

[9] A. Linard, I. Torre, E. Bartoli, A. Sleat, I. Leite, and J. Tumova, "Real-time RRT∗ with Signal Temporal Logic Preferences." [Pages 2 and 9.]

[10] Y. V. Pant, H. Abbas, R. A. Quaye, and R. Mangharam, "Fly-by-Logic: Control of Multi-Drone Fleets with Temporal Logic Objectives," in *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*, Apr. 2018. doi: 10.1109/ICCPS.2018.00026 pp. 186–197. [Pages xi, 2, 4, 7, 10, 31, 33, 41, 51, and 70.]

[11] V. Raman, A. Donzé, M. Maasoumy, R. M. Murray, A. Sangiovanni-Vincentelli, and S. A. Seshia, "Model Predictive Control for Signal Temporal Logic Specification," Mar. 2017, arXiv:1703.09563 [cs]. [Online]. Available: http://arxiv.org/abs/1703.09563 [Pages 2 and 6.]

[12] V. Kurtz and H. Lin, "Mixed-Integer Programming for Signal Temporal Logic with Fewer Binary Variables," May 2022, arXiv:2204.06367 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2204.06367 [Pages 2, 6, 38, 51, 52, and 67.]

[13] Q. H. Ho, R. B. Ilyes, Z. N. Sunberg, and M. Lahijanian, "Automaton-Guided Control Synthesis for Signal Temporal Logic Specifications," Oct. 2022, arXiv:2207.03662 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2207.03662 [Page 5.]

[14] L. Lindemann and D. V. Dimarogonas, "Control Barrier Functions for Signal Temporal Logic Tasks," *IEEE Control Systems Letters*, vol. 3, no. 1, pp. 96–101, Jan. 2019. doi: 10.1109/LCSYS.2018.2853182 Conference Name: IEEE Control Systems Letters. [Page 5.]

[15] S. Karaman, R. G. Sanfelice, and E. Frazzoli, "Optimal control of Mixed Logical Dynamical systems with Linear Temporal Logic specifications," in *2008 47th IEEE Conference on Decision and Control*. Cancun, Mexico: IEEE, 2008. doi: 10.1109/CDC.2008.4739370. ISBN 978-1-4244-3123-6 pp. 2117–2122. [Online]. Available: https://ieeexplore.ieee.org/document/4739370 [Page 5.]

[16] E. M. Wolff, U. Topcu, and R. M. Murray, "Optimization-based Control of Nonlinear Systems with Linear Temporal Logic Specifications." [Page 5.]

[17] V. Raman, M. Maasoumy, and A. Donzé, "Model predictive control from signal temporal logic specifications: a case study," in *Proceedings of the 4th ACM SIGBED International Workshop on Design, Modeling, and Evaluation of Cyber-Physical Systems*. Berlin Germany: ACM, Apr. 2014. doi: 10.1145/2593458.2593472. ISBN 978-1-4503-2871-5 pp. 52–55. [Online]. Available: https://dl.acm.org/doi/10.1145/2593458.2593472 [Pages 6 and 14.]

[18] S. Saha and A. A. Julius, "An MILP approach for real-time optimal controller synthesis with Metric Temporal Logic specifications," in *2016 American Control Conference (ACC)*, Jul. 2016. doi: 10.1109/ACC.2016.7525063 pp. 1105–1110, iSSN: 2378-5861. [Page 6.]

[19] Y. V. Pant, H. Abbas, and R. Mangharam, "Smooth operator: Control using the smooth robustness of temporal logic," in *2017 IEEE Conference on Control Technology and Applications (CCTA)*. Mauna Lani Resort, HI, USA: IEEE, Aug. 2017. doi: 10.1109/CCTA.2017.8062628. ISBN 978-1-5090-2182-6 pp. 1235–1240. [Online]. Available: http://ieeexplore.ieee.org/document/8062628/ [Pages 7, 10, 17, and 33.]

