TÉCNICO LISBOA

Passively Safe Trajectory Generation using Model Predictive Control

João Guilherme Mendes da Costa Felizardo

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Advisor(s)/Supervisor(s): Prof. Daniel de Matos Silvestre
Prof. Rita Maria Mendes de Almeida Correia da Cunha

Examination Committee

Chairperson: Prof. João Luís Da Costa Campos Gonçalves Sobrinho
Advisor: Prof. Daniel de Matos Silvestre
Members of the Committee: Prof. Rodrigo Martins de Matos Ventura

November 2024

Declaration

I declare that this document is an original work of my own authorship and that it fulfils all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Acknowledgments

This thesis marks the end of my journey at IST. I will forever remember with joy these last five years, and I want to thank all the people who left their mark on this wonderful experience.

Firstly, I would like to express my gratitude to my supervisor, Professor Daniel Silvestre, and to Dr. Pedro Lourenço. I am deeply thankful for the time and effort you dedicated to guiding me throughout this work.

To my family, thank you for your constant care, encouragement and belief in me. I am forever grateful for the love and support I felt throughout my life. Each one of you has helped shape the person I am today, and I could not have done this without you.

To my childhood friends, the good times we shared together are invaluable. I am grateful to each of you for being there through both the ups and downs, and I will always appreciate your friendship.

Finally, to my IST friends, I want to thank you for your companionship and support. Sharing this experience with you all has made it infinitely more meaningful. You have made this journey truly memorable.

This work was partially supported by the Portuguese Fundação para a Ciência e a Tecnologia (FCT) through project FirePuma (https://doi.org/10.54499/PCIF/MPG/0156/2019), through LARSyS FCT funding (DOI: 10.54499/LA/P/0083/2020, 10.54499/UIDP/50009/2020, and 10.54499/UIDB/50009/2020) and through COPELABS, University Lusófona project 10.54499/UIDB/ 04111/2020.

Abstract

The rapid advancement of autonomous systems in areas such as aerospace, automotive, and robotics has intensified the need for safe and reliable control strategies. Operating in uncertain and dynamic environments, these systems face significant challenges in maintaining safe trajectories under real-time disturbances and limited computational resources. Ensuring passive safety, i.e. keeping systems within safe operational limits without active intervention, is critical, especially for missions in remote or unpredictable conditions. Model Predictive Control (MPC) has shown promise in handling the essential constraints for such environments. However, effective application of MPC under uncertainty demands innovative approaches to balance computational efficiency with precise state estimation.

This thesis presents a contribution to the development of a framework based on MPC for trajectory generation, using zonotope-based state estimation to ensure passive safety in autonomous systems. By considering the simple example of the unicycle vehicle and future state estimation process, this work serves as a guide for the implementation of passively safe trajectory generation. Furthermore, using Constrained Convex Generator (CCG)s, the thesis presents a minimal conservatism estimation of the progression of a unicycle's uncertain set for a given input.

In simulations, we show the adequate performance of the proposed framework, which maintains safe trajectories under varying levels of uncertainty. This research is a starting point for the design of passively safe trajectories, where optimizations in the state estimation process and further constraint definition can effectively ensure the passive safety of generated trajectories.

Keywords: Model Predictive Control, Trajectory Generation, Passive Safety, State Estimation, Autonomous Systems

Resumo

O rápido crescimento da utilização de sistemas autónomos em áreas como a aeroespacial, automóvel e robótica motiva o desenvolvimento de estratégias de controlo seguras. Garantir segurança passiva, ou seja, manter os sistemas dentro de limites operacionais seguros sem intervenção ativa, é fundamental para os sistemas em ambientes incertos e dinâmicos. A utilização de estratégias de Controlo Preditivo tem mostrado ser uma solução promissora. No entanto, uma aplicação eficaz de Controlo Preditivo em condições de incerteza requer abordagens inovadoras que equilibram eficiência computacional com a precisão da estimativa do estado.

Esta dissertação desenvolve uma estrutura baseada em Controlo Preditivo para a geração de trajetórias, utilizando uma estimativa do conjunto do estado para garantir a segurança passiva em sistemas autónomos. Ao considerar o exemplo simples de um uniciclo e o processo de estimação de estados futuros, este trabalho serve como um guia para a implementação da geração de trajetórias passivamente seguras. Adicionalmente, com recurso à representação de conjuntos por CCGs, esta dissertação apresenta uma estimativa de mínima conservatividade para a progressão do conjunto incerto de um uniciclo.

Os resultados das simulações mostram a eficácia da estrutura proposta, que preserva trajetórias seguras sob diferentes níveis de incerteza. Esta pesquisa é um ponto de partida para o design de trajetórias passivamente seguras, onde otimizações no processo de estimação de estados e a definição de mais restrições podem garantir de forma eficaz a segurança passiva das trajetórias geradas.

Keywords: Controlo Preditivo, Geração de Trajetórias, Segurança Passiva, Estimação do Estado, Sistemas Autónomos

Contents

List of Tables

List of Figures

Acronyms

ADR Active Debris Removal. 1

AV Autonomous Vehicle. 1, 2

CBF Control Barrier Function. 8

CCG Constrained Convex Generator. v, vii, 10, 13, 15, 17, 20–24, 51

CLF Control Lyapunov Function. 7, 8

CMPC Corridor Model Predictive Control. 8

GNC Guidance, Navigation and Control. 1

LTI Linear time-invariant. 26, 50

LTV Linear time-variant. 10, 27

MPC Model Predictive Control. v, xiv, 2–9, 11–15, 24, 27, 28, 30, 32–36, 38–40, 42–44, 49–51

MPPI Model Predictive Path Integral. 8

NMPC Nonlinear Model Predictive Control. 4–6, 8

OSAM On-orbit Servicing, Assembly and Manufacturing. 1

ZCBF Zeroing Control Barrier Function. 8

Chapter 1

Introduction

This Chapter aims to establish the motivation and objectives for the thesis.

1.1 Motivation

In modern society, the rapid rise of autonomous systems has led to significant transformations across various industries, including aerospace, automotive, and robotics. These systems are expected to operate reliably and safely in complex, dynamic environments where the margin for error is minimal. As autonomous systems become more integrated into critical applications, the demand for systems that are not only efficient but also inherently safe has never been more critical.

In the aerospace industry, the recent expansion in space exploration has contributed to an increase in space debris, presenting serious hazards to spacecraft and satellites. According to the United Nations Office for Outer Space Affairs[2], over 17,000 objects have been launched into space since the beginning of space exploration. This large number highlights the significance of space-based technology today, with approximately half of all recorded launches occurring within the last five years alone. The vast accumulation of debris, which includes fragments generated during collisions or mission failures, is especially problematic. NASA reports that there are currently around 25,000 pieces of debris larger than 10 cm and between 300,000 to 500,000 pieces from 1 to 10 cm [3]. The risk of debris-related collisions increases with the rise in space traffic, intensifying the need for accurate debris tracking and avoidance techniques for operational spacecraft.

Addressing this challenge, Active Debris Removal (ADR) technologies have been developed to either prevent or remove hazardous debris. In addition, trajectory generation techniques for collision avoidance have become essential. An example of this is orbital rendezvous, which is crucial in missions involving spacecraft docking or debris retrieval. This operation requires precise trajectory planning and real-time adjustments, as even minor errors can result in mission failure or create additional debris in orbit. The rendezvous is a well-studied manoeuvre and part of the On-orbit Servicing, Assembly and Manufacturing (OSAM) operations that represent a major development in the industry. By enabling in-space operations, OSAM effectively improves space systems' performance and operational lifetimes and provides other possibilities for more complex operations. These operations require various complex safety demands that motivate continuous innovation in the field of Guidance, Navigation and Control (GNC) systems for spacecraft.

Another critical application of safe trajectory generation relates to Autonomous Vehicle (AV) control. The emergence of AVs has the potential to drastically reduce traffic accidents. According to the World Health Organization, road traffic accidents currently result in approximately 1.19 million fatalities each

year, with 20 to 50 million non-fatal injuries worldwide [4]. AVs, equipped with advanced sensors and control algorithms, have the capacity to navigate safely while reducing energy consumption, pollution, and traffic congestion and also improving the accessibility of transportation for a broader segment of the population. However, the complexity of AV control lies in ensuring safe trajectory planning while navigating complex urban environments with pedestrians, cyclists, and other vehicles. This requires real-time adjustments, taking into account road conditions, regulations, and the unpredictable behaviour of nearby drivers. One of the primary difficulties is ensuring that these adjustments prioritize safety while maintaining efficient travel.

In recent decades, with the boost in computing power, the utilization of numerical optimization in control systems has also risen. Model Predictive Control has been widely studied as a control strategy [5][6] consisting of solving an optimization problem that considers the system dynamics and actuator constraints with careful trajectory design to be applied in complex missions. Its ability to handle multivariable systems without adding complexity to the design process and to handle system constraints explicitly during the control process is particularly useful in scenarios that demand both safety and efficiency. For this specific work of designing passively safe trajectories, MPC's predictive nature is a great advantage since it provides optimal control over a future time horizon, considering both the current state and potential future conditions.

In the automotive and aerospace industries, given the safety-critical nature of these applications, MPC has gained popularity as a promising control strategy for safe trajectory generation. The missions using MPC are continuously being studied, with multiple applications already extensively researched, as the use for spacecraft rendezvous [7] [8], automotive operations [9] and mobile robots [10], serving as examples for many more implementations being investigated.

However, despite its advantages, MPC presents challenges, especially concerning computational demand. Solving an optimization problem at each control step can be computationally intensive, particularly for systems with limited onboard processing power, such as those found in autonomous spacecraft and vehicles. Another critical part of the MPC implementation is the correct modelling of the system since many spacecraft and automotive vehicles have complex processes and dynamics that need to be adequately modelled to reduce uncertainty in the control strategy and improve performance.

Using the MPC, this work will strive to guarantee a safe operation, not only incorporating the typical constraints of actuator limits, performance requirements, and active collision avoidance (defining restricted areas) but also considering passive safety. The system will take into account the passive safety of the generated trajectory. Passive safety relates to trajectories that are safe even without active intervention or with an actuation failure. The MPC output may consider paths that briefly involve unsafe or collision routes (corrected later on by a defined action). This kind of trajectory is not passively safe, since a possible actuation failure can lead to an unsafe state, such as an unsafe drift trajectory for example.

In MPC, the system's continuous-time model is typically discretized to operate in discrete time, and the choice of sample time is crucial, as it affects the accuracy of the discrete-time approximation and the computational requirements. Larger sample times effectively reduce computational complexity, which is essential given the limited onboard computational power in multiple missions, but might result in less accurate control actions. On the other hand, shorter sampling times correspond to a more frequent solving of complex optimization problems and a more significant computational load in an already limited context. Safe trajectory generation should minimize these concerns about disturbances, actuation failure, and long sampling time consequences.

1.2 Objectives and Approach

In this thesis, we will develop a Model Predictive Control (MPC)-based framework for generating passively safe trajectories for autonomous systems with uncertain states. This involves encoding a set of conditions into the problem constraints and cost function, which should ensure safety for the designated route by preemptively accounting for possible actuation failures, unmodeled disturbances and other uncertainties. In extreme malfunctioning situations, it is impossible for the method to guarantee safety. The framework employs zonotope-based state estimation to predict future states accurately, balancing safety with minimal conservativeness. This state estimation is used to define the set of safety conditions into the MPC problem.

This work can further validate the use of MPC for critical and complex missions in both aerospace and automotive industries.

1.3 Outline

This document consists of seven chapters. In this first chapter, we address the motivation, the work objectives, and the notation used throughout the document. Afterwards, we introduce technical concepts related to the work. This introduction is essential for comprehending the State of the Art Chapter, where current related work is examined. The subsequent chapter provides the problem formulation, and the following chapter of the Proposed Solution describes everything regarding the methods used to tackle the problem. To conclude the thesis, there is a chapter which shows the results of the proposed solution, followed by a conclusion chapter.

1.4 Notation

Throughout this work, scalar variables are denoted by plain lowercase letters, vectors by bold lowercase letters, and matrices by bold uppercase letters. The identity, null, and ones matrices, each appropriately sized, are denoted by $\mathbf{I}$, $\mathbf{0}$, and $\mathbf{1}$, respectively. Specifically, $\mathbf{I}_n$, $\mathbf{0}_{m\times n}$, and $\mathbf{1}_{m\times n}$ represent the $n \times n$ identity matrix, and the $m \times n$ null and ones matrices, respectively. The absolute value of a real number x is denoted by $\|x\|$. For a vector $\mathbf{v} \in \mathbb{R}^n$, its infinity norm, Euclidean norm, and general p-norm are denoted by $|\mathbf{v}|_\infty$, $|\mathbf{v}|$, and $|\mathbf{v}|_p$, respectively. The i^{th} component of vector $\mathbf{v}$ is denoted by $(\mathbf{v})_i$ and the transpose of a vector $\mathbf{v}$ is denoted by $\mathbf{v}^\top$. Given vector $\mathbf{v}$, $\mathrm{diag}(\mathbf{v})$ denotes the $n \times n$ square diagonal matrix whose diagonal is $\mathbf{v}$, while $\mathrm{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_N)$ denotes the block diagonal matrix whose diagonal blocks are given by matrices $\mathbf{A}_1, \ldots, \mathbf{A}_N$. The Minkowski sum of two sets is denoted by $\oplus$, and the intersection after applying matrix $\mathbf{R}$ to the set is denoted by $\cap_{\mathbf{R}}$.

Chapter 2

Theoretical Background

This chapter introduces the fundamental concepts of model predictive control which will be further explored throughout this work.

2.1 Model Predictive Control

As stated in Section 1.1, the fundamental idea behind the MPC control strategy is iteratively solving an optimization problem (in each control step). The goal is to determine the optimal control inputs that minimize a particular cost function while satisfying the imposed constraints.

2.1.1 Nonlinear Model Predictive Control

Consider a discrete-time system characterized by a state $\mathbf{x}_k \in \mathbb{R}^n$ at time k and a control input $\mathbf{u}_k \in \mathbb{R}^m$ at time k. The process can be described by the model

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k), \tag{2.1}$$

and by an output equation

$$\mathbf{y}_k = g(\mathbf{x}_k, \mathbf{u}_k), \tag{2.2}$$

where $\mathbf{y}_k \in \mathbb{R}^p$, and, in this case of Nonlinear Model Predictive Control (NMPC), f and g are nonlinear functions.

The sequence of optimal control inputs $\mathbf{U} \in \mathbb{R}^{m \times N}$, corresponding state sequence $\mathbf{X} \in \mathbb{R}^{n \times (N+1)}$ and output sequence $\mathbf{Y} \in \mathbb{R}^{p \times (N+1)}$, are defined by

$$\begin{aligned} \mathbf{U} &= [\mathbf{u}_0 \ \mathbf{u}_1 \ ... \ \mathbf{u}_{N-1}], \\ \mathbf{X} &= [\mathbf{x}_0 \ \mathbf{x}_1 \ ... \ \mathbf{x}_N], \\ \mathbf{Y} &= [\mathbf{y}_0 \ \mathbf{y}_1 \ ... \ \mathbf{y}_N], \end{aligned} \tag{2.3}$$

and are obtained by minimizing a cost function $J(\mathbf{Y}, \mathbf{U})$, over a finite prediction horizon N, subject to a set of constraints and given an initial state $\hat{\mathbf{x}}_0$.

To model MPC, we can define the prediction horizon and the control horizon as T_p and T_c, respectively. The prediction horizon determines the timeframe for forecasting future system behavior, offering insight into the system's evolution. The control horizon specifies how far into the future the controller

actively adjusts control inputs. In the considered case, both are considered to be N, as the control horizon is always smaller or equal to the prediction horizon. The cost function, dependent on both $\mathbf{Y}$ and $\mathbf{U}$, serves to evaluate the performance of a provided sequence of control inputs. The following scheme from [1] accurately represents the MPC functioning for a different continuous-time system. Since the system considered by (2.1) is not continuous, we would find both $\mathbf{x}$ and $\mathbf{u}$ discretized in the scheme of Fig. 2.1.

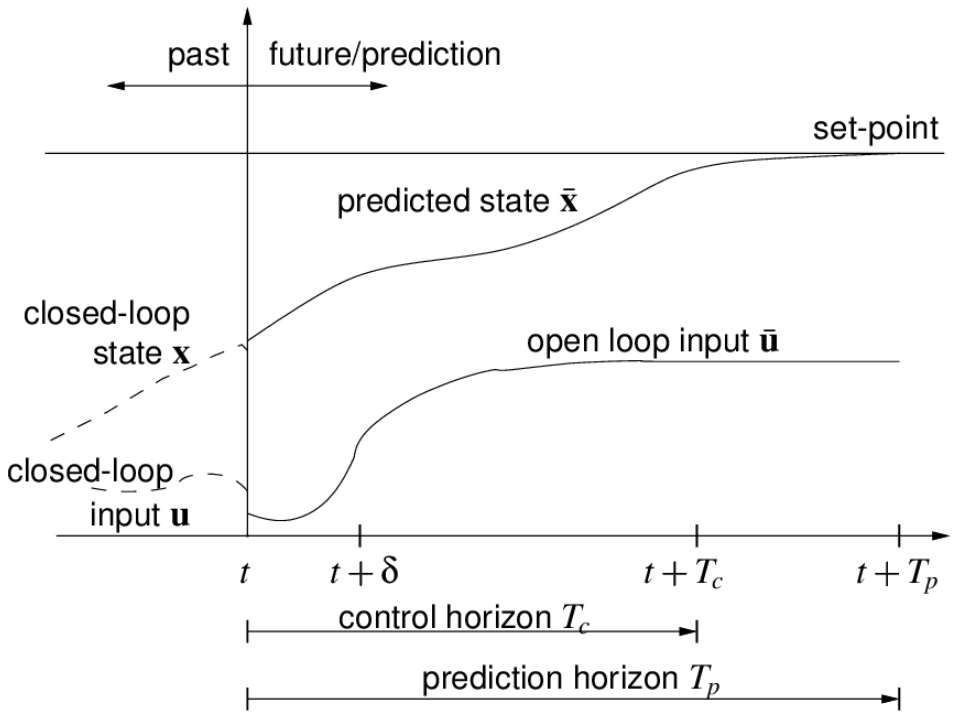


Figure 2.1: Principle of an MPC formulation for a continuous-time system, with a target set-point [1]

With the designated cost function, the state constraint set $\mathcal{X}$, the control input constraint $\mathcal{U}$ and the system output set $\mathcal{Y}$, the MPC solves the problem, giving as output the control input sequence that minimizes the cost function and, therefore, performs best in the defined horizon. The problem can be formulated as such:

$$\begin{aligned}
\underset{\mathbf{Y},\mathbf{U}}{\text{minimize}} \quad & J(\mathbf{Y},\mathbf{U}) \\
\text{s.t.} \quad & \mathbf{x}_{k+1} = f(\mathbf{x}_k,\mathbf{u}_k), && k=0,...,N-1 \\
& \mathbf{y}_{k+1} = g(\mathbf{x}_k,\mathbf{u}_k), && k=0,...,N-1 \\
& \mathbf{x}_k \in \mathcal{X}, && k=0,...,N \\
& \mathbf{y}_k \in \mathcal{Y}, && k=0,...,N \\
& \mathbf{u}_k \in \mathcal{U}, && k=0,...,N-1\ .
\end{aligned} \tag{2.4}$$

The MPC will then solve the problem and apply $\mathbf{u}_0$ to the system, where $\mathbf{u}_0$ is the first control input in the obtained control sequence. In the next time step, the state is updated, the prediction horizon shifts forward, and this process is repeated to find the following optimal control input.

While MPC has proven effective in managing linear systems with constraints, NMPC[1][11] extends its capabilities to address the inherently nonlinear nature of many real-world processes. NMPC can handle nonlinearity in both the system dynamics and the objective function. This adjustment opens up new possibilities for tackling a broader range of control problems, including those where linear approximations fall short.

This extension of the MPC excels in handling nonlinear dynamics, making it well-suited for controlling complex processes. However, it poses some implementation challenges. The initial challenge stems directly from the nonlinearity of the plant, necessitating the real-time resolution of non-convex optimal control problems. Failing to address this challenge significantly hampers the application of nonlinear model predictive control. The second challenge involves the necessary task of developing robust versions of model predictive control capable of managing inherent uncertainties, which arise from either model inaccuracies or unmeasured disturbances.

2.1.2 Linear Model Predictive Control

Due to the substantial computational complexity of NMPC, we can often consider linearizations of more complex models, effectively reducing the complexity at the expense of some degree of precision in the designated model dynamics. Assuming the models f and g considered in (2.1) and (2.2) are linear, the system would then be characterized by the linear state model

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k, \tag{2.5}$$

$$\mathbf{y}_k = \mathbf{C}\mathbf{x}_k + \mathbf{D}\mathbf{u}_k, \tag{2.6}$$

where generally $\mathbf{D} = 0$ since only for systems with instantaneous behaviours $\mathbf{D}$ is not null. This model will not consider the existence of disturbances in the system.

An effective and common cost function for the optimization problem is the quadratic cost, consisting of a sum of a quadratic cost on the state and control inputs and a terminal cost on the final state,

$$J(\mathbf{X}, \mathbf{U}) = \sum_{k=0}^{N-1} \mathbf{x}_k^\intercal \mathbf{Q}\mathbf{x}_k + \mathbf{u}_k^\intercal \mathbf{R}\mathbf{u}_k + \mathbf{x}_N^\intercal \mathbf{Q_f}\mathbf{x}_N, \tag{2.7}$$

where $\mathbf{Q}$ and $\mathbf{Q}_f$ are positive semidefinite matrices, and $\mathbf{R}$ is a positive definite matrix. The matrices are adjustable parameters, which makes it possible, in the considered problem, to tune the relative importance of closely following the defined trajectory and minimizing actuation like fuel consumption. Increasing the elements in $\mathbf{R}$ compared to $\mathbf{Q}$ in the cost function intensifies the penalization of the control variable, leading to a constrained actuator action in the optimal solution.

Other modifications to the cost functions may be considered, including adjustments related to reference tracking ($\mathbf{x}_{ref}$ or $\mathbf{y}_{ref}$). This is particularly relevant in the context of the problem of trajectory generation under investigation in this study. Another approach involves formulating the problem by considering the system output instead of the state. In this scenario, the cost function becomes a function of $\mathbf{U} \in \mathbb{R}^{m \times N}$ and $\mathbf{Y} \in \mathbb{R}^{p \times (N+1)}$. The problem would then be articulated as follows,

$$\begin{aligned}
\min_{\mathbf{Y},\mathbf{U}} \quad & \sum_{k=0}^{N-1} (\mathbf{y}_k - \mathbf{y}_{ref})^\intercal \mathbf{Q} (\mathbf{y}_k - \mathbf{y}_{ref}) + \mathbf{u}_k^\intercal \mathbf{R}\mathbf{u}_k + (\mathbf{y}_N - \mathbf{y}_{ref})^\intercal \mathbf{Q}_f (\mathbf{y}_N - \mathbf{y}_{ref}), \\
\text{s.t.} \quad & \mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k, \quad k = 0, ..., N-1 \\
& \mathbf{y}_k = \mathbf{C}\mathbf{x}_k, \quad k = 0, ..., N \\
& \mathbf{x}_k \in \mathcal{X}, \quad k = 0, ..., N \\
& \mathbf{y}_k \in \mathcal{Y}, \quad k = 0, ..., N \\
& \mathbf{u}_k \in \mathcal{U}, \quad k = 0, ..., N-1 \ .
\end{aligned} \tag{2.8}$$

where $\mathcal{Y}$ represents the constraint set for the system output.

Chapter 3

State-of-the-art

The following Chapter delves into the exploration and analysis of state-of-the-art methods in the context of employing MPC for the secure generation of trajectories. Serving as an introduction to the state-of-the-art, research studies will be discussed and evaluated to gain comprehensive insights into the current landscape of trajectory generation and safety considerations. Considering the problem at hand, ensuring the safety of the trajectory relies significantly on precise state estimation and handling unknown disturbances. Both aspects are focal points of the following research studies.

3.1 Robust control

Robust control is a branch of control theory that deals with systems operating under uncertainty. In real-world applications, models often contain uncertainties due to unmodeled dynamics, external disturbances, or parameter variations. Linearization of nonlinear models, for example, reduces the problem complexity while frequently adding uncertainty to the system due to the model approximation. Robust control methods focus on designing a controller to maintain proper system function, stability, and performance while considering bounded uncertainties and disturbances. The main objective is to secure robust control, ensuring the system behaves as intended.

Model predictive control can be used to implement robust control. In the work [12], MPC is presented as a robust solution to the control of constrained, linear, discrete-time systems affected by bounded disturbances. A crucial part of this study is defining a disturbance invariant set inside the state space. The idea is that no matter the disturbances, the system trajectory will not escape the invariant set, ensuring containment under all admissible uncertainties. Although an invariant set simplifies dealing with disturbances, large invariant sets can limit system performance since they could pose overly conservative solutions. The authors incorporate a Control Lyapunov Function (CLF), valued at zero in the invariant set, acting as the origin of the uncertain system. In the proposed solution, the initial state is not fixed as the system's current state, introducing it as a decision variable in the control problem to be solved. This allows the proof of robust exponential stability of the invariant set for the controlled system. The control output results in a "tube" of possible trajectories that account for uncertainties. An extension for nonlinear systems was further developed in [13], and tube-based methods are discussed in [14]. The nonlinear implementation is comprised of a nominal MPC, generating a reference trajectory, and an ancillary MPC, which maintains the system's actual state within a tube around the reference trajectory. It acts as a secondary control mechanism that complements the primary controller by providing additional stabilization. Monte Carlo methods [15] are used to tighten the constraint sets, reducing the conservativeness associated with the solution.

Another possible implementation of robust MPC is to consider sampling-based control to mitigate the effect of uncertainties and disturbances. In [16], a solution for robust sampling-based MPC is presented, which combines nonlinear tube MPC [13] and Model Predictive Path Integral (MPPI) control. MPPI is the sampling-based model predictive control algorithm implemented in the referenced work, using the principles of MPC and Path Integral theory frequently associated with stochastic optimal control. The authors propose a framework of information-theoretic MPPI to deal with the problem of disturbances in the control input. Considering this approach, the challenge becomes approximating the inputs to the optimal control inputs. This is done by iterative importance sampling, as for many other sampling-based methods. A review of different MPPI methods can be found in [17]. These methods rely on sampling and evaluating many control sequences and respective trajectories in each optimization step, making the process very computationally complex. Many algorithms consider reusing parts of the previous control sequence to reduce computation time since it is closer to the optimal solution. While this can effectively reduce computation time, the control inputs can become inappropriate for the new state, given some significant disturbances that contradict the assumption of the system's state and lead to undesirable outcomes.

3.1.1 Corridor MPC

In the work [18], it is studied Corridor Model Predictive Control (CMPC) a framework for safe and optimal trajectory tracking by combining NMPC and a Zeroing Control Barrier Function (ZCBF) formulation. A Control Barrier Function (CBF) [19] is a mathematical tool used in control theory to enforce safety constraints in dynamic systems by restraining the state to a designated safe set. In simple terms, the CBF represents a function whose derivative consistently rises as it moves away from a safe set, ensuring the invariance of the set and providing a safety bound on the system's state set. ZCBFs distinguish themselves from regular CBFs by having the property that the function reaches zero only at a predefined equilibrium point in the safe set, while in CBFs, the value is zero throughout the set. This property ensures the system converges to a desired state while satisfying safety constraints.

This novel method guarantees safety for the continuous-time system by limiting the state to a certain corridor surrounding the reference trajectory. Aiming to surpass the results of tube-based robust MPC, the strategy proposed by the research consists of using sampled-data ZCBFs, incorporating robustness margins in the ZCBF framework:

$$\begin{aligned} \sup_{u \in \mathbb{U}} \quad & [\mathrm{L}_{f_c} h(\tilde{x}, t) + \mathrm{L}_{g_c} h(\tilde{x}, t) u + \frac{\partial h(\tilde{x}, t)}{\partial t} + \alpha(h(\tilde{x}, t))] \geq 0, \\ & \forall \tilde{x} \in \mathcal{D}, t \in \mathbb{R}_{\geq 0}, \end{aligned} \tag{3.1}$$

where the system is defined by f_c and g_c, and the Lie derivatives of h along f_c and g_c, are L_{f_c} and L_{g_c}, respectively. α is considered to be Lipschitz continuous.

Lie derivatives describe how a tensor field changes along the flow of a vector field. In this case, the Lie derivatives of the barrier function are analyzed, modelling the variation along the functions that represent the system. Lipschitz continuity characterizes the behaviour of a function concerning how much its output can change concerning changes in its input. The researchers use these parameters to formulate and test the ZCBF.

Another aspect to note is the inclusion of a CLF in the cost function, as seen in previous implementations of robust MPC. The cost function (including the CLF) ensures that the predicted states are not only driven toward the reference but also remain within certain constraints.

3.2 Simulation Methods

Dealing with uncertainties in the system's dynamics, state uncertainty, and uncertain or dynamic environments is a major challenge in any critical mission where safety is fundamental. This problem has been a focus of study for many years, with multiple simulation solutions surging as a possible way to tackle this challenge.

A standard research direction has been solutions that rely on computing collision probabilities for different generated trajectories, frequently associated with the problem of robot motion planning. In [20] and [21], scenario-based MPC is employed to tackle the issue of trajectory safety. Scenario-based MPC serves as a solution to trajectory optimization affected by uncertainties, namely in the environment. The work [20] considers an uncertainty model that represents probabilistic constraints, predicting the movement of dynamic objects while accounting for uncertainty in their future positions and velocities. From this model, different scenarios are generated by sampling multiple potential obstacle positions from the probability distribution, turning the probabilistic constraints into deterministic. As such, the numerous scenarios account for a potential future position of an obstacle at a given time step. These scenarios represent different possible realities, which could be a very high number and computationally demanding process. During the optimization process by the MPC, the generated trajectories are evaluated in the different scenarios for parameters defined in the cost function and also for their collision risk in each scenario. The trajectory evaluation can be done by constraining the collision probability of the result to be below a defined value, as proposed in [20] where Monte Carlo [15] sampling was used to evaluate the results. In [21], another solution is proposed where the risk of collision is present as a parameter in the cost function, guaranteeing a feasible solution since there is no constraint value for collision risk. A collision probability constraint could possibly be set to value zero, where the trajectory considered would have zero risk of colliding. However, this could frequently prove to be unfeasible in these problems of uncertain and dynamic environments. This work estimates the collision probabilities using Monte Carlo simulation associated with a Kalman Filter. The filter reduces the statistical noise created by running the simulations for a finite number of samples. While this approach of scenario-based MPC is a powerful method for handling uncertainties, it comes with several challenges, such as the number of scenarios considered, since high computational complexity is associated with solving optimization problems for many scenarios. With fewer scenarios, the set of possibilities might not be sufficiently covered, and some outcomes might not be portrayed in the scenarios. Another challenge to consider is the accuracy of the uncertainty model, which could lead to unsafe behaviours when underestimating the risks or to overly conservative solutions when considering models with more significant uncertainty.

Simulation of possible collisions to assess a trajectory's safety is a straightforward approach which is not only considered in scenario-based MPC. Monte Carlo methods are frequently used in these problems, as showcased in [22] for safety assessment in uncertain and dynamic environments. The work consists of sampling using sequential Monte Carlo methods to generate control inputs for each object in the environment, generating multiple trajectories. The collision probability of the robot trajectory is estimated by Monte Carlo simulation, which samples the possible states and trajectories of the objects, attributes a weight to each sample based on a goal function and defines the collision probability as the sum of the weights of all samples that lead to a collision. As in previously analyzed works, the safety assessment metric is based on collision risk.

A Monte Carlo approach can be used for trajectory generation and optimization, motion planning and other topics, maintaining similar methods of generating samples from uncertain inputs, positions or systems states, evaluating each of those samples in a deterministic computation and combining the individual simulations to return the final result. [23] introduces variance-reduced Monte Carlo applied to motion planning to more accurately compute collision probability, striving to achieve more reliable

results with fewer samples. [24] [25] refer to other use examples of Monte Carlo, this time applicable in the safety assessment of a road environment.

3.3 State estimation

Complex operations where autonomous vehicles and other systems operate can lead to difficult state estimation processes. Either by the sensor malfunctioning, noisy measures, or lack of available sensors, the system's state might not be exactly known. This leads to the definition of an uncertain state and state estimation procedures to provide a set of possible states that includes the actual system state.

One general problem is locating an autonomous vehicle for an environment with restricted GPS availability. Estimating the state given range and bearing measurements is a common problem tackled in the literature that develops set representations to enclose the uncertain state given by noisy measurements.

Typical solutions in the literature involve over-approximations of these sets for range only sensor such as interval partition in [26] or using ellipsoid sets in [27]. With these approaches, the solution is often characterized by excessive conservatism. An alternative method approximates the measurement set by polytopes, represented using hyper-plane definitions [28] or constrained zonotopes [29]. In [30] it is developed a novel set representation motivated by the need for unbounded set representations for bearing measurement and mixed representations for range and bearing data.

Extensive research in the field of set representations for Linear time-variant (LTV) systems in discrete time has introduced various methods for state estimation. Among these are interval arithmetic studied in [31], zonotopes described in [32] [33], constrained zonotopes [29], ellipsoids [34], polytopes [35] and more recently, CCGs in [30]. These techniques propagate and update set-valued estimates, often approximating the nonconvex annulus, given by the noisy measurements, using convex sets. For the problem considering nonlinear systems, these methods require linear approximations of the dynamics as explored in the following works, [36], [37], [38], [39], and [40].

Another research direction is the problem concerning autonomous vehicles, these are often addressed through stochastic estimation using Kalman filters. The works [41] and [42] presented solutions involving nonlinear Extended Kalman Filters for vehicle state estimation. In [43], a Kalman filter is applied to the dynamics which are transformed to an LTV system.

Chapter 4

Problem Formulation

In this thesis, we consider the control problem of a unicycle with state uncertainty. The nonlinear function of the unicycle vehicle dynamics is expressed as

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) = \begin{bmatrix} v \cdot \cos\theta \\ v \cdot \sin\theta \\ \omega \end{bmatrix}, \tag{4.1}$$

where $\mathbf{x} \in \mathbb{R}^3 : \mathbf{x} = [x\ y\ \theta]^\top$ represents the uncertain state, and $\mathbf{u} \in \mathbb{R}^2$ denotes a certain input composed by velocity v and angular velocity ω.