[20] Y. Mao, B. Acikmese, P.-L. Garoche, and A. Chapoutot, "Successive Convexification for Optimal Control with Signal Temporal Logic Specifications," in *25th ACM International Conference on Hybrid Systems: Computation and Control*. Milan Italy: ACM, May 2022. doi: 10.1145/3501710.3519518. ISBN 978-1-4503-9196-2 pp. 1–7. [Online]. Available: https://dl.acm.org/doi/10.1145/3501710.3519518 [Pages 7, 14, 15, and 16.]

[21] Y. Mao, M. Szmuk, X. Xu, and B. Acikmese, "Successive Convexification: A Superlinearly Convergent Algorithm for Non-convex Optimal Control Problems," Feb. 2019, arXiv:1804.06539 [math]. [Online]. Available: http://arxiv.org/abs/1804.06539 [Page 7.]

[22] I. Haghighi, N. Mehdipour, E. Bartocci, and C. Belta, "Control from Signal Temporal Logic Specifications with Smooth Cumulative Quantitative Semantics," Apr. 2019, arXiv:1904.11611 [cs]. [Online]. Available: http://arxiv.org/abs/1904.11611 [Page 7.]

[23] Y. Takayama, K. Hashimoto, and T. Ohtsuka, "STLCCP: An Efficient Convex Optimization-based Framework for Signal Temporal Logic Specifications," May 2023, arXiv:2305.09441 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2305.09441 [Pages xi, 8, 70, and 76.]

[24] J. Karlsson, F. S. Barbosa, and J. Tumova, "Sampling-based Motion Planning with Temporal Logic Missions and Spatial

Preferences," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 15 537–15 543, 2020. doi: 10.1016/j.ifacol.2020.12.2397. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2405896320330792 [Page 9.]

[25] F. S. Barbosa, D. Duberg, P. Jensfelt, and J. Tumova, "Guiding Autonomous Exploration With Signal Temporal Logic," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3332–3339, Oct. 2019. doi: 10.1109/LRA.2019.2926669. [Online]. Available: https://ieeexplore.ieee.org/document/8754788/ [Page 9.]

[26] C.-I. Vasile, V. Raman, and S. Karaman, "Sampling-based synthesis of maximally-satisfying controllers for temporal logic specifications," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Vancouver, BC: IEEE, Sep. 2017. doi: 10.1109/IROS.2017.8206235. ISBN 978-1-5386-2682-5 pp. 3840–3847. [Online]. Available: http://ieeexplore.ieee.org/document/8206235/ [Page 9.]

[27] Y. V. Pant, H. Yin, M. Arcak, and S. A. Seshia, "Co-design of Control and Planning for Multi-rotor UAVs with Signal Temporal Logic Specifications," in *2021 American Control Conference (ACC)*, May 2021. doi: 10.23919/ACC50511.2021.9483206 pp. 4209–4216, iSSN: 2378-5861. [Page 10.]

[28] A. Rodionova, L. Lindemann, M. Morari, and G. J. Pappas, "Time-Robust Control for STL Specifications," in *2021 60th IEEE Conference on Decision and Control (CDC)*, Dec. 2021. doi: 10.1109/CDC45484.2021.9683477 pp. 572–579, iSSN: 2576-2370. [Page 10.]

[29] A. Rodionova, L. Lindemann, M. Morari, and G. Pappas, "Temporal Robustness of Temporal Logic Specifications: Analysis and Control Design," *ACM Transactions on Embedded Computing Systems*, vol. 22, no. 1, pp. 1–44, Jan. 2023. doi: 10.1145/3550072. [Online]. Available: https://dl.acm.org/doi/10.1145/3550072 [Pages 10, 11, 18, 19, and 20.]

[30] A. Rodionova, L. Lindemann, M. Morari, and G. J. Pappas, "Combined Left and Right Temporal Robustness for Control Under

STL Specifications," *IEEE Control Systems Letters*, vol. 7, pp. 619–624, 2023. doi: 10.1109/LCSYS.2022.3209928 Conference Name: IEEE Control Systems Letters. [Pages 11 and 21.]

[31] X. Yu, X. Yin, and L. Lindemann, "Efficient STL Control Synthesis under Asynchronous Temporal Robustness Constraints," Jul. 2023, arXiv:2307.12855 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2307.12855 [Page 11.]

[32] J. Verhagen, L. Lindemann, and J. Tumova, "Temporally Robust Multi-Agent STL Motion Planning in Continuous Time," Oct. 2023, arXiv:2310.10585. [Online]. Available: http://arxiv.org/abs/2310.10585 [Page 11.]

[33] Z. Lin and J. S. Baras, "Optimization-based Motion Planning and Runtime Monitoring for Robotic Agent with Space and Time Tolerances," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 1874–1879, 2020. doi: 10.1016/j.ifacol.2020.12.2606. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2405896320333590 [Page 11.]

[34] B. Xu and K. Sreenath, "Safe Teleoperation of Dynamic UAVs Through Control Barrier Functions," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018. doi: 10.1109/ICRA.2018.8463194 pp. 7848–7855, iSSN: 2577-087X. [Page 21.]

[35] D. W. Mellinger, "Trajectory Generation and Control for Quadrotors." [Page 22.]

[36] M. W. Mueller, M. Hehn, and R. D'Andrea, "A Computationally Efficient Motion Primitive for Quadrocopter Trajectory Generation," *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1294–1310, Dec. 2015. doi: 10.1109/TRO.2015.2479878 Conference Name: IEEE Transactions on Robotics. [Pages 23 and 25.]

[37] E. Catmull and R. Rom, "A CLASS OF LOCAL INTERPOLATING SPLINES," in *Computer Aided Geometric Design*, R. E. Barnhill and R. F. Riesenfeld, Eds. Academic Press, Jan. 1974, pp. 317–326. ISBN 978-0-12-079050-0. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780120790500500205 [Pages 26 and 27.]

[38] C. Yuksel, S. Schaefer, and J. Keyser, "Parameterization and applications of Catmull–Rom curves," *Computer-Aided Design*, vol. 43, no. 7, pp. 747–755, Jul. 2011. doi: 10.1016/j.cad.2010.08.008. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0010448510001533 [Pages 26 and 29.]

[39] J. Kim, M. Jin, S. Park, S. Y. Chung, and M. Hwang, "Task Space Trajectory Planning for Robot Manipulators to Follow 3-D Curved Contours," *Electronics*, vol. 9, p. 1424, Sep. 2020. doi: 10.3390/electronics9091424 [Pages 26 and 45.]

[40] "A TD-RRT* Based Real-Time Path Planning of a Nonholonomic Mobile Robot and Path Smoothening Technique Using Catmull-Rom Interpolation," 2022. [Page 26.]

[41] J.-w. Choi, R. Curry, and G. Elkaim, "Path Planning Based on Bézier Curve for Autonomous Ground Vehicles," in *Advances in Electrical and Electronics Engineering - IAENG Special Edition of the World Congress on Engineering and Computer Science 2008*, Oct. 2008. doi: 10.1109/WCECS.2008.27 pp. 158–166. [Online]. Available: https://ieeexplore.ieee.org/document/5233184 [Page 26.]

[42] S. Deolasee, Q. Lin, J. Li, and J. M. Dolan, "Spatio-temporal Motion Planning for Autonomous Vehicles with Trapezoidal Prism Corridors and B\'{e}zier Curves," Sep. 2022, arXiv:2209.15150 [cs]. [Online]. Available: http://arxiv.org/abs/2209.15150 [Page 26.]

[43] C. Yuksel, S. Schaefer, and J. Keyser, "On the parameterization of Catmull-Rom curves," in *2009 SIAM/ACM Joint Conference on Geometric and Physical Modeling*. San Francisco California: ACM, Oct. 2009. doi: 10.1145/1629255.1629262. ISBN 978-1-60558-711-0 pp. 47–53. [Online]. Available: https://dl.acm.org/doi/10.1145/1629255.1629262 [Pages ix, 28, and 30.]

[44] P. J. Barry and R. N. Goldman, "A recursive evaluation algorithm for a class of Catmull-Rom splines," *SIGGRAPH Comput. Graph.*, vol. 22, no. 4, pp. 199–204, Jun. 1988. doi: 10.1145/378456.378511. [Online]. Available: https://dl.acm.org/doi/10.1145/378456.378511 [Page 28.]