Through linearization and discretization, the objective is to find a suitable model to estimate the state progression and impose constraints to generate passively safe trajectories based on this estimation. The proposed solution will provide the state estimation procedure to include in the MPC formulation. At time k, the uncertain state set is denoted by X_k, with centre $\mathbf{x}_k$, and the uncertainty distribution of the state. These are estimated before constructing the optimization problem. The MPC is formulated based on $\mathbf{x}_{k+j}$, with j denoting the time steps of the MPC formulation, from 0 to horizon N. The vector $\mathbf{x}_{k+j}$ represents then the estimates of the state used to construct the constraints and the cost function of the optimization.

Considering then, a discretized system to represent the problem's dynamics,

$$\mathbf{x}_{k+j+1} = \mathbf{A}\mathbf{x}_{k+j} + \mathbf{B}\mathbf{u}_{k+j}, \tag{4.2}$$

where $\mathbf{x}_{k+j}$ is represented by the algorithm's set estimation, and matrices $\mathbf{A}$ and $\mathbf{B}$ are the product of computations to simplify the model in (4.1).

The general representation of the MPC optimization problem will consider a reference tracking quadratic cost function that assumes a determined position in 2-D space $\mathbf{p}_{ref}$ as the vehicle reference goal. The position $\mathbf{p}_{k+j}$ represents the centre of the uncertain 2-D set and is obtained from the state as

$$\mathbf{p}_{k+j} = \mathbf{C}\mathbf{x}_{k+j}, \qquad \mathbf{C} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{4.3}$$

and the cost function to be minimized refers to the difference of the reference position and $\mathbf{p}_{k+j}$.

The prediction horizon and the control horizon have the same duration N. During this horizon, the trajectory to be optimized must satisfy the constraints imposed by the combination of obstacles and model uncertainties to ensure passive safety. The obstacles considered in the problem restrict the position but not the angle of orientation, assuming the form of a wall by limiting the position state space to

$$\mathbf{o} \leq \mathbf{p} \leq \mathbf{q}, \qquad \mathbf{o} = \begin{bmatrix} o_x \\ o_y \end{bmatrix}, \mathbf{q} = \begin{bmatrix} q_x \\ q_y \end{bmatrix}, \tag{4.4}$$

where it is important to note that $\mathbf{p} \in \mathbb{R}^2$ denotes any position in the relevant uncertain set, effectively limiting the whole system to progress past the obstacle.

The MPC's optimization problem can be explicitly represented as,

$$\begin{aligned}
\min_{\mathbf{P},\mathbf{U}} \quad & \sum_{j=0}^{N-1} (\mathbf{p}_{k+j} - \mathbf{p}_{ref})^\top \mathbf{Q} (\mathbf{p}_{k+j} - \mathbf{p}_{ref}) + \mathbf{u}_{k+j}^\top \mathbf{R} \mathbf{u}_{k+j} + (\mathbf{p}_{k+N} - \mathbf{p}_{ref})^\top \mathbf{Q}_f (\mathbf{p}_{k+N} - \mathbf{p}_{ref}), \\
\text{s.t.} \quad & \mathbf{x}_{k+j+1} = \mathbf{A}\mathbf{x}_{k+j} + \mathbf{B}\mathbf{u}_{k+j}, \quad j = 0, ..., N-1 \\
& X_{k+j+1} = \mathrm{g}(X_{k+j}, \mathbf{x}_{k+j}, \mathbf{u}_{k+j}), \quad j = 0, ..., N-1 \\
& \mathbf{p}_{k+j} = \mathbf{C}\mathbf{x}_{k+j}, \quad j = 0, ..., N \\
& \mathbf{o} \leq \mathbf{p} \leq \mathbf{q}, \quad \mathbf{p} \in X_{k+j}, \quad j = 0, ..., N \\
& \mathbf{x}_{k+j} \in X_{k+j}, \quad j = 0, ..., N \\
& X_{k+j} \subset \mathcal{X}, \quad j = 0, ..., N \\
& \mathbf{p}_{k+j} \in \mathcal{P}, \quad j = 0, ..., N \\
& \mathbf{u}_{k+j} \in \mathcal{U}, \quad j = 0, ..., N-1
\end{aligned} \tag{4.5}$$

where function $\mathrm{g}(X_{k+j}, \mathbf{x}_{k+j}, \mathbf{u}_{k+j})$ refers to the proposed solution for state estimation.

The objective of this thesis is to properly define the state estimation function and the other constraints, such as obstacle avoidance, to ensure the passive safety of the trajectories generated by the problem stated here. This optimization problem is solved for an optimal sequence of control inputs. The first input of the sequence, $\mathbf{u}_k$, is the output of the problem.

Chapter 5

Proposed Solution

This chapter introduces the proposed solution to address the problem described in the previous chapter, which is associated with the unicycle model.

The first section shows the mathematical formulation associated with set estimation. In the following section, a first approach to solve this problem, identified as set propagation algorithm, serves as a minimal conservativeness solution to estimate the next state of an uncertain set given a certain input. In Section 5.3, the unicycle model is properly linearized and discretized, serving as the foundation for the proposed solution considered in Section 5.4, where the solution is defined and explained, completing the goal of ensuring the passive safety of the generated trajectories.

5.1 Set estimation

This section introduces the set representations used in this thesis and the respective mathematic operations used for state estimation. Many different set representations are studied to tackle the problem of state estimation with uncertainties. For the MPC control algorithm, zonotopes represent the state space, while in the set propagation algorithm, CCG is the chosen representation for the uncertain set.

Firstly, the zonotope formulation is introduced along with the corresponding set operations, and the following are the CCGs and associated computations. Lastly, this section comments on the state estimation problem in this work.

5.1.1 Set representations

A zonotope is a convex polytope in n-dimensional space, defined as the Minkowski sum of line segments or generators [32], where each generator represents a vector in the space. Its convex property makes it suitable for encoding the set of possible states into the MPC formulation. Reviewing the definitions in [44][29], the zonotope Z can be defined using a generator-representation, as

$$Z = \{\mathbf{G}\xi + \mathbf{c} : \|\xi\|_\infty \leq 1\}, \tag{5.1}$$

where $\mathbf{c} \in \mathbb{R}^n$ represents the center, and $\mathbf{G} \in \mathbb{R}^{n \times n_g}$ is composed by n_g column vectors, the generators, $\mathbf{g}_1, \ldots, \mathbf{g}_{n_g} \in \mathbb{R}^n$. The set Z can be defined by the tuple $Z = (\mathbf{G}, \mathbf{c}) \subset \mathbb{R}^n$.

Regarding state estimation, there are three basic set operations essential to the problem and handling these set operations accurately and efficiently is critical for any set representation. These set operations: affine map, Minkowski sum and intersection after a linear map, are shown below in (5.2),

(5.3), (5.4), respectively. Considering the sets $Z, W \subset \mathbb{R}^n, Y \subset \mathbb{R}^k$, matrix $\mathbf{R} \in \mathbb{R}^{k \times n}$ and vector $\mathbf{t} \in \mathbb{R}^k$, the operations can be defined as

$$\mathbf{R}Z + \mathbf{t} \equiv \{\mathbf{R}\mathbf{z} + \mathbf{t} : \mathbf{z} \in Z\}, \tag{5.2}$$

$$Z \oplus W \equiv \{\mathbf{z} + \mathbf{w} : \mathbf{z} \in Z, \mathbf{w} \in W\}, \tag{5.3}$$

$$Z \cap_{\mathbf{R}} Y \equiv \{\mathbf{z} \in Z : \mathbf{R}\mathbf{z} \in Y\}, \tag{5.4}$$

where the affine map and the Minkowski sum can be exactly computed through zonotopes. The intersection, however, generates an output that cannot be exactly reproduced in this representation. Since zonotopes are not closed under an intersection, this operation between zonotopes does not result in a zonotope. It requires the computation of a zonotope enclosure to approximate the result.

The affine map and the Minkowski sum can be defined as set operations involving zonotopes and computed exactly. Let zonotopes $Z, W \subset \mathbb{R}^n$ be $Z = (\mathbf{G}_{\mathcal{Z}}, \mathbf{c}_{\mathcal{Z}})$, $W = (\mathbf{G}_{\mathcal{W}}, \mathbf{c}_{\mathcal{W}})$, matrix $\mathbf{R} \in \mathbb{R}^{k \times n}$, and vector $\mathbf{t} \in \mathbb{R}^k$. Applying the considered set operations, it shows,

$$\mathbf{R}Z + \mathbf{t} = (\mathbf{R}\mathbf{G}_{\mathcal{Z}}, \mathbf{R}\mathbf{c}_{\mathcal{Z}} + \mathbf{t}) \subset \mathbb{R}^k, \tag{5.5}$$

$$Z \oplus W = ([\mathbf{G}_{\mathcal{Z}}\ \mathbf{G}_{\mathcal{W}}], \mathbf{c}_{\mathcal{Z}} + \mathbf{c}_{\mathcal{W}}) \subset \mathbb{R}^n, \tag{5.6}$$

efficiently representing the outcome of the affine map and Minkowski sum. Zonotopes and associated set operations are used in the MPC algorithm of this work since they correspond to the least computationally complex set representation described here.

Given the zonotope limitations, the author provides an extension to the set representation in [29]. In the referenced work, constrained zonotopes are introduced as an expansion of zonotopes which is able to effectively compute more complex set operations such as the intersection while maintaining the zonotope's computational efficiency. Based on the zonotope configuration, a constrained zonotope differs by allowing linear equality constraints on ξ.

Considering $Z \subset \mathbb{R}^n$, a constrained zonotope Z is characterized by a tuple $(\mathbf{G}, \mathbf{c}, \mathbf{A}, \mathbf{b}) \in \mathbb{R}^{n \times n_g} \times \mathbb{R}^n \times \mathbb{R}^{n_c \times n_g} \times \mathbb{R}^{n_c}$ which results in the expression,

$$Z = \{\mathbf{G}\xi + \mathbf{c} : \|\xi\|_\infty \leq 1, \mathbf{A}\xi = \mathbf{b}\}, \tag{5.7}$$

where the term $\mathbf{A}\xi = \mathbf{b}$ reflects the linear equality constraints. Using this formulation, denominated by constrained generator representation, the set operations (5.2)-(5.4) prove to be easily propagated.

Given three constrained zonotopes $Z = (\mathbf{G}_{\mathcal{Z}}, \mathbf{c}_{\mathcal{Z}}, \mathbf{A}_{\mathcal{Z}}, \mathbf{b}_{\mathcal{Z}}) \subset \mathbb{R}^n, W = (\mathbf{G}_{\mathcal{W}}, \mathbf{c}_{\mathcal{W}}, \mathbf{A}_{\mathcal{W}}, \mathbf{b}_{\mathcal{W}}) \subset \mathbb{R}^n, Y = (\mathbf{G}_{\mathcal{Y}}, \mathbf{c}_{\mathcal{Y}}, \mathbf{A}_{\mathcal{Y}}, \mathbf{b}_{\mathcal{Y}}) \subset \mathbb{R}^k$, matrix $\mathbf{R} \in \mathbb{R}^{k \times n}$, and vector $\mathbf{t} \in \mathbb{R}^k$, the outcome of the set operations can be computed in a constrained zonotope representation as

$$\mathbf{R}Z + \mathbf{t} = (\mathbf{R}\mathbf{G}_{\mathcal{Z}}, \mathbf{R}\mathbf{c}_{\mathcal{Z}} + \mathbf{t}, \mathbf{A}_{\mathcal{Z}}, \mathbf{b}_{\mathcal{Z}}) \subset \mathbb{R}^k, \tag{5.8}$$

$$Z \oplus W = \left([\mathbf{G}_{\mathcal{Z}}\ \mathbf{G}_{\mathcal{W}}], \mathbf{c}_{\mathcal{Z}} + \mathbf{c}_{\mathcal{W}}, \begin{bmatrix} \mathbf{A}_{\mathcal{Z}} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{\mathcal{W}} \end{bmatrix}, \begin{bmatrix} \mathbf{b}_{\mathcal{Z}} \\ \mathbf{b}_{\mathcal{W}} \end{bmatrix}\right) \subset \mathbb{R}^n, \tag{5.9}$$

$$Z \cap_{\mathbf{R}} Y = \left([\mathbf{G}_{\mathcal{Z}} \ \mathbf{0}], \mathbf{c}_{\mathcal{Z}}, \begin{bmatrix} \mathbf{A}_{\mathcal{Z}} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{\mathcal{W}} \\ \mathbf{R}\mathbf{G}_{\mathcal{Z}} & -\mathbf{G}_{\mathcal{Y}} \end{bmatrix}, \begin{bmatrix} \mathbf{b}_{\mathcal{Z}} \\ \mathbf{b}_{\mathcal{W}} \\ \mathbf{c}_{\mathcal{Y}} - \mathbf{R}\mathbf{c}_{\mathcal{Z}} \end{bmatrix} \right) \subset \mathbb{R}^n, \tag{5.10}$$

corresponding to the affine map, Minkowski sum and intersection after a linear map, respectively.

Constrained zonotopes offer greater flexibility compared to zonotopes, providing simple expressions for propagation through these critical set operations. The better accuracy when representing the set comes at the cost of being more complex than the zonotope representation. The dimensionality of ξ is often increased during the set operations, which requires more complexity to be expressed but, in turn, can lead to less conservative approximations. In both zonotopes, constrained zonotopes, and set representations in general, processes to reduce complexity are vital to improving efficiency in the computations.

Based on the constrained zonotope configuration, CCGs are properly defined in [30] along with the respective set operations, which will be shortly described here. This paper addresses the problem of estimating the state of a dynamical system described by a linear model that relies on nonlinear measurements, specifically a combination of bearing and range measurements. The CCG formulation relies on the fact that the constrained zonotopes generators must belong to a convex set, but are not restricted to the unit ℓ_∞ norm ball as $\|\xi\|_\infty \leq 1$ in (5.7). This means the generators can be a part of unit ℓ_∞ norm ball, unit ℓ_2 norm ball, cones and other convex sets. Consider $Z \subset \mathbb{R}^n$, a constrained convex generator Z can be defined as

$$Z = \{\mathbf{G}\xi + \mathbf{c} : \mathbf{A}\xi = \mathbf{b}, \xi \in \mathcal{C}_1 \times ... \times \mathcal{C}_{n_p}\}, \tag{5.11}$$

with $\mathbf{G} \in \mathbb{R}^{n \times n_g}$, $\mathbf{c} \in \mathbb{R}^n$, $\mathbf{A} \in \mathbb{R}^{n_c \times n_g}$, $\mathbf{b} \in \mathbb{R}^{n_c}$ and $\{\mathcal{C}_1 \times ... \times \mathcal{C}_{n_p}\} = \mathfrak{C}$ that represents the convex sets of which the generators belong to. It can be characterized in short form as a tuple, $Z = (\mathbf{G}, \mathbf{c}, \mathbf{A}, \mathbf{b}, \mathfrak{C})$.

By using ℓ_2 norm unit balls, ℓ_∞ norm balls, and cones, this representation shows to have minimum conservatism, less than constrained zonotopes in multiple applications. Additionally, this solution can represent more cases, namely unbounded sets, generated by cones.

Let $Z, W \subset \mathbb{R}^n$, and $Y \subset \mathbb{R}^k$ be CCGs described by the tuples, $Z = (\mathbf{G}_{\mathcal{Z}}, \mathbf{c}_{\mathcal{Z}}, \mathbf{A}_{\mathcal{Z}}, \mathbf{b}_{\mathcal{Z}}, \mathfrak{C}_{\mathcal{Z}})$, $W = (\mathbf{G}_{\mathcal{W}}, \mathbf{c}_{\mathcal{W}}, \mathbf{A}_{\mathcal{W}}, \mathbf{b}_{\mathcal{W}}, \mathfrak{C}_{\mathcal{W}})$, $Y = (\mathbf{G}_{\mathcal{Y}}, \mathbf{c}_{\mathcal{Y}}, \mathbf{A}_{\mathcal{Y}}, \mathbf{b}_{\mathcal{Y}}, \mathfrak{C}_{\mathcal{Y}})$, and also matrix $\mathbf{R} \in \mathbb{R}^{k \times n}$, and vector $\mathbf{t} \in \mathbb{R}^k$. The three set operations can be expressed as

$$\mathbf{R}Z + \mathbf{t} = (\mathbf{R}\mathbf{G}_{\mathcal{Z}}, \mathbf{R}\mathbf{c}_{\mathcal{Z}} + \mathbf{t}, \mathbf{A}_{\mathcal{Z}}, \mathbf{b}_{\mathcal{Z}}, \mathfrak{C}_{\mathcal{Z}}) \subset \mathbb{R}^k, \tag{5.12}$$

$$Z \oplus W = \left([\mathbf{G}_{\mathcal{Z}} \ \mathbf{G}_{\mathcal{W}}], \mathbf{c}_{\mathcal{Z}} + \mathbf{c}_{\mathcal{W}}, \begin{bmatrix} \mathbf{A}_{\mathcal{Z}} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{\mathcal{W}} \end{bmatrix}, \begin{bmatrix} \mathbf{b}_{\mathcal{Z}} \\ \mathbf{b}_{\mathcal{W}} \end{bmatrix}, \{\mathfrak{C}_{\mathcal{Z}}, \mathfrak{C}_{\mathcal{W}}\} \right) \subset \mathbb{R}^n, \tag{5.13}$$

$$Z \cap_{\mathbf{R}} Y = \left([\mathbf{G}_{\mathcal{Z}} \ \mathbf{0}], \mathbf{c}_{\mathcal{Z}}, \begin{bmatrix} \mathbf{A}_{\mathcal{Z}} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{\mathcal{W}} \\ \mathbf{R}\mathbf{G}_{\mathcal{Z}} & -\mathbf{G}_{\mathcal{Y}} \end{bmatrix}, \begin{bmatrix} \mathbf{b}_{\mathcal{Z}} \\ \mathbf{b}_{\mathcal{W}} \\ \mathbf{c}_{\mathcal{Y}} - \mathbf{R}\mathbf{c}_{\mathcal{Z}} \end{bmatrix}, \{\mathfrak{C}_{\mathcal{Z}}, \mathfrak{C}_{\mathcal{Y}}\} \right) \subset \mathbb{R}^n, \tag{5.14}$$

in a very similar manner as the constrained zonotopes, with the difference being the specification of the generators in each CCG, when in constrained zonotopes only the unit ℓ_∞ norm ball was considered.

The introduced constrained convex generators produce tighter enclosures than constrained zonotopes, namely in nonconvex sets. This set representation is used in the first approach of the set propagation algorithm, to allow a minimum conservativeness solution. However, the additional operations and complexity introduced by the CCGs result in a more computationally demanding task, when compared to alternative solutions, such as zonotopes, the chosen set representation for the MPC implementation.

5.1.2 State estimation

Since this work involves uncertain states and/or systems with disturbances, it is important to recognize the set-based state estimation problem which is tackled by producing enclosures using the set representations. Considering a discrete-time linear system,

$$\begin{aligned} \mathbf{x}_k &= \mathbf{A}_k\mathbf{x}_{k-1} + \mathbf{B}_k\mathbf{u}_k + \mathbf{D}_w\mathbf{w}_{k-1} \\ \mathbf{y}_k &= \mathbf{C}\mathbf{x}_k + \mathbf{E}_v\mathbf{v}_k \end{aligned} \tag{5.15}$$

where the system state is $\mathbf{x}_k \in \mathbb{R}^{n_x}$, the output is $\mathbf{y}_k \in \mathbb{R}^{n_y}$, with $\mathbf{u}_k \in \mathbb{R}^{n_u}$ denoting a known input, the disturbance is $\mathbf{w}_{k-1} \in \mathbb{R}^{n_w}$ and $\mathbf{v}_k \in \mathbb{R}^{n_v}$ is the measurement error. Assuming the disturbances and measurement errors are bounded and belong to a compact set, $(\mathbf{w}_k, \mathbf{v}_k) \in W \times V$ as well as the initial state $\mathbf{x}_0 \in X_0$, with X_0 as a compact set. The objective is to use the set operations to compute an enclosure of the estimated state set $\hat{X}_k$, which is represented as

$$\hat{X}_k = (\mathbf{A}_k\hat{X}_{k-1} + \mathbf{B}_k\mathbf{u}_k \oplus \mathbf{D}_wW) \cap_{\mathbf{C}} (\mathbf{y}_k - \mathbf{E}_vV), \tag{5.16}$$

with $\hat{X}_0 = X_0 \cap_{\mathbf{C}} (\mathbf{y}_0 - \mathbf{E}_vV)$. The expression (5.16) describes the formula to compute an estimate of the state set, at each time step k. In simpler form, it consists of applying the model dynamics to the previous state set estimate and intersecting such set with the measurements $\mathbf{y}_k$. In most cases, it is very challenging or impossible to compute (5.16) exactly. Therefore, set representations are used to provide enclosures $\mathcal{O}_k$ which include the state set $\mathcal{O}_k \supset \hat{X}_k$. A critical point of optimization is the tightness of the enclosures, striving to have the least over-approximation possible of the state set to improve the accuracy of the estimate. The enclosures $\mathcal{O}_k$ can replace $\hat{X}_k$ in (5.16), leading to

$$\mathcal{O}_k \supset (\mathbf{A}_k\mathcal{O}_{k-1} + \mathbf{B}_k\mathbf{u}_k \oplus \mathbf{D}_wW) \cap_{\mathbf{C}} (\mathbf{y}_k - \mathbf{E}_vV), \tag{5.17}$$

with $\mathcal{O}_0 \supset X_0 \cap_{\mathbf{C}} (\mathbf{y}_0 - \mathbf{E}_vV)$. Unlike the other, this formulation can be computed using the set representations described here. The three set operations considered critical are present in (5.16) and (5.17), the affine map represents the evolution of the set given the system dynamics. Following, there is a Minkowski sum with the disturbance set, and the intersection with the measurements set.

In the work of this thesis, the state set estimation does not consider measurements of the state $\mathbf{y}_k$, meaning the state estimation does not consider the intersection operation, only the affine map of the previous set according to the system dynamics and the Minkowski sum with a disturbance set. The expression (5.17) transforms into

$$\mathcal{O}_k \supset (\mathbf{A}_k\mathcal{O}_{k-1} + \mathbf{B}_k\mathbf{u}_k \oplus \mathbf{D}_wW), \tag{5.18}$$

with $\mathcal{O}_0 \supset X_0$. Given the use of zonotopes as the set representation, the exclusion of the intersection simplifies the computations since zonotopes are not closed under this set operation. For a nonlinear system, this problem is much more complex since reaching an expression for the system model and state estimation must consider nonlinear dynamics. This aspect further validates the model linearization considered in the solution proposed by this work.

5.2 Set Propagation Algorithm

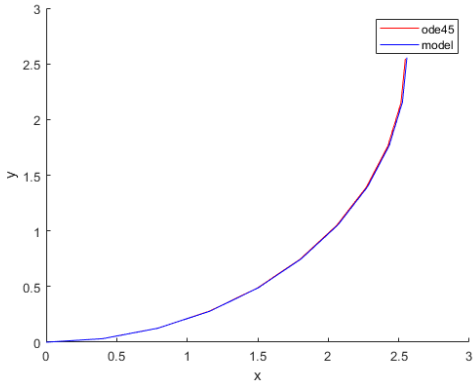
The set propagation algorithm is the first method described in this thesis to solve the considered state estimation problem. Firstly, the model choice is explained, following the algorithm description with explanatory figures and the respective mathematical formulation. As described in the previous section, the

main objective is to use CCGs to find a less conservative solution than we would find using other less complex set representations. The complexity added to the algorithm can bring more exact representations satisfying the minimal conservativeness goal. Based on the unicycle model dynamics in (4.1), this solution considers an Euler method discretization process and a second-order approximation Taylor expansion of the cosine and sine terms related to defining $\theta(t) = \theta_k + \omega_k t$. The following model is chosen to approximate the unicycle dynamics:

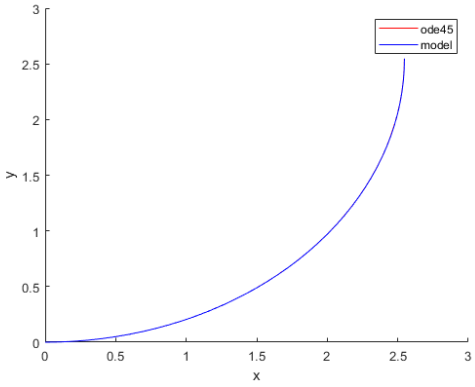
$$\mathbf{x}_{k+1} = \mathbf{A}_k \mathbf{x}_k + \mathbf{B}_k \mathbf{u}_k = \mathbf{x}_k + \begin{bmatrix} h\cos\theta_k & -\frac{h^2}{2} v_k \sin\theta_k \\ h\sin\theta_k & \frac{h^2}{2} v_k \cos\theta_k \\ 0 & h \end{bmatrix} \mathbf{u}_k, \tag{5.19}$$

where the inclusion of the variables θ_k and v_k in the matrix $\mathbf{B}$ leads to a model with nonlinear dynamics given by the presence of v_k. The model becomes more complex since matrix $\mathbf{B}$ is recalculated at each time step.

As intended by using this approximation, the model better represents the curvature of the trajectory leading to a very accurate approximation of the unicycle model. As a validation method of the model, Fig. 5.1 represents the model (5.19) and the unicycle model (derived by the function *ode45*), considering the same initial state $\mathbf{x} = [0, 0, 0]^\top$ and subject to the same input, $\mathbf{u} = \left[2, \frac{\pi}{4}\right]^\top$ during 2 seconds. It is important to note that the model considered in this section is updated at each time step.



(a) Representation of the model and *MATLAB's* unicycle model for $h = 0.2s$, the percentual error equals to 0.4115%.



(b) Representation of the model and *MATLAB's* unicycle model for $h = 0.05s$, the percentual error equals to 0.0257%.

Figure 5.1: Simulation results for $\mathbf{u} = \left[2, \frac{\pi}{4}\right]^\top$ considering different time step sizes h, and corresponding percentual errors.

These graphs reveal accurate results, where the approximated model deviates just 0.0257% from the original unicycle model in the two seconds of simulation time with $h = 0.05$ s. The simulations confirm the use of this model to reach the desired solution.

Having established the model choice, what follows is the explanation of the algorithm. As described in Chapter 4, the proposed solution provides the estimation of the state to which can be applied the constraints that guarantee safe trajectories. The algorithm will aim to provide a tight enclosure of the set to aid in the optimization of the generated trajectories.

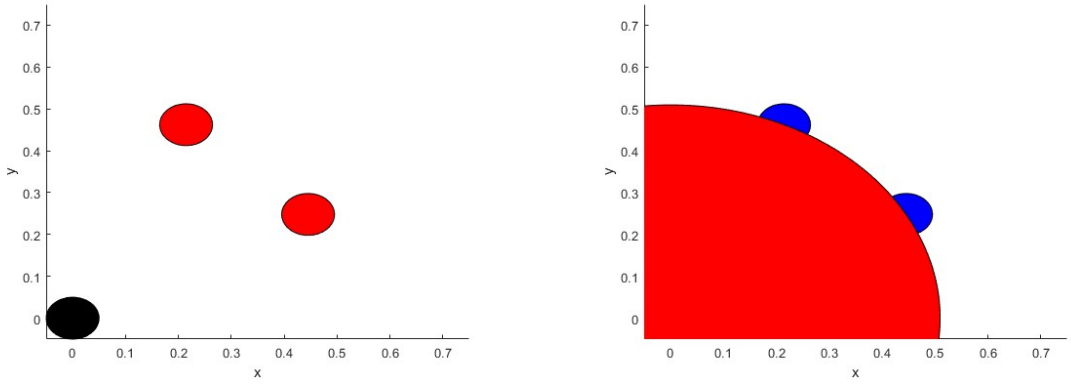
Consider the problem of a unicycle with an uncertain state, a set that represents the current possible positions and an uncertain orientation given by an interval, $[\theta_{min}, \theta_{max}]$. This orientation uncertainty will also be represented as $[\theta_n - \theta_d, \theta_n + \theta_d]$, where θ_n refers to the nominal value of θ and θ_d as the deviation value. Since most constraints would take effect around the position set given by the uncertain x and y, the goal is correctly predicting the position set evolution according to a certain input while

minimizing conservativeness. To reach these objectives, the algorithm is composed of the following steps:

- **Step 1:** compute the model and position set for the next time step, for both extreme values of θ. Where points p_1 and p_2 relate to the propagation with θ_{max} and θ_{min}, respectively;
- **Step 2:** record the coordinates for both points (p_1 and p_2) and compute the distance traveled r;
- **Step 3:** generate a set defined as a segment of a circle with radius r, cut by the line segment between the extreme values in points p_1 and p_2;
- **Step 4:** compute a Minkowski sum with the original set, and another Minkowski sum to account for the model error.

Since the distance travelled relies only on the inputs and not the initial orientation angle θ, the movement can be handled in two separate parts. Firstly, the distance travelled can be defined as a movement with amplitude r defining a circle of r radius around the initial state. Secondly, we consider the interval of initial orientation θ which results in slicing a segment of the circle. This restricts the uncertain position set to the range of positions between the corresponding extreme values of θ. The following Minkowski sums return the initial position uncertainty to the generated set and add uncertainty related to the model error. The model error can be estimated through several methods of over-approximation, in the following demonstration of the algorithm, the error is simply obtained by considering the position difference to the exact model (calculated with *ode45* MATLAB function).

The Fig. 5.2 and Fig. 5.3 illustrate the explained steps, for an example situation comprised of a position set of a circle with a 0.05 meters radius centred at the origin, and θ values of $\left[\frac{\pi}{5} - \frac{\pi}{10}, \frac{\pi}{5} + \frac{\pi}{10}\right]$. The inputs are $\mathbf{u} = [v, \omega]^\top = \left[1, \frac{\pi}{4}\right]^\top$, the time step considered is $h = 0.5$ seconds and the algorithm result is shown in Fig. 5.4.



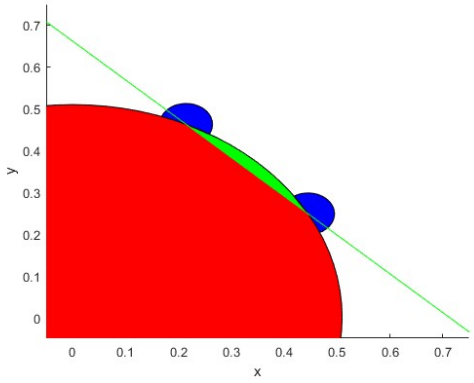
(a) Step 1: Original set (in black) and the set evolution for θ_{max} and θ_{min} (in red).

(b) Step 2: Circle of length r (red) and set evolution for θ_{max} and θ_{min} (blue).

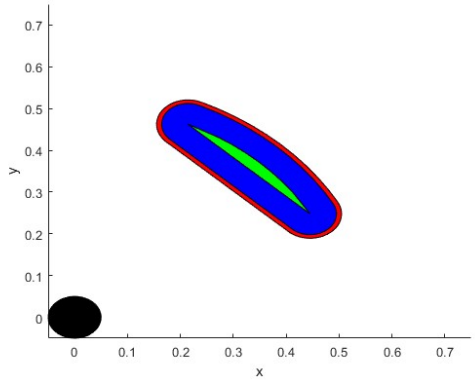
Figure 5.2: Representation of the first two steps of the set propagation algorithm

With these steps, the objective of defining a set that encloses every possible value for the uncertain state of the unicycle is successfully achieved. Considering the set represented in Fig. 5.4, the designed set tightly encloses the different possible positions, minimizing the conservativeness by over-approximating just enough to guarantee the convexity of the set.

Another situation to be considered is the similar case of uncertainty in the angular velocity ω. This particular situation can also be successfully tackled by separately executing the algorithm for multiple values of ω such as ω_{max}, ω_{min} and ω_n for example, and combining these sets to represent the actual



(a) Step 3: Green line which crosses both centers of the sets of extreme θ values, cuts the desired green circle segment.



(b) Step 4: In green, the previous circle segment; in blue, the result of the Minkowski sum of the green plus the original set (in black); and in red is the result of the Minkowski sum of the blue set with the computed error set.

Figure 5.3: Representation of the last two steps of the set propagation algorithm

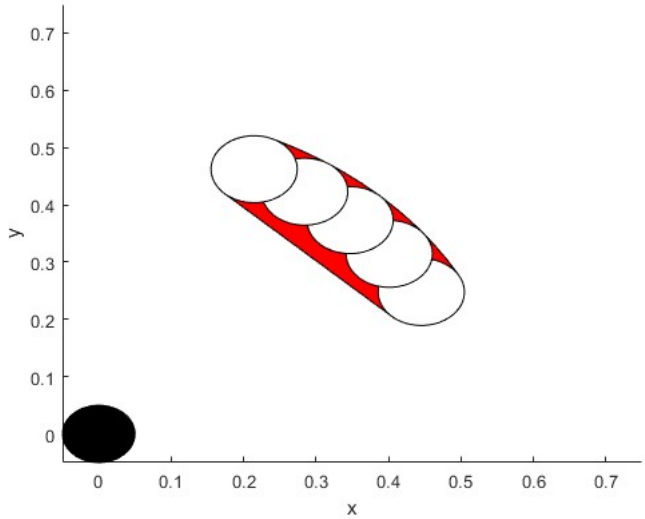


Figure 5.4: Result of the algorithm, showing the evolution from the initial set (black) to the set for the next time step (in red). The white sets, represent set propagation for different θ values in the considered uncertain values, subject to a Minkowski sum to account for the model error.

uncertain set given uncertainty in ω. Although this method is not fault-proof, it would require larger step sizes h and unusually large ω uncertainties to prove that it is not able to complete the objective. In that unlikely case, at some point, the sets of different ω values will be too far apart, and the convex hull of the sets does not include possible positions. This could be fixed, since more sets with more values for ω can be implemented.