[45] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE International*

Conference on Robotics and Automation, May 2011. doi: 10.1109/ICRA.2011.5980409 pp. 2520–2525, iSSN: 1050-4729. [Page 45.]

[46] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, Mar. 2019. doi: 10.1007/s12532-018-0139-4. [Online]. Available: https://doi.org/10.1007/s12532-018-0139-4 [Page 51.]

[47] A. S. Nemirovski and M. J. Todd, "Interior-point methods for optimization," *Acta Numerica*, vol. 17, pp. 191–234, May 2008. doi: 10.1017/S0962492906370018 Publisher: Cambridge University Press. [Online]. Available: https://www.cambridge.org/core/journals/acta-numerica/article/interiorpoint-methods-for-optimization/F6097FE6068CB228A724F28C3E3814A1 [Page 51.]

[48] P. Blanchard, D. J. Higham, and N. J. Higham, "Accurate Computation of the Log-Sum-Exp and Softmax Functions," Sep. 2019, arXiv:1909.03469 [cs, math]. [Online]. Available: http://arxiv.org/abs/1909.03469 [Page 52.]

[49] E. F. Camacho and C. Bordons, *Model Predictive control*, ser. Advanced Textbooks in Control and Signal Processing, M. J. Grimble and M. A. Johnson, Eds. London: Springer, 2007. ISBN 978-1-85233-694-3 978-0-85729-398-5. [Online]. Available: http://link.springer.com/10.1007/978-0-85729-398-5 [Page 60.]

TRITA-EECS-EX 2023:0000
Stockholm, Sweden 2025

www.kth.se

€€€€ For DIVA €€€€

{
"Author1": { "Last name": "Lock",
"First name": "Matthew W.",
"Local User Id": "u1wr4fjl",
"E-mail": "mwlock@kth.se",
"organisation": {"L1": "School of Electrical Engineering and Computer Science",
}
},
"Cycle": "2",
"Course code": "EA248X",
"Credits": "30.0",
"Degree1": {"Educational program": "Master's Programme, Embedded Systems, 120 credits"
,"programcode": "TEBSM"
,"subjectArea": "Electrical Engineering"
},
"Title": {
"Main title": "Robust STL Planning Through Interpolating Splines",
"Language": "eng" },
"Alternative title": {
"Main title": "Robust STL-Planering Genom Interpolerande Splines",
"Language": "swe"
},
"Supervisor1": { "Last name": "Vahs",
"First name": "Matti ",
"Local User Id": "u17v01ps",
"E-mail": "vahs@kth.se",
"organisation": {"L1": "School of Electrical Engineering and Computer Science",
"L2": "Division of Robotics, Perception and Learning" }
},
"Supervisor2": { "Last name": "Tumova",
"First name": "Jana ",
"Local User Id": "u19qofjk",
"E-mail": "tumova@kth.se",
"organisation": {"L1": "School of Electrical Engineering and Computer Science",
"L2": "Division of Robotics, Perception and Learning" }
},
"Examiner1": { "Last name": "Dimarogonas",
"First name": "Dimos V.",
"Local User Id": "u1zrhfkn",
"E-mail": "dimos@kth.se",
"organisation": {"L1": "School of Electrical Engineering and Computer Science",
"L2": "Division of Decision and Control Systems" }
},
"National Subject Categories": "10201, 10206",
"Other information": {"Year": "2025", "Number of pages": "xiii,86"},
"Copyrightleft": "copyright",
"Series": { "Title of series": "TRITA-EECS-EX" , "No. in series": "2023:0000" },
"Opponents": { "Name": "A. B. Normal & A. X. E. Normalè"},
"Presentation": { "Date": "2022-03-15 13:00"
,"Language":"eng"
,"Room": "via Zoom https://kth-se.zoom.us/j/ddddddddddd"
,"Address": "Isafjordsgatan 22 (Kistagången 16)"
,"City": "Stockholm" },
"Number of lang instances": "2",
"Abstract[eng]": €€€€