A simulation was performed to test the evolution of the system under uncertain angular velocity, as shown in the following Fig. 5.5. As expected, the uncertainty of ω leads to a consistently expanding state set, leading away from the initial orientation into a funnel-like geometry.

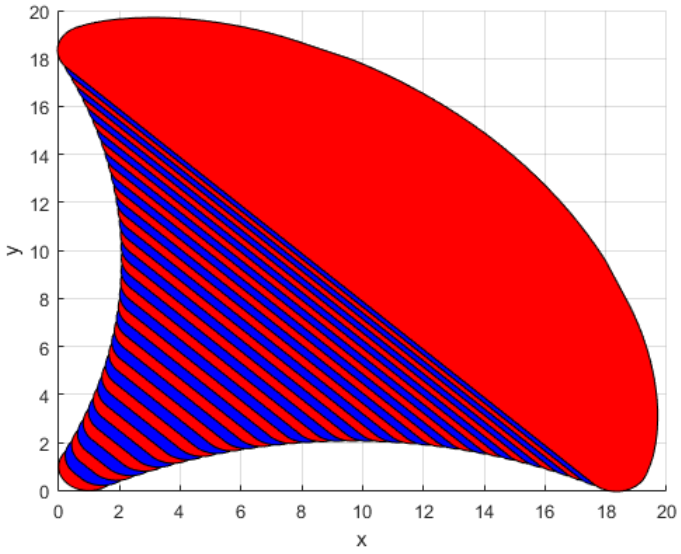


Figure 5.5: Result of the algorithm for uncertain ω, the different colours represent the corresponding set for the consecutive iterations of the set propagation. Considering an interval for $\omega \in \left[-\frac{\pi}{8}, \frac{\pi}{8}\right]$, and a smaller step size of $h = 0.05$ s.

The algorithm is able to consistently design tight enclosures for the uncertain set. The mathematical formulation related to the set generation for this solution will be presented in the next paragraphs. For the sake of simplicity, the indexes k of the state and input variables are omitted since the computations always refer to the current time step k.

The sets are represented by CCG, which is described in Subsection 5.1.1 as well as the corresponding set operations used in this algorithm. As detailed previously in step 3, the enclosure set is generated based on the distance travelled r and two interest points, p_1 and p_2, obtained by considering the extreme values of θ.

Firstly, it is required to obtain an expression for the distance traveled by the unicycle. According to (5.19), the distance is obtained by the first two rows of the term $\mathbf{Bu}$ that determine the variation of the position (x and y). The distance is therefore obtained by the norm of those values as,

$$r = \left\| \begin{bmatrix} hv\cos\theta - \frac{h^2}{2}v\omega\sin\theta \\ hv\sin\theta + \frac{h^2}{2}v\omega\cos\theta \end{bmatrix} \right\| = hv\sqrt{1+\omega^2\frac{h^2}{4}}, \tag{5.20}$$

where an important note is that the result does not rely on the angle θ as expected since the angle will define the orientation of the movement and not the distance.

Considering S a CCG that represents a set of possible positions in the 2-D state space, as the tuple $S = (\mathbf{G}_S, \mathbf{c}_S, \mathbf{A}_S, \mathbf{b}_S, \mathfrak{C}_S) \subset \mathbb{R}^2$. Step 3 references the generation of S which initially considers $\mathbf{G}_S$ as 2x2 identity matrix multiplied by the value r, centre $\mathbf{c}_S$ as $\mathbf{0}_{2\times1}$ and unit ℓ_2 norm ball in $\mathfrak{C}_S$ for the

two generators. This implementation leads to the circle of radius r around the origin, to represent the movement around the initial state. The next step is to define $\mathbf{A}_{\mathcal{S}}$ and $\mathbf{b}_{\mathcal{S}}$ to correctly restrain the set to the desired circle segment. To accurately define the constraint, the computation relies on the geometric properties of the problem and the definition of some auxiliary variables to describe the constraint. Firstly, we formalize the expressions to find p_1, p_2 and also another point of interest p_n for $\theta = \theta_n$, representing the expected movement without uncertainty d. The points can be expressed as follows,

$$\begin{aligned} p_1 &= (x_{P1}, y_{P1}) = (hv(\cos(\theta + d) - \frac{h}{2}\omega \sin(\theta + d)) \ , \ hv(\sin(\theta + d) + \frac{h}{2}\omega \cos(\theta + d))) \\ p_2 &= (x_{P2}, y_{P2}) = (hv(\cos(\theta - d) - \frac{h}{2}\omega \sin(\theta - d)) \ , \ hv(\sin(\theta - d) + \frac{h}{2}\omega \cos(\theta - d))) \\ p_n &= (x_n, y_n) = (hv(\cos\theta - \frac{h}{2}\omega \sin\theta) \ , \ hv(\sin\theta + \frac{h}{2}\omega \cos\theta)) \end{aligned} \tag{5.21}$$

describing the position difference from the initial state, which is the reason why the CCG's centre is placed at the origin in this step. These points are shown in Fig. 5.6, and help define the constraint of the CCG.

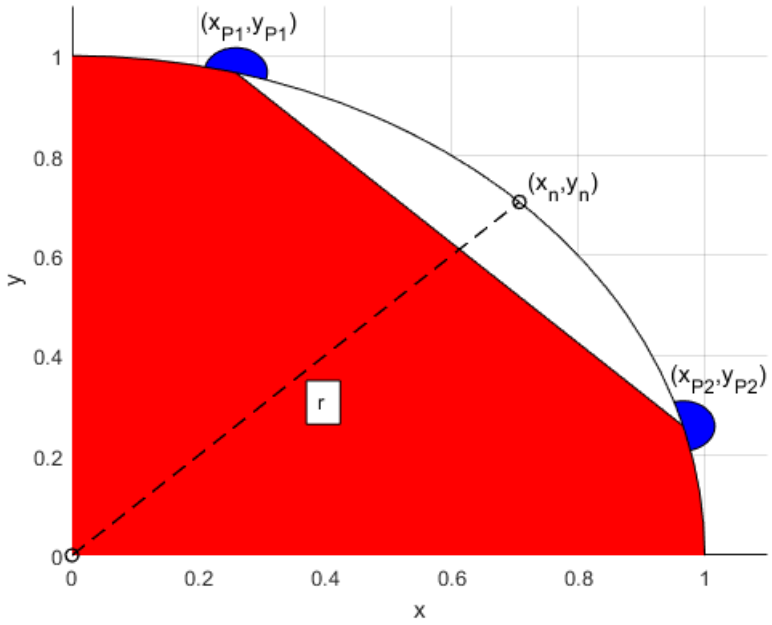


Figure 5.6: Representation of the points of interest, where points $p1$ and $p2$ represent the centre of the respective blue sets, and the white segment represents the desired circle segment defined by $\mathbf{A}_{\mathcal{S}}$ and $\mathbf{b}_{\mathcal{S}}$.

As depicted in the figure, the line between points p_1 and p_2 marks the limit of the wanted circle segment. As such, the constraint should only accept the set points present between the line and p_n. The key idea, then, is to represent the line which crosses p_1 and p_2 and the parallel line crossing p_n and define an inequality constraint to consider only the points between the lines that belong to the set. Since the set representation only allows linear constraints, we introduce a slack variable to convert the inequality to equality and implement the constraint.

Considering a cartesian representation of the lines with y as a function of x, the slack variable's value will range from 0 to the difference between the lines for a given x. Associating the slack variable to the interval $[0, s]$, the lines can be expressed as

$$\begin{aligned} y_1 &= ax + b \\ y_2 &= ax + b + s \end{aligned} \tag{5.22}$$

which leads to the representation in Fig. 5.7.

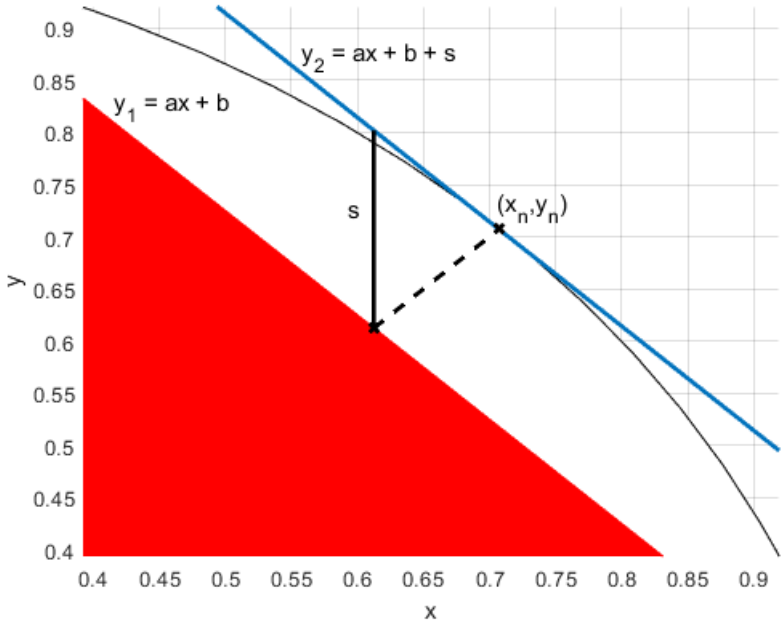


Figure 5.7: Representation of the auxiliary variables to define the inequality constraint, s represent the maximum of the slack variable value

Analyzing the figure, the parameters of both lines are trivially reached by noticing the lines are perpendicular to the movement of p_n and thus the slope a can be defined as

$$a = -\frac{y_n}{x_n} = \frac{\frac{h}{2}\omega \sin(\theta) - \cos(\theta)}{\sin(\theta) + \frac{h}{2}\omega \cos(\theta)} \tag{5.23}$$

leading to b when considering the point $p1$ or $p2$

$$b = y_{P1} - ax_{P1} = y_{P2} - ax_{P2} \tag{5.24}$$

which allows the computation of s. Since p_n belongs to y_2, s can be described as

$$s = y_n - ax_n - b \tag{5.25}$$

concluding the description of the variables required to define $\mathbf{A}_{\mathcal{S}}$ and $\mathbf{b}_{\mathcal{S}}$ and successfully implement the constraint of the CCG. The constraint relies on a slack variable c and is defined as

$$y = ax + b + c, \qquad 0 \leq c \leq s \tag{5.26}$$

where x and y belong to the set.

Considering the CCG formulation in (5.11), $\mathbf{A}\xi = \mathbf{b}$, we can define x and y to be associated with the two previous generators of $\mathbf{G}_{\mathcal{S}}$ and add another generator related to the slack variable, of unit ℓ_∞ norm ball. In this way, c represents a range of values that can be specified to fit the desired interval. Modifying the representation from x, y and c to auxiliary variables ξ_x, ξ_y and ξ_c respectively, the matrices $\mathbf{A}_{\mathcal{S}}$ and

$\mathbf{b}_\mathcal{S}$ can be defined based on the following equality

$$\begin{aligned} r\xi_y &= a(r\xi_x) + b + \xi_c, && -1 \leq \xi_c \leq 1 \\ \Leftrightarrow -a(r\xi_x) + r\xi_y - \frac{s}{2}\xi_c &= b + \frac{s}{2}, && 0 \leq \frac{s}{2}(\xi_c + 1) \leq s \end{aligned} \tag{5.27}$$

where the coefficients of the auxiliary variables characterize $\mathbf{A}_\mathcal{S}$ and the right side of the equation defines $\mathbf{b}_\mathcal{S}$. The coefficient r refers to the operation $\mathbf{G}_\mathcal{S}\xi + \mathbf{c}_\mathcal{S}$ that describes the set. The intervals on the right show the transformation of the slack variable to accurately represent the desired interval $[0, s]$.

The generated set S can then be updated with $\mathbf{A}_\mathcal{S}$ and $\mathbf{b}_\mathcal{S}$ that impose the constraint. This constraint represents an increase of dimension of ξ to three, and so the matrix $\mathbf{G}_\mathcal{S}$ should also reflect this change. The set is represented by the tuple $S = (\mathbf{G}_\mathcal{S}, \mathbf{c}_\mathcal{S}, \mathbf{A}_\mathcal{S}, \mathbf{b}_\mathcal{S}, \mathfrak{C}_\mathcal{S}) \subset \mathbb{R}^2$, which can be explicitly described as

$$S = \left(\begin{bmatrix} r & 0 & 0 \\ 0 & r & 0 \end{bmatrix} \ , \ \mathbf{0}_{2\times 1} \ , \ \left[-ar, r, -\frac{s}{2}\right] \ , \ b + \frac{s}{2} \ , \ \mathfrak{C}_\mathcal{S} \right) \subset \mathbb{R}^2 \tag{5.28}$$

where a column of zeroes is added to $\mathbf{G}_\mathcal{S}$ to have the correct dimension and the convex sets of the generators, $\mathfrak{C}_\mathcal{S}$ are comprised by two unit ℓ_2 norm balls and a unit ℓ_∞ norm ball.

As explained in the algorithm steps, the final computation consists of two consecutive Minkowski sums of CCGs. Defining the final solution set as $T = (\mathbf{G}_\mathcal{T}, \mathbf{c}_\mathcal{T}, \mathbf{A}_\mathcal{T}, \mathbf{b}_\mathcal{T}, \mathfrak{C}_\mathcal{T}) \subset \mathbb{R}^2$, the initial state set as $Z = (\mathbf{G}_\mathcal{Z}, \mathbf{c}_\mathcal{Z}, \mathbf{A}_\mathcal{Z}, \mathbf{b}_\mathcal{Z}, \mathfrak{C}_\mathcal{Z}) \subset \mathbb{R}^2$ and another CCG, $L = (\mathbf{G}_\mathcal{L}, \mathbf{c}_\mathcal{L}, \mathbf{A}_\mathcal{L}, \mathbf{b}_\mathcal{L}, \mathfrak{C}_\mathcal{L}) \subset \mathbb{R}^2$, that accounts for the errors of the chosen unicycle model, the final equation to reach the set is

$$T = S \oplus Z \oplus L \tag{5.29}$$

where only the formulation of Z is unknown, and L can be characterized as

$$L = \left(\begin{bmatrix} e_x & 0 \\ 0 & e_y \end{bmatrix} \ , \ \mathbf{0}_{2\times 1} \ , \ \mathbf{0}_{1\times 2} \ , \ 0 \ , \ \mathfrak{C}_\mathcal{L} \right) \subset \mathbb{R}^2 \tag{5.30}$$

with e_x and e_y refering to the computed error in the x and y direction, respectively, and $\mathbf{A}_\mathcal{L}$ and $\mathbf{b}_\mathcal{L}$ as blank matrices since no constraint is required. Both these e_x and e_y parameters and the convex sets $\mathfrak{C}_\mathcal{L}$ are tunable and vary when considering different methods of approximating the model error. For the demonstrations, this error was computed as the position difference to the exact unicycle model using the $ode45$ function for the nominal θ.

The result of the set propagation algorithm is then the CCG T, which accurately represents the propagation of the initial state set Z for a given input $\mathbf{u}$. The CCG has this general formulation of Minkowski sums between three CCGs,

$$T = \left(\begin{bmatrix} \mathbf{G}_\mathcal{S} & \mathbf{G}_\mathcal{Z} & \mathbf{G}_\mathcal{L} \end{bmatrix} \ , \ \mathbf{c}_\mathcal{S} + \mathbf{c}_\mathcal{Z} + \mathbf{c}_\mathcal{L} \ , \ \mathrm{diag}\left(\mathbf{A}_\mathcal{S} \ , \ \mathbf{A}_\mathcal{Z} \ , \ \mathbf{A}_\mathcal{L}\right) \ , \ \begin{bmatrix} \mathbf{b}_\mathcal{S} \\ \mathbf{b}_\mathcal{Z} \\ \mathbf{b}_\mathcal{L} \end{bmatrix} \ , \ \{\mathfrak{C}_\mathcal{S}, \mathfrak{C}_\mathcal{Z}, \mathfrak{C}_\mathcal{L}\} \right) \subset \mathbb{R}^2 \tag{5.31}$$

where $\mathbf{A}_\mathcal{T}$ is represented by the diagonal blocks of the other CCGs matrices.

By switching the specific generator matrices as well as the constraints, for their expressions, the final CCG formulation is found,

$$T = \left(\begin{bmatrix} r & 0 & 0 & \mathbf{G}_\mathcal{Z}[1] & e_x & 0 \\ 0 & r & 0 & \mathbf{G}_\mathcal{Z}[2] & 0 & e_y \end{bmatrix}, \mathbf{c}_\mathcal{Z}, \begin{bmatrix} -ar & r & -\frac{s}{2} & \mathbf{0} & \mathbf{0}_{1\times 2} \\ 0 & 0 & 0 & \mathbf{A}_\mathcal{Z} & \mathbf{0}_{1\times 2} \end{bmatrix}, \begin{bmatrix} b + \frac{s}{2} \\ \mathbf{b}_\mathcal{Z} \end{bmatrix}, \{\mathfrak{C}_\mathcal{S}, \mathfrak{C}_\mathcal{Z}, \mathfrak{C}_\mathcal{L}\} \right) \subset \mathbb{R}^2 \tag{5.32}$$

where $\mathbf{G}_\mathcal{Z}[i]$ represents the i-th line of the matrix $\mathbf{G}_\mathcal{Z}$ and the last line of $\mathbf{A}_\mathcal{T}$ and $\mathbf{b}_\mathcal{T}$ is omitted as it

would be represented by zeroes since the CCG L does not define a constraint.

The set propagation algorithm is finalized, successfully representing the evolution of the set for a given input. It provides a tight enclosure of the uncertain position set with minimal conservativeness, as was desired. However, this formulation poses some challenges, given it represents a high-complexity approach to the problem.

To define the desired MPC constraints one initial set must be consecutively propagated N times, which corresponds to the MPC horizon. The state $\mathbf{x}_k$ and input $\mathbf{u}_k$ are present in the set propagation computations, and the resulting set is defined in (5.32). As a note, the set in (5.32) shows only the variation of the position uncertainty set, since θ_k evolves from the interval $[\theta_{min}, \theta_{max}]$ to $[\omega_k\theta_{min}, \omega_k\theta_{max}]$. The propagation of a set in the MPC is then dependent on both $\mathbf{x}_k$ and $\mathbf{u}_k$, which are optimization variables, and the previous CCG set which is also expressed by optimization variables. This leads to complicated nonlinearities that the solvers used with the MPC problem struggle to solve. Furthermore, to accurately define constraints we must obtain the limits of the CCG, with the representation in (5.11). This operation is based on YALMIP, a MATLAB toolbox, which cannot perform the representation since the CCG is composed of complex relations between optimization variables.

The nonlinearities in the model contribute to this problem since the set propagation algorithm uses the model to define the elements of the CCG that represents the state in the following time step. Although it is not suitable for this problem, this algorithm proves to be very valuable in assessing the progression of a unicycle uncertain set for a given input. Also, the interpretation of separating the movement into the travelled distance and orientation of the movement can be relevant in unicycle model state estimation computations.

5.3 Model Linearization

This section provides a detailed description of the linearization and discretization processes considered to compute the model that can be used in the MPC formulation. The first approach employs a nonlinear model which is not ideal to solve the problem, given its high complexity. The model presented here will be used for the proposed solution algorithm. With this model, it is possible to develop a control system that can predict the state-set evolution and, therefore, accomplish the objectives of this thesis.

The unicycle vehicle model used considers the following function $f(\mathbf{x})$ which depicts the evolution of the system by providing an equation for the derivative of each component of $\mathbf{x} = [x\ y\ \theta]^\top$. The function, also represented in (4.1), is presented here in more detail as,

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = f(\mathbf{x}, \mathbf{u}) \iff \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v \cdot \cos\theta \\ v \cdot \sin\theta \\ \omega \end{bmatrix} \iff \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}, \tag{5.33}$$

where x and y denote the position of the vehicle and θ, the corresponding vehicle heading. The input $\mathbf{u}$ is a vector composed of v, the vehicle speed, and ω, the vehicle heading angular velocity. The system is characterized by this nonlinear function, represented in continuous time.

Using Taylor's first-order polynomial results in a linear approximation of the system, which is adequately discretized next. Firstly we consider the following linear approximation,

$$f(\mathbf{x}, \mathbf{u}) \approx P_1(\mathbf{x}, \mathbf{u}), \tag{5.34}$$

at a determined state $\mathbf{x}^*$ and input $\mathbf{u}^*$ which define the linearization point. The first-order Taylor series is represented by P_1. The state derivative $\dot{\mathbf{x}}$ can then be approximated as such

$$\dot{\mathbf{x}} \approx P_1(\mathbf{x}, \mathbf{u}) = f(\mathbf{x}^*, \mathbf{u}^*) + \frac{\partial f}{\partial(\mathbf{x})_i}(\mathbf{x}^*, \mathbf{u}^*)(\mathbf{x} - \mathbf{x}^*) + \frac{\partial f}{\partial(\mathbf{u})_j}(\mathbf{x}^*, \mathbf{u}^*)(\mathbf{u} - \mathbf{u}^*), \tag{5.35}$$

where the gradient of f, ∇f, is given by the partial derivatives of f in regards to $\mathbf{x}$ and $\mathbf{u}$, which can be represented as

$$\frac{\partial f}{\partial(\mathbf{x})_i}(\mathbf{x}^*, \mathbf{u}^*)(\mathbf{x}-\mathbf{x}^*)+\frac{\partial f}{\partial(\mathbf{u})_j}(\mathbf{x}^*, \mathbf{u}^*)(\mathbf{u}-\mathbf{u}^*) = \begin{bmatrix} 0 & 0 & -v^* \cdot \sin\theta^* \\ 0 & 0 & v^* \cdot \cos\theta^* \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \delta x \\ \delta y \\ \delta\theta \end{bmatrix} + \begin{bmatrix} \cos\theta^* & 0 \\ \sin\theta^* & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \delta v \\ \delta\omega \end{bmatrix}, \tag{5.36}$$

with $\delta\mathbf{x} = \mathbf{x} - \mathbf{x}^*$ and $\delta\mathbf{u} = \mathbf{u} - \mathbf{u}^*$. Combining with (5.35), and considering the following relation $\dot{\mathbf{x}} = \dot{\mathbf{x}}^* + \delta\dot{\mathbf{x}}$. The expression develops as

$$\dot{\mathbf{x}}^* + \delta\dot{\mathbf{x}} \approx \begin{bmatrix} v^* \cdot \cos\theta^* \\ v^* \cdot \sin\theta^* \\ \omega^* \end{bmatrix} + \begin{bmatrix} 0 & 0 & -v^* \cdot \sin\theta^* \\ 0 & 0 & v^* \cdot \cos\theta^* \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \delta x \\ \delta y \\ \delta\theta \end{bmatrix} + \begin{bmatrix} \cos\theta^* & 0 \\ \sin\theta^* & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \delta v \\ \delta\omega \end{bmatrix}, \tag{5.37}$$

where the first term of the right hand side of the equation, corresponding to $f(\mathbf{x}^*, \mathbf{u}^*)$, cancels out the term $\dot{\mathbf{x}}^*$, as both represent the derivative of the state around the linearization point.

The final linearized model describes the derivative of $\delta\mathbf{x}$ as,

$$\delta\dot{\mathbf{x}} \approx \mathbf{A}(\mathbf{x}^*, \mathbf{u}^*) \begin{bmatrix} \delta x \\ \delta y \\ \delta\theta \end{bmatrix} + \mathbf{B}(\mathbf{x}^*) \begin{bmatrix} \delta v \\ \delta\omega \end{bmatrix} = \begin{bmatrix} 0 & 0 & -v^* \cdot \sin\theta^* \\ 0 & 0 & v^* \cdot \cos\theta^* \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \delta x \\ \delta y \\ \delta\theta \end{bmatrix} + \begin{bmatrix} \cos\theta^* & 0 \\ \sin\theta^* & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \delta v \\ \delta\omega \end{bmatrix}, \tag{5.38}$$

with this model represented as a function of $\delta\mathbf{x}$ and $\delta\mathbf{u}$. The dynamics are linearized around a certain state $\mathbf{x}^*$ and input $\mathbf{u}^*$, which introduces some degree of simplification error due to the difference of $\delta\mathbf{x}$ and $\delta\mathbf{u}$ and the linearization point.

The next required step is the discretization of the linearized model. The equation (5.38), represents the continuous-time linearized system. It is possible to discretize such a system, considering zero-order hold for the input $\mathbf{u}$ and computing the discretized matrices $\mathbf{A}_k$ and $\mathbf{B}_k$ from the continuous-time system matrices $\mathbf{A}(\mathbf{x}^*, \mathbf{u}^*)$ and $\mathbf{B}(\mathbf{x}^*)$. Firstly this model can be represented by the system,

$$\begin{bmatrix} \delta\dot{\mathbf{x}} \\ \delta\dot{\mathbf{u}} \end{bmatrix} = \begin{bmatrix} \mathbf{A}(\mathbf{x}^*, \mathbf{u}^*) & \mathbf{B}(\mathbf{x}^*) \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \delta\mathbf{x} \\ \delta\mathbf{u} \end{bmatrix} \tag{5.39}$$

which represents an alternate configuration of (5.38).

Since the following homogeneous ODE,

$$\dot{\mathbf{z}} = \mathbf{M}\mathbf{z}, \quad \mathbf{z}(t_0) = \mathbf{z}_0 \tag{5.40}$$

has a known solution,

$$\mathbf{z}(t) = e^{\mathbf{M}(t-t_0)}\mathbf{z}_0 \tag{5.41}$$

which can be transformed into discrete-time, by considering h as the time step size given by $t - t_0$, where $\mathbf{z}(t)$ and $\mathbf{z}_0$ correspond to $\mathbf{z}_{k+1}$ and $\mathbf{z}_k$, respectively.

The solution would then be,

$$\mathbf{z}_{k+1} = e^{\mathbf{M}h}\mathbf{z}_k \tag{5.42}$$

where, given the problem context, $\mathbf{z}$ can be defined as the vector combination of the state and input variables, as $\mathbf{z}^\top = \left[\delta\mathbf{x}^\top, \delta\mathbf{u}^\top\right]$.

Considering $\mathbf{M}$ as the matrix in (5.39), leads to this formulation,

$$\mathbf{M} = \begin{bmatrix} 0 & 0 & -v^* \cdot \sin\theta^* & \cos\theta^* & 0 \\ 0 & 0 & v^* \cdot \cos\theta^* & \sin\theta^* & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{5.43}$$

and to the discretized system model below,

$$\begin{bmatrix} \delta\mathbf{x}_{k+1} \\ \delta\mathbf{u}_{k+1} \end{bmatrix} = e^{\mathbf{M}h} \begin{bmatrix} \delta\mathbf{x}_k \\ \delta\mathbf{u}_k \end{bmatrix} \iff \begin{bmatrix} \delta\mathbf{x}_{k+1} \\ \delta\mathbf{u}_{k+1} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_k & \mathbf{B}_k \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \delta\mathbf{x}_k \\ \delta\mathbf{u}_k \end{bmatrix}, \tag{5.44}$$

where $\mathbf{I}$ is the identity matrix with the appropriate dimensions, and the exponential of the matrix $\mathbf{M}$ must be computed by combining (5.44) and (5.43). This is done through the following equation,

$$e^{\mathbf{M}h} = \mathcal{L}^{-1}\{(s\mathbf{I} - \mathbf{M})^{-1}\}_{t=h}, \tag{5.45}$$

which leads to the discretized model matrices $\mathbf{A}_k$ and $\mathbf{B}_k$:

$$e^{\mathbf{M}h} = \begin{bmatrix} \mathbf{A}_k & \mathbf{B}_k \\ \mathbf{0} & \mathbf{I} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -hv^* \cdot \sin\theta^* & h\cos\theta^* & -\frac{h^2}{2}v^*\sin\theta^* \\ 0 & 1 & hv^* \cdot \cos\theta^* & h\sin\theta^* & \frac{h^2}{2}v^*\cos\theta^* \\ 0 & 0 & 1 & 0 & h \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5.46}$$

where the matrices $\mathbf{A}_k$ and $\mathbf{B}_k$, define an expression for $\delta\mathbf{x}_{k+1}$ based on the previous state $\delta\mathbf{x}_k$ and a given input $\delta\mathbf{u}_k$.

The final linearized and discretized model $F(\delta\mathbf{x}_k, \delta\mathbf{u}_k, \mathbf{x}^*, \mathbf{u}^*)$ is complete and describes the evolution of the state variation from the operating point. This variation $\delta\mathbf{x}_{k+1}$ is defined as

$$\begin{aligned} \delta\mathbf{x}_{k+1} &= F(\delta\mathbf{x}_k, \delta\mathbf{u}_k, \mathbf{x}^*, \mathbf{u}^*) = \mathbf{A}_k \begin{bmatrix} \delta x_k \\ \delta y_k \\ \delta\theta_k \end{bmatrix} + \mathbf{B}_k \begin{bmatrix} \delta v_k \\ \delta\omega_k \end{bmatrix} \\ \delta\mathbf{x}_{k+1} &= \begin{bmatrix} 1 & 0 & -hv^* \cdot \sin\theta^* \\ 0 & 1 & hv^* \cdot \cos\theta^* \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \delta x_k \\ \delta y_k \\ \delta\theta_k \end{bmatrix} + \begin{bmatrix} h\cos\theta^* & -\frac{h^2}{2}v^*\sin\theta^* \\ h\sin\theta^* & \frac{h^2}{2}v^*\cos\theta^* \\ 0 & h \end{bmatrix} \begin{bmatrix} \delta v_k \\ \delta\omega_k \end{bmatrix}, \end{aligned} \tag{5.47}$$

which can also serve as a model to compute the next state $\mathbf{x}_{k+1}$ by considering the important definition of $\delta\mathbf{x}$, as $\delta\mathbf{x}_{k+1} = \mathbf{x}_{k+1} - \mathbf{x}^*$.

After the steps of linearization and discretization the nonlinear continuous time unicycle vehicle model in (5.33) leads to the system described in (5.47). The matrices $\mathbf{A}_k$ and $\mathbf{B}_k$, which characterize the system, are a function of the operating point values $(\mathbf{x}^*, \mathbf{u}^*)$ and the time step size h. The system can then be defined as a Linear time-invariant (LTI) system since the nonlinearities in the original function were eliminated by the linear approximation, and the matrices maintain their value regardless of the particular time step. Considering a LTI system as the object of control greatly simplifies the process, as

nonlinear or even LTV systems add considerable complexity to the control problem, as noticed in the previous section, where the nonlinear model leads to an overly complex MPC implementation. However, since the model best describes small variations around a certain operating point $(\mathbf{x}^*, \mathbf{u}^*)$, the accuracy of the model decreases as the state evolves away from this point. Addressing the model error introduced by the linear approximation is a key focus of this work, as the trajectory safety problem must also account for model inaccuracies and uncertainties.

5.4 Solution Algorithm

In this section, we present the design of the MPC controller aimed at ensuring safe trajectory generation in the presence of uncertainties. The primary objective is to generate trajectories that maintain safety despite uncertainties in the system state and the presence of obstacles in the environment.

Firstly we provide an overview of the MPC control strategy. The Algorithm will generate estimates of future states in the MPC horizon and compute the discrete model error for each of the time steps. The model error estimation is shown on the next subsection. It will follow the construction and solving of the MPC problem, to obtain input $\mathbf{u}_k$.

Similarly to other MPC strategies, the input $\mathbf{u}_k$ returned by the optimization problem serves to estimate the propagation of the uncertain state set X_k. Afterwards, the new uncertainty state set X_{k+1} is defined by the intersection with the measurement set. The state $\mathbf{x}_{k+1}$ is defined as the centre of X_{k+1}, and both variables are used for the next iteration of the algorithm until the end of the simulation when $k = N_T$.

The MPC controller relies on a model that approximates system dynamics, introduced in the previous section. Throughout this solution, the state $\mathbf{x}_k$ is expressed as $\mathbf{x}_k = \delta\mathbf{x}_k + \mathbf{x}^*$ which introduces error. To address the challenges posed by this model approximation, the solution will account for the model error through over-approximation techniques computed in the model error estimation Algorithm. It will estimate the model uncertainty in a zonotope representation for a worst-case scenario for each time step, from k to $k + N$, with N representing the horizon of the MPC problem. This process is explained in the next subsection.

After estimating the model error, the following subsection will explain the integration of the over-approximations in the MPC problem and the overall construction of the MPC control. The MPC constraints to ensure safe operation are defined and explained. These constraints include state constraints, which define the system's evolution, input constraints, and obstacle avoidance constraints. All of these are represented in the problem formulation (4.5).

Finally, the rest of the algorithm is described, explaining the steps that use the output of the optimization problem, $\mathbf{u}_k$, which lead to the estimation of the next state, $\mathbf{x}_{k+1}$.

Before describing the algorithm, it is important to clarify the variables considered to define the MPC problem and the variables that define the actual state.

The MPC optimization problem is already defined in (4.5) with $k + j$ defining the time steps from k to the horizon $k + N$ to use in the constraint formulation. In each time step j the problem will estimate the uncertain state set X_{k+j} that defines the uncertainty around state $\mathbf{x}_{k+j}$, using this to formulate the problem and solve for $\mathbf{u}_k$. Both X_{k+j} and $\mathbf{x}_{k+j}$ refer to possible states and not the actual state defined by $\mathbf{x}_k$ and respective uncertainty set X_k. It should also be noted that the used model is a function of the variation $\delta\mathbf{x}_k = \mathbf{x}_k + \mathbf{x}^*$ and $\delta\mathbf{u}_k = \mathbf{u}_k + \mathbf{u}^*$ and not the state $\mathbf{x}_k$ or the input $\mathbf{u}_k$. As such, the following consideration should be made that the state set X_k leads to the state variation set T_k by subtracting $\mathbf{x}^*$ to the centre of X_k, $\mathbf{c}_{\mathcal{X}_k} = \mathbf{x}_k$. It leads to $\mathbf{c}_{\mathcal{T}_k} = \delta\mathbf{x}_k$, as X_k and T_k differ only in the centre component of the zonotope. Given this relation, the MPC problem computations are done according to the state

variation $\delta\mathbf{x}_{k+j}$ and respective set T_{k+j}, however, since the cost function and position reference relate to the system's state $\mathbf{x}_{k+j}$ the final constraints are represented as a function of $\mathbf{x}_{k+j}$ and X_{k+j}.