In autonomous systems, robust spatio-temporal planning is essential for navigating complex environments, where precise spatial trajectories and temporal constraints coordination is critical. Signal Temporal Logic (STL) provides a powerful formalism for defining these constraints, with quantitative semantics offering robustness measures vital for safety-critical and disturbance-resilient applications. However, STL-based planning is computationally intensive, posing challenges for real-time applications, particularly in long-horizon missions. \\

\noindent
This thesis works towards addressing this limitation by developing a smooth, spline-based STL encoding for an STL fragment that excludes the until operator and supports single-level recursion, including the eventually-always and always-eventually operators, enabling more efficient trajectory planning. The encoding, designed for a Crazyflie 2.1 quadrotor, leverages rest-to-rest and Catmull-Rom splines, decoupling optimization variable count from STL horizon length to enhance computational efficiency in extended missions. Evaluated across benchmarks, the encoding improved performance over previous STL planning methods, with solve times scaling more efficiently with horizon length. For further validation, we applied the encoding in a numerical simulation of a Crazyflie 2.1 performing a complex reach-avoid task, achieving robust trajectory tracking with improved solve times, though still limited for stringent real-time applications.

€€€€,

"Keywords[eng]": €€€€
Signal Temporal Logic (STL), Spatio-temporal planning, Spline-based planning, Crazyflie 2.1 €€€€,
"Abstract[swe]": €€€€

% \generalExpl{Enter your Swedish abstract or summary here!}
% \sweExpl{Alla avhandlingar vid KTH \textbf{måste ha} ett abstrakt på både \textit{engelska} och \textit{svenska}.
% Om du skriver din avhandling på svenska ska detta göras först (och placera det som det första abstraktet) - och du bör revidera det vid behov.}

% \engExpl{If you are writing your thesis in English, you can leave this until the draft version that goes to your opponent for the written opposition. In this way, you can provide the English and Swedish abstract/summary information that can be used in the announcement for your oral presentation.\\If you are writing your thesis in English, then this section can be a summary targeted at a more general reader. However, if you are writing your thesis in Swedish, then the reverse is true - your abstract should be for your target audience, while an English summary can be written targeted at a more general audience.\\This means that the English abstract and Swedish sammnfattning % or Swedish abstract and English summary need not be literal translations of each other.}

% \warningExpl{Do not use the \textbackslash glspl\{\} command in an abstract that is not in English, as my programs do not know how to generate plurals in other languages. Instead, you will need to spell these terms out or give the proper plural form. In fact, it is a good idea not to use the glossary commands at all in an abstract/summary in a language other than the language used in the \texttt{acronyms.tex file} - since the glossary package does \textbf{not} support use of more than one language.}

% \engExpl{The abstract in the language used for the thesis should be the first abstract, while the Summary/Sammanfattning in the other language can follow}

I autonoma system är robust spatiotemporal planering, det vill säga planering i tid och rum, avgörande för att navigera i komplexa miljöer, där noggrann koordinering av banor och tidsbegränsningar är kritisk. Signal Temporal Logic (STL) är en kraftfull formalism för att definiera dessa begränsningar, med kvantitativ semantik som erbjuder robusthetsmått som är viktiga för säkerhetskritiska och störningståliga applikationer. STL-baserad planering är dock beräkningsintensiv, vilket innebär utmaningar för realtidsapplikationer, särskilt i uppdrag med lång tidshorisont.\\

\noindent
Denna avhandling syftar till att ta itu med denna begränsning genom att utveckla en jämn, splinebaserad STL-kodning för ett STL-fragment som utesluter "until"-operatorn och stöder rekursion på en nivå, inklusive operatörerna "eventually-always" och "always-eventually", vilket möjliggör effektivare banplanering. Kodningen, som är utformad för en Crazyflie 2.1-drönare med fyra rotorer, utnyttjar "rest-to-rest"- och "Catmull-Rom"-splines, som frikopplar antalet optimeringsvariabler från STL-horisontlängden för att förbättra beräkningseffektiviteten vid längre uppdrag. Utvärderat över diverse riktmärken förbättrade kodningen prestandan jämfört med tidigare STL-planeringsmetoder, med lösningstider som skalar mer effektivt med horisontlängden. För ytterligare validering tillämpade vi kodningen i en numerisk simulering av en Crazyflie 2.1 som utförde en komplex "reach-avoid"-uppgift och uppnådde robust banföljning med förbättrade lösningstider, även om den fortfarande är begränsad för mer krävande realtidsapplikationer.