5.4.1 Estimation of model error

In this approach, the model error is over-approximated by the Lagrange remainder, which is represented as a zonotope, along with the state sets under consideration. The model is an approximation based on a first-order Taylor series, with the Lagrange remainder capturing an over-approximation of the infinite higher-order terms of the Taylor series, as detailed in [45]. Consider the vector $\mathbf{z}$ as the combination of $\mathbf{x}$ and $\mathbf{u}$, which leads to $\mathbf{z} = [x, y, \theta, v, \omega]^\top$ and represents the variables of interest to define the next state. This vector can be used to describe an approximation of the infinite Taylor series as

$$\dot{\mathbf{x}} \in f(\mathbf{z}^*) + \nabla_{\mathbf{z}} f(\mathbf{z})|_{\mathbf{z}=\mathbf{z}^*}(\mathbf{z}-\mathbf{z}^*) + \frac{1}{2}(\mathbf{z}-\mathbf{z}^*)^\top \nabla^2_{\mathbf{z}} f(\mathbf{z})|_\zeta (\mathbf{z}-\mathbf{z}^*), \tag{5.48}$$

where $\zeta \in \{\mathbf{z}^* + \alpha(\mathbf{z}-\mathbf{z}^*) \mid \alpha \in [0,1]\}$ and $\mathbf{z}^*$ denotes the linearization point. The last term of the expression refers to the Lagrange remainder, while the rest refers to the first-order Taylor approximation, approached in Section 5.3.

The expression for the Lagrange remainder can be written as,

$$(\mathbf{L})_i = \frac{1}{2}(\mathbf{z}-\mathbf{z}^*)^\top \mathbf{J}_i(\zeta)(\mathbf{z}-\mathbf{z}^*), \tag{5.49}$$

where i denotes the i^{th} component of $\mathbf{x}$ that the remainder $(\mathbf{L})_i$ and $\mathbf{J}_i(\zeta)$ refer to. The matrix $\mathbf{J}_i(\zeta)$, expressed as

$$\mathbf{J}_i(\zeta) = \nabla^2_{\mathbf{z}} f_i(\mathbf{z})|_\zeta, \tag{5.50}$$

relates to the Hessian of $f_i(\zeta)$, with i again referring to the components of $\mathbf{x}$.

To determine the exact values of the Lagrange remainder, we would need to know the values of $\mathbf{z}$ for the given time step, and since $\mathbf{z}$ represents the variables of the MPC, the values are unknown during the calculations. We can obtain an over-approximation for the absolute values of the Lagrange remainder by considering $\mathbf{z} \in Z$ where Z is a zonotope, with a determined centre $\mathbf{c}$ and generators $\mathbf{g}^{(j)}$, $Z = (\mathbf{c}, \mathbf{g}^{(1)}, ..., \mathbf{g}^{(j)})$ such that it represents the interval of possible values for every component of $\mathbf{z}$. To implement this method as an estimate for the linearization error, we need to compute the maximum value of $|\mathbf{J}_i(\zeta)|$. Since the maximum remainder absolute values are reached through this expression, developed in [45],

$$(\mathbf{l})_i = \frac{1}{2}\gamma^\top \max(|\mathbf{J}_i(\zeta)|)\gamma, \tag{5.51}$$

where the max operator and absolute operator are considered as element-wise and γ is represented as,

$$\gamma = |\mathbf{c} - \mathbf{z}^*| + \sum_m^j |g^{(m)}|, \tag{5.52}$$

with $\mathbf{z}^* = [\mathbf{x}^{*\top}, \mathbf{u}^{*\top}]^\top$. The computed value $(\mathbf{l})_i$ represents an approximation of the absolute values for the remainder, in

$$|(\mathbf{L})_i| \subseteq [0, (\mathbf{l})_i], \tag{5.53}$$

adequately defines the range of the absolute values of the Lagrange remainder. This over-approximation relies on assuming the worst-case scenario related to the parameters' variation, considering every possible value of $\mathbf{z} \in Z$ for the γ computations. Considering only the maximum absolute value of each

element in $\mathbf{J}_i(\zeta)$ can be another source of conservativeness in the solution. However, overapproximating the error allows the definition of the safety constraints which is the ultimate objective of these processes.

Based on the unicycle model (5.33), the general expression for $\mathbf{J}_i(\zeta)$ is

$$\mathbf{J}_i(\zeta) = \begin{bmatrix} \frac{\partial f_i}{\partial x^2} & \frac{\partial f_i}{\partial x \partial y} & \frac{\partial f_i}{\partial x \partial \theta} & \frac{\partial f_i}{\partial x \partial v} & \frac{\partial f_i}{\partial x \partial \omega} \\ \frac{\partial f_i}{\partial y \partial x} & \frac{\partial f_i}{\partial y^2} & \frac{\partial f_i}{\partial y \partial \theta} & \frac{\partial f_i}{\partial y \partial v} & \frac{\partial f_i}{\partial y \partial \omega} \\ \frac{\partial f_i}{\partial \theta \partial x} & \frac{\partial f_i}{\partial \theta \partial y} & \frac{\partial f_i}{\partial \theta^2} & \frac{\partial f_i}{\partial \theta \partial v} & \frac{\partial f_i}{\partial \theta \partial \omega} \\ \frac{\partial f_i}{\partial v \partial x} & \frac{\partial f_i}{\partial v \partial y} & \frac{\partial f_i}{\partial v \partial \theta} & \frac{\partial f_i}{\partial v^2} & \frac{\partial f_i}{\partial v \partial \omega} \\ \frac{\partial f_i}{\partial \omega \partial x} & \frac{\partial f_i}{\partial \omega \partial y} & \frac{\partial f_i}{\partial \omega \partial \theta} & \frac{\partial f_i}{\partial \omega \partial v} & \frac{\partial f_i}{\partial \omega^2} \end{bmatrix}, \tag{5.54}$$

where θ leads to a matrix of zeros, $\mathbf{0}_{5\times5}$, for $\mathbf{J}_3(\zeta)$. In turn, for x, the matrix $\mathbf{J}_1(\zeta)$ is represented as

$$\mathbf{J}_1(\zeta) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -v_\zeta \cos\theta_\zeta & -\sin\theta_\zeta & 0 \\ 0 & 0 & -\sin\theta_\zeta & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{5.55}$$

where θ_ζ denotes $\theta^* + \alpha(\theta - \theta^*)$ and v_ζ denotes $v^* + \alpha(v - v^*)$, according to the definition of $\zeta \in \{\mathbf{z}^* + \alpha(\mathbf{z} - \mathbf{z}^*) \mid \alpha \in [0,1]\}$. For y it shows $\mathbf{J}_2(\zeta)$ as

$$\mathbf{J}_2(\zeta) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -v_\zeta \sin\theta_\zeta & \cos\theta_\zeta & 0 \\ 0 & 0 & \cos\theta_\zeta & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{5.56}$$

where the maximum absolute value of each element of both matrices is required in (5.51) to obtain the Lagrange remainder over-approximation. Each one of these values is reached by solving an optimization problem which considers variables $\alpha \in [0,1]$ and θ, v, given by the generators of Z.

In (5.51), we reach the values that approximate the Lagrange remainder of the continuous model error, the vector $\mathbf{l}$, where the last component is 0. This vector defines the maximum absolute value of the linearization error and is used to define vector $\tilde{\mathbf{l}}$, that contains the error values between $-\mathbf{l}$ and $\mathbf{l}$. It leads to $(\tilde{\mathbf{l}})_i \in [-(\mathbf{l})_i, (\mathbf{l})_i]$, defining the error interval for x and y, since θ leads to no error.

We can consider the expression (5.48) for $\mathbf{z}$ separated back into $\mathbf{x}$ and $\mathbf{u}$, and the Lagrange term as a vector $\tilde{\mathbf{l}}$. The Taylor series can then be represented as

$$\dot{\mathbf{x}} \in f(\mathbf{x}^*, \mathbf{u}^*) + \frac{\partial f}{\partial x_i}(\mathbf{x}^*, \mathbf{u}^*)(\mathbf{x} - \mathbf{x}^*) + \frac{\partial f}{\partial u_j}(\mathbf{x}^*, \mathbf{u}^*)(\mathbf{u} - \mathbf{u}^*) + \tilde{\mathbf{l}}, \tag{5.57}$$

where $\tilde{\mathbf{l}} \in \mathbb{R}^3$, referring to the state error. A discretization process, similar to the previous section, leads to the model in (5.47) and the discretized $\tilde{\mathbf{l}}_k$. We can consider the input of the system as $\left[\delta\mathbf{u}^\top, \tilde{\mathbf{l}}^\top\right]^\top$, and represent the system similarly to (5.39) as,

$$\begin{bmatrix} \delta\dot{\mathbf{x}} \\ \delta\dot{\mathbf{u}} \\ \dot{\tilde{\mathbf{l}}} \end{bmatrix} = \begin{bmatrix} \mathbf{A}(\mathbf{x}^*, \mathbf{u}^*) & \mathbf{B}(\mathbf{x}^*) & \mathbf{D} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \delta\mathbf{x} \\ \delta\mathbf{u} \\ \tilde{\mathbf{l}} \end{bmatrix} \tag{5.58}$$

where $\mathbf{A}(\mathbf{x}^*, \mathbf{u}^*)$ and $\mathbf{B}(\mathbf{x}^*)$ matrices are represented in (5.38) and $\mathbf{D} = \mathbf{I}_3$. Following the steps described in (5.40)-(5.42), and the consequent matrix exponential computation we reach the desired dis-

cretized system with a term for the over-approximation of the Lagrange remainder. The discretization steps are not explicitly shown here since this process is already described in Section 5.3.

The discretized version of the system in (5.57) can then be represented as the model in (5.47) summed with the term $\mathbf{D}_k\tilde{\mathbf{l}}_k$, related to $\tilde{\mathbf{l}}$, that accounts for the linearization error for the discrete-time model,

$$\delta\mathbf{x}_{k+1} = \begin{bmatrix} 1 & 0 & -hv^* \cdot \sin\theta^* \\ 0 & 1 & hv^* \cdot \cos\theta^* \\ 0 & 0 & 1 \end{bmatrix} \delta\mathbf{x}_k + \begin{bmatrix} h\cos\theta^* & -\frac{h^2}{2}v^*\sin\theta^* \\ h\sin\theta^* & \frac{h^2}{2}v^*\cos\theta^* \\ 0 & h \end{bmatrix} \delta\mathbf{u}_k + \begin{bmatrix} h & 0 & -\frac{h^2}{2}v^*\sin\theta^* \\ 0 & h & \frac{h^2}{2}v^*\cos\theta^* \\ 0 & 0 & h \end{bmatrix} \tilde{\mathbf{l}}_k \quad (5.59)$$

where the matrix on the right is denoted by $\mathbf{D}_k$. Now we are able to define the zonotope L_k, with $\mathbf{D}_k\tilde{\mathbf{l}}_k$ as,

$$\mathbf{D}_k\tilde{\mathbf{l}}_k = \begin{bmatrix} h & 0 & -\frac{h^2}{2}v^*\sin\theta^* \\ 0 & h & \frac{h^2}{2}v^*\cos\theta^* \\ 0 & 0 & h \end{bmatrix} \begin{bmatrix} [-(\mathbf{l}_k)_1, (\mathbf{l}_k)_1] \\ [-(\mathbf{l}_k)_2, (\mathbf{l}_k)_2] \\ 0 \end{bmatrix} \quad (5.60)$$

it leads to generator matrix of L_k as $\mathrm{diag}(\mathbf{D}_k\tilde{\mathbf{l}}_k)$ which yields

$$\mathbf{G}_{L_k} = \begin{bmatrix} h(\mathbf{l}_k)_1 & 0 & 0 \\ 0 & h(\mathbf{l}_k)_2 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad (5.61)$$

and centre $\mathbf{c}_{L_k} = \mathbf{0}_{3\times1}$. With this formulation of L_k the interval of $\tilde{\mathbf{l}}_k$ is assured by the ℓ_∞ norm of the generators of L_k.

As explained in the introduction, given the dynamics rely on $\delta\mathbf{x}_k$ and $\delta\mathbf{u}_k$, this algorithm will consider the state variation set T_k for its computations, which is easily adjustable since T_k and X_k are simply computed through one another. One notation change required given these considerations is $\mathbf{z}_k = [\delta x_k, \delta y_k, \delta\theta_k, \delta v_k, \delta\omega_k]^\top$, and in a similar fashion, the equations presented will refer to $\delta\mathbf{x}_k$ and $\delta\mathbf{u}_k$.

The main idea of this algorithm is to compute the zonotope L_{k+j} in each time step j of the considered MPC horizon. To compute L_{k+j}, we obtain estimates for the next state variation set, $\tilde{T}_{k+j+1}$, obtained by applying the model dynamics to the previous set with U as input and the previous Lagrange remainder as a disturbance. The set U denotes the set of possible inputs $\delta\mathbf{u}$ which characterizes the physical limits of the actuators. Since the inputs at each time step are unknown until the problem resolution, the set U is used as the input. This approximation considers the set evolution for the whole set of possible inputs when, in the control problem, solving the optimization problem leads to a specific value of $\mathbf{u}_k$. As such, it proves to be a major source of over-approximation which provides, however, important safety guarantees. The model estimation process will result in an array of zonotopes L_k to $L_k + N$.

This result is then incorporated into the MPC formulation as disturbances to the computed sets, mathematically expressed as Minkowski sums between the computed sets and the Lagrange remainder approximation, at each time step. This approximation is represented as a zonotope set, denoted by L_{k+j}. The developed set is used to define the safety constraints according to the determined obstacles.

Having explained an overview of the algorithm for the sequential estimation of Lagrange remainders, the mathematical formulation is described in the rest of the subsection. Consider the initial state set zonotope for $j = 0$ and $k = 0$, $X_0 = (\mathbf{G}_{\mathcal{X}_0}, \mathbf{c}_{\mathcal{X}_0}) \subset \mathbb{R}^3$ that represents the initial uncertain state. The initial state $\mathbf{x}_0$ coincides with the centre of X_0. The matrix $\mathbf{G}_{\mathcal{X}_0}$ contains the range of uncertainty for each variable of the state. In the computation of the Lagrange remainders, T_0 is the zonotope of interest, and it can be expressed by the tuple $T_0 = (\mathbf{G}_{\mathcal{T}_0}, \mathbf{c}_{\mathcal{T}_0}) \subset \mathbb{R}^3$ which is closely related to X_0. The affine map

of X_0 with matrix $\mathbf{R}$ as the identity matrix and vector $\mathbf{t} = -\mathbf{x}^*$ leads to the formulation of T_0. We can describe X_0 explicitly as

$$X_0 = \left(\begin{bmatrix} d_x & 0 & 0 \\ 0 & d_y & 0 \\ 0 & 0 & d_\theta \end{bmatrix}, \mathbf{x}_0 \right) \subset \mathbb{R}^3 \tag{5.62}$$

where d_x, d_y and d_θ refer to the uncertainty of x_0, y_0 and θ_0 respectively. The affine map leads to the explicit formulation of T_0

$$\begin{aligned} T_0 &= \mathbf{R}X_0 + \mathbf{t} = (\mathbf{G}_{\mathcal{X}_0}, \mathbf{c}_{\mathcal{X}_0} - \mathbf{x}^*) \subset \mathbb{R}^3 \\ T_0 &= \left(\begin{bmatrix} d_x & 0 & 0 \\ 0 & d_y & 0 \\ 0 & 0 & d_\theta \end{bmatrix}, \delta\mathbf{x}_0 \right) \subset \mathbb{R}^3 \end{aligned} \tag{5.63}$$

with this relation maintained for the zonotopes of the same time step.

The zonotope T_k serves as input in the considered algorithm, as well as the set U, which defines the input limits for velocity $\delta v \in [0, v_{max}]$ and the angular velocity $\delta\omega \in [-\omega_{max}, \omega_{max}]$. This process refers to computing the Lagrange remainder approximation for each time step j. Here we will explain an iteration j of the for-loop, which serves as a generalization for the rest of the iterations. The generalization does not apply to L_k which is computed before the first cycle. The different process is explained at the end of the subsection. The loop begins with computing the estimate of the set $\tilde{T}_{k+j+1}$ by applying the dynamics to $\tilde{T}_{k+j}$ for the set of inputs U. This operation can be defined as a linear map with matrix $\mathbf{A}_k$ from (5.47) followed by a Minkowski sum with zonotope $\mathbf{B}_k U$ described as

$$\mathbf{B}_k U = (\mathbf{B}_k \mathbf{G}_{\mathcal{U}}, \mathbf{B}_k \mathbf{c}_{\mathcal{U}}) = \left(\mathbf{B}_k \begin{bmatrix} \frac{v_{max}}{2} & 0 \\ 0 & \omega_{max} \end{bmatrix}, \mathbf{B}_k \begin{bmatrix} \frac{v_{max}}{2} \\ 0 \end{bmatrix} \right) \subset \mathbb{R}^3 \tag{5.64}$$

where $\mathbf{G}_{\mathcal{U}}$ and $\mathbf{c}_{\mathcal{U}}$ define the zonotope U that encloses the possible input. The zonotope is then used for the Minkowski sum and linear map

$$\mathbf{A}_k \tilde{T}_{k+j} \oplus \mathbf{B}_k U = \left(\left[\mathbf{A}_k \mathbf{G}_{\tilde{\mathcal{T}}_{k+j}} \ \mathbf{B}_k \mathbf{G}_{\mathcal{U}} \right], \mathbf{A}_k \mathbf{c}_{\tilde{\mathcal{T}}_{k+j}} + \mathbf{B}_k \mathbf{c}_{\mathcal{U}} \right) \subset \mathbb{R}^3 \tag{5.65}$$

which represents the propagation of the system's dynamics for the input U. Finally, the estimate $\tilde{T}_{k+j+1}$ is reached by applying a Minkowski sum with the disturbance L_{k+j} to the previous result,

$$\tilde{T}_{k+j+1} = (\mathbf{A}_k \tilde{T}_{k+j} \oplus \mathbf{B}_k U) \oplus L_{k+j} = \left(\left[\mathbf{A}_k \mathbf{G}_{\tilde{\mathcal{T}}_{k+j}} \ \mathbf{B}_k \mathbf{G}_{\mathcal{U}} \ \mathbf{G}_{\mathcal{L}_{k+j}} \right], \mathbf{A}_k \mathbf{c}_{\tilde{\mathcal{T}}_{k+j}} + \mathbf{B}_k \mathbf{c}_{\mathcal{U}} + \mathbf{c}_{\mathcal{L}_{k+j}} \right) \subset \mathbb{R}^3 \tag{5.66}$$

where L_{k+j} is a zonotope characterized by the Lagrange remainder at time step j and is defined by (5.61). The estimate $\tilde{T}_{k+j+1}$ in (5.66) considers a zonotope that represents the variation of $\delta\mathbf{x}_{k+j+1}$. In the next step of updating the set Z_{k+j} to Z_{k+j+1}, the variation of $\mathbf{z}_{k+1}$ and its components must be described in the zonotope Z_{k+j+1}. Taking this into account, the set Z_{k+j+1} is defined as,

$$Z_{k+j+1} = \left(\operatorname{diag} \left(\mathbf{G}_{\tilde{\mathcal{T}}_{k+j+1}}, \begin{bmatrix} \frac{v_{max}}{2} & 0 \\ 0 & \omega_{max} \end{bmatrix} \right), \begin{bmatrix} \mathbf{c}_{\tilde{\mathcal{T}}_{k+j+1}} \\ \frac{v_{max}}{2} \\ 0 \end{bmatrix} \right) \subset \mathbb{R}^5 \tag{5.67}$$

effectively describing the variable variation in $\tilde{T}_{k+j+1}$ where the state variation is defined by matrix $\mathbf{G}_{\tilde{\mathcal{T}}_{k+j+1}}$ and $\mathbf{c}_{\tilde{\mathcal{T}}_{k+j+1}}$. The interval of values of $\delta\mathbf{u}_{k+j+1}$ is defined by the matrix of the second diago-

nal block based on the zonotope U.

The next step refers to optimizing each element of matrices (5.54) for a defined interval of $\alpha \in [0, 1]$, $\delta\theta_{k+j}$, and δv_{k+j}. The possible values of $\delta\theta_k$ can be simply calculated by considering the maximum variation it can suffer in a given time step, which is time step size h times the maximum angular velocity, ω_{max}. At a given time step j, $\delta\theta_{k+j} \in [\delta\theta_{k+j} - d_{\theta_k} - j(h\omega_{max}), \delta\theta_{k+j} + d_{\theta_k} + j(h\omega_{max})]$ with d_{θ_k} denoting the uncertainty associated with $\delta\theta_k$ in the beginning of the model error estimation process. The input δv_{k+j} is always limited by the interval $[0, v_{max}]$. The variable intervals described here serve as input in the optimization problem of separately computing the maximum absolute value for each element of matrices $\mathbf{J}_1(\zeta)$ and $\mathbf{J}_2(\zeta)$.

Lastly, the over-approximation of the Lagrange remainder is calculated with (5.51), which is inserted into the zonotope formulation of L_{k+j+1} similarly to (5.61) and used for the following constraints definition in the MPC. To end the algorithm explanation a final note needs to be made about the step of L_k. The key difference is that the computation of Z_k relies only on zonotope T_k, and not an estimate $\tilde{T}_k$, leading to a different process in computing L_k.

This algorithm returns the array of N zonotopes L_{k+j} that provide an over-approximation of the model error at each time step $k + j$. The process of reaching the estimate is demonstrated, where an over-approximation of the Lagrange remainder is based on estimates of the variation state set $\tilde{T}_{k+j}$ that also introduce conservativeness in the enclosure of the model error. This means the error estimate might frequently prove to be a great overestimation but will never be less than the actual model error, representing the goal of ensuring safety.

In Algorithm 1 it is presented in pseudo-code the Algorithm of Sequential estimation of Lagrange remainders. It shows concisely the steps described in this subsection.

Algorithm 1 Sequential estimation of Lagrange remainder

Inputs: Initial state variation zonotope set T_k, $\mathbf{x}^*$, $\mathbf{u}^*$ and set of possible inputs U
Initialization:
    $\delta\mathbf{x}_k = \mathbf{c}_{\mathcal{T}_k}$
    Obtain the linearized model in (5.47)
for $j = 0$ to horizon N **do**
    Compute $\tilde{T}_{k+j}$ using (5.66)
    Compute Z_{k+j} according to $\tilde{T}_{k+j}$ in (5.67)
    Obtain γ making use of (5.52)
    Obtain $\max(|\mathbf{J}_i(\zeta)|)$ by optimizing (5.54)
    Compute $\mathbf{l}_{k+j}$ with (5.51)
    Compute $\mathbf{L}_{k+j}$ with (5.61)
end for
return L_{k+j}, for $j = 0, ..., N$

5.4.2 MPC formulation

Now that the estimation of the model error is defined, this subsection is dedicated to the mathematical step-by-step explanation that leads to the formulation of the optimization problem (4.5), particularly the constraints' definition.

The MPC optimization problem receives as input the current uncertain state zonotope X_k with state $\mathbf{x}_k \in X_k$. We can start by defining each constraint in (4.5), for $\mathbf{x}_{k+j}$, and respective uncertain set X_{k+j}, and $\mathbf{u}_{k+j}$ as optimization variables. Given the relation, $\mathbf{x}_{k+j} = \delta\mathbf{x}_{k+j} + \mathbf{x}^*$ where $\mathbf{x}^*$ is known, $\delta\mathbf{x}_{k+j}$ is also considered an optimization variable, as well as $\mathbf{p}_{k+j}$ since both derive from $\mathbf{x}_{k+j}$. Similarly, $\delta\mathbf{u}_{k+j}$ is optimized, given $\mathbf{u}_{k+j} = \delta\mathbf{u}_{k+j} + \mathbf{u}^*$.

Consider the state set zonotope $X_{k+j} = (\mathbf{G}_{\mathcal{X}_{k+j}}, \mathbf{c}_{\mathcal{X}_{k+j}}) \subset \mathbb{R}^3$ that represents the uncertain state at time step j and the state variation zonotope $T_{k+j} = (\mathbf{G}_{\mathcal{T}_{k+j}}, \mathbf{c}_{\mathcal{T}_{k+j}}) \subset \mathbb{R}^3$. The expressions for both zonotopes when $k = 0$ and $j = 0$ are found in (5.62) and (5.63), as well as the transformation between them. The transformation can be generalized for a given time step j, from the state set X_{k+j} to the variation state set T_{k+j}. Considering the fact that the centre of X_{k+j} coincides with $\mathbf{x}_{k+j}$ and the centre of T_{k+j} indicates $\delta\mathbf{x}_{k+j}$, the transformation between sets represents the change from the state $\mathbf{x}_{k+j}$ to $\delta\mathbf{x}_{k+j}$. It follows a generalization of this operation for zonotopes X_{k+j} and T_{k+j} as,

$$T_{k+j} = \mathbf{R}X_{k+j} + \mathbf{t} = \left(\mathbf{G}_{\mathcal{X}_{k+j}}, \mathbf{x}_{k+j} - \mathbf{x}^*\right) \subset \mathbb{R}^3 \tag{5.68}$$

with matrix $\mathbf{R}$ as the identity matrix $\mathbf{I}_3$, and vector $\mathbf{t} = -\mathbf{x}^*$, which leads to other possible formulation of $\mathbf{c}_{\mathcal{T}_{k+j}} = \delta\mathbf{x}_{k+j}$. The transformation from T_{k+j} to X_{k+j} simply considers the same affine map but with the inverse sign of vector t.

Following that step, the update of T_{k+j} to T_{k+j+1} uses an affine map that represents the system's evolution,

$$T_{k+j+1} = \mathbf{A}_k T_{k+j} + \mathbf{B}_k \delta\mathbf{u}_{k+j} = \left(\mathbf{A}_k\mathbf{G}_{\mathcal{X}_{k+j}}, \mathbf{A}_k\delta\mathbf{x}_{k+j} + \mathbf{B}_k\delta\mathbf{u}_{k+j}\right) \subset \mathbb{R}^3, \tag{5.69}$$

where matrices $\mathbf{A}_k$ and $\mathbf{B}_k$ refer to the linearized model.

The state set X_{k+j+1} is then obtained by considering the reverse operation of (5.68) with the resulting set from a Minkowski sum between T_{k+j+1} and disturbance L_{k+j} associated with the previous time step. The state set is then represented as follows,

$$X_{k+j+1} = \mathbf{R}(T_{k+j+1} \oplus L_{k+j}) + \mathbf{t} = \left(\left[\mathbf{A}_k\mathbf{G}_{\mathcal{X}_{k+j}}\ \mathbf{G}_{\mathcal{L}_{k+j}}\right], \mathbf{A}_k\delta\mathbf{x}_{k+j} + \mathbf{B}_k\delta\mathbf{u}_{k+j} + \mathbf{x}^*\right) \subset \mathbb{R}^3, \tag{5.70}$$

effectively describing function g defined in the MPC problem description (4.5).

The state progression to $\mathbf{x}_{k+j+1}$ is equivalent to the centre of the uncertain set X_{k+j+1} that is propagated from X_{k+j} through function g.

The last step to fully represent the MPC is to define the obstacle constraint, more specifically, the condition of $\mathbf{p} \in X_{k+j}$. To impose the position limitations, the constraint should reflect the uncertainty of the state and limit the whole set to the designated position state space. To accomplish that, we can express the position $\mathbf{p}$ as $\mathbf{p}_{k+j}$ and shift the obstacles by the adequate uncertainty distance that characterizes the set X_{k+j}. This distance can be approximated as the sum of the elements in the generator matrix $\mathbf{G}_{\mathcal{X}_{k+j}}$ associated with the specified state component. The first row refers to the uncertainty of x_{k+j} and the second to the uncertainty of y_{k+j}. The constraint can then be defined as,

$$\mathbf{o} \leq \mathbf{p}_{k+j} \leq \mathbf{q}, \qquad \mathbf{o} = \begin{bmatrix} o_x + \sum_i^n (g_{1i}) \\ o_y + \sum_i^n (g_{2i}) \end{bmatrix},\ \mathbf{q} = \begin{bmatrix} q_x - \sum_i^n (g_{1i}) \\ q_y - \sum_i^n (g_{2i}) \end{bmatrix}, \tag{5.71}$$

where g_{1i} denotes the coordinate of the first row and i^{th} column of the matrix $\mathbf{G}_{\mathcal{X}_{k+j}}$, and g_{2i} denotes the elements of the second row. With this formulation, the constraint effectively prevents the whole set from crossing the walls defined by elements (o_x, o_y, q_x, q_y).

Finally, the MPC optimization problem outlined in Chapter 4 is fully described. The constraints are clearly defined, enabling the next step of solving the optimization problem.

5.4.3 Next state estimation

The last processes of the considered loop in Algorithm 2 refer to the state estimation problem defined in Subsection 5.1.2. After the MPC problem is solved to return input $\mathbf{u}_k$, the goal is to estimate the next state, based on the system's dynamics and a measurement set. Considering the zonotope that describes the state uncertainty X_k as the enclosure $\mathcal{O}_{k-1}$ in (5.17), the computation of X_{k+1} is clearly defined. Firstly, we consider the propagation of X_k as $\hat{X}_{k+1}$ which has a very similar formulation as X_{k+j+1} in (5.70) since (5.68)-(5.70) considers a similar evolution as from X_k to $\hat{X}_{k+1}$. The sequence can be adapted as,

$$\hat{X}_{k+1} = \mathbf{R}((\mathbf{A}_k T_k + \mathbf{B}_k \delta\mathbf{u}_k) \oplus L_k) + \mathbf{t} = ([\mathbf{A}_k \mathbf{G}_{\mathcal{X}_k} \ \mathbf{G}_{\mathcal{L}_k}], \mathbf{A}_k \delta\mathbf{x}_k + \mathbf{B}_k \delta\mathbf{u}_k + \mathbf{x}^*) \subset \mathbb{R}^3, \tag{5.72}$$

where T_k denotes the zonotope set related to X_k that refers to $\delta\mathbf{x}_k$ instead of $\mathbf{x}_k$, L_k denotes the model error disturbance applied on the dynamics and is computed in Algorithm 1 for $j = 0$. The matrix R represents the identity matrix and $\mathbf{t} = \mathbf{x}^*$ that defines the affine map from $\hat{T}_{k+1}$ to $\hat{X}_{k+1}$. Matrices $\mathbf{A}_k$ and $\mathbf{B}_k$ from the model, define the affine map that characterizes the propagation of the dynamics, along with $\delta\mathbf{x}_k$ and $\delta\mathbf{u}_k$.

The next step in the state estimation consists of numerically integrating the dynamics, given $\mathbf{x}_k$ and $\mathbf{u}_k$. This is done through the MATLAB function $ode45$ and it allows the algorithm to consider a measurement set based on the result. Given the state obtained by the numerical integration, we consider a tunable sensor noise to define it as a measurement set. Following (5.17), the update of the state set to X_{k+1} is done by considering the intersection of the estimate $\hat{X}_{k+1}$ and this measurement set.

As described in Subsection 5.1.1, the zonotope representation is not closed under the intersection set operation, which motivates the use of an over-approximation method in the simulations to define the result as a zonotope.

With the computation of the state uncertainty set X_{k+1} and state $\mathbf{x}_{k+1}$ as the centre of the set, the solution is now fully described. The for-cycle in Algorithm 2 generates consecutive states that target a determined position reference and guarantee safety conditions.

The state estimation in the MPC considers disturbances L that are produced assuming the worst-case scenario of every possible actuation. The evolution of these consecutive state estimates which include a Minkowski sum with L, leads to a very conservative obstacle avoidance constraint in (5.71). In the Results Chapter, we will assess the capacity of this process to ensure passive safety, since the major overestimation of the uncertain state might ensure safety for a number of time steps, with null or unwanted actuation that would resemble actuation failures.

As such, the solution presented here describes an algorithm that generates passively safe trajectories as were the objectives defined for this thesis.

5.4.4 Overview

An overview of the proposed method in represented in Algorithm 2, written in pseudo-code. The algorithm description shows the required steps to adequately construct the MPC problem and the processes after solving the problem that lead to the design of a safe trajectory. These steps were explained throughout the section.

In the next chapter, the proposed solution is tested for multiple parameters to better understand the quality of the performance and its limitations.

Algorithm 2 Trajectory generation Algorithm

1: **Inputs:** Initial uncertain state set X_0 and target position $\mathbf{p}_{ref}$
2: **for** $k = 0$ to simulation time N_T **do**
3: Compute model error estimates as L_k to L_{k+N} (Algorithm 1)
4: Construct the MPC formulation (4.5), with safety constraints in (5.71) based on estimates developed with (5.70)
5: Solve the MPC problem, obtaining $\mathbf{u}_k$
6: Propagate X_k in (5.72)
7: Numerical integration of the dynamics to obtain measurement set
8: Update X_{k+1} from intersection of X_k propagation and measurement set
9: **end for**
10: **return** trajectory

Chapter 6

Results

This chapter presents the simulation results for the previously described MPC algorithm. The chapter will show many cases that will try to evaluate the approach given different parameter variations.

These will be assessed by performance metrics such as the ability to remain safe throughout the simulation, the accuracy of reaching the goal position, and other aspects of the generated trajectory, such as the control effort and computational efficiency. Since the problem refers to tracking a reference position, we could consider the simulation to be finished when the state is near the position goal. However, in the simulations, we consider the full simulation time to better comprehend the system's behaviour. The time step of the minimum positional error will still be recorded as a performance metric. The elapsed time of the algorithm is also reported, having minor significance in terms of actual values and more in terms of relative values to other simulations. Since the code could be optimized or performed with better hardware, the time elapsed is not considered a good measure of computational efficiency. We can, however, analyze the results of different simulations to understand the time-consuming processes.

Throughout the simulation process, we will consider the use of a highly accurate sensor, and the intersection step will only return the set of measures from $ode45$. This is done because methods that approximate the intersection operation between zonotopes were not successfully implemented during the available time. In the approximation methods tested, the uncertainty would keep increasing beyond the measure uncertainty, which does not accurately represent the intersection operation. By updating the state as the result of the sensor measure, the system will have a constant uncertainty value, defined by the measurement uncertainty. In an accurate representation of the intersection operation, the uncertainty associated