€€€€,
"Keywords[swe]": €€€€
Signal Temporal Logic (STL), Spatio-temporal planering, Spline-baserad planering, Crazyflie 2.1 €€€€,
}

acronyms.tex

```
%%% Local Variables:
%%% mode: latex
%%% TeX-master: t
%%% End:
% The following command is used with glossaries-extra
\setabbreviationstyle[acronym]{long-short}
% The form of the entries in this file is \newacronym{label}{acronym}{phrase}
%                                    or \newacronym[options]{label}{acronym}{phrase}
% see "User Manual for glossaries.sty" for the  details about the options, one example is shown below
% note the specification of the long form plural in the line below
\newacronym[longplural={Debugging Information Entities}]{DIE}{DIE}{Debugging Information Entity}
%
% The following example also uses options
\newacronym[shortplural={OSes}, firstplural={operating systems (OSes)}]{OS}{OS}{operating system}

% note the use of a non-breaking dash in long text for the following acronym
\newacronym{IQL}{IQL}{Independent -QLearning}

\newacronym{KTH}{KTH}{KTH Royal Institute of Technology}

\newacronym{LAN}{LAN}{Local Area Network}
\newacronym{VM}{VM}{virtual machine}
% note the use of a non-breaking dash in the following acronym
\newacronym{WiFi}{-WiFi}{Wireless Fidelity}

\newacronym{WLAN}{WLAN}{Wireless Local Area Network}
\newacronym{UN}{UN}{United Nations}
\newacronym{SDG}{SDG}{Sustainable Development Goal}

% ======================================================================
% My acronyms
% ======================================================================

\newacronym{ROS}{ROS}{Robot Operating System}
\newacronym{ITRL}{ITRL}{Integrated Transport Research Lab}

% MPC Thesis Acronyms
\newacronym{MPC}{MPC}{Model Predictive Control}
\newacronym{RHC}{RHC}{Receding Horizon Controller}
% \newacronym{UAV}{UAV}{unmanned aerial vehicle}
\newacronym[plural=UAVs]{UAV}{UAV}{unmanned aerial vehicle}


% Solvers
\newacronym{IPOPT}{IPOPT}{Interior Point Optimizer}

% Reachability
\newacronym{HJ}{HJ}{Hamilton-Jacobian}

% Systems
\newacronym{MLD}{MLD}{Mixed Logic Dynamical}

% Temporal Logics
\newacronym{TL}{TL}{Temporal Logic}
\newacronym{STL}{STL}{Signal Temporal Logic}
\newacronym{scLTL}{scLTL}{syntactically co-safe Linear Temporal Logic}
\newacronym{LTL}{LTL}{Linear Temporal Logic}
\newacronym{MTL}{MTL}{Metric Temporal Logic}

\newacronym{PNF}{PNF}{Positive Normal Form}


% Programming
\newacronym{NLP}{NLP}{Non-Linear Program}
\newacronym{MIP}{MIP}{Mixed Integer Programming}
\newacronym{MILP}{MILP}{Mixed Integer Linear Programming}
\newacronym{MIQP}{MIQP}{Mixed Integer Quadratic Programming}
\newacronym{MICP}{MICP}{Mixed Integer Convex Programming}
\newacronym{SQP}{SQP}{Sequential Quadratic Programming}
\newacronym{SCP}{SCP}{Sequential Convex Programming}
\newacronym{CCP}{CCP}{convex-concave procedure}
\newacronym{DC}{DC}{difference of convex}

\newacronym{BMC}{BMC}{Bounded Model Checking}

% PLanning
```

```
\newacronym{PRM}{PRM}{Probabilistic Roadmaps}
\newacronym{RRT}{RRT}{Rapidly-exploring Random Trees}
\newacronym{RT-RRT*}{RT-RRT*}{Real-time RRT*}
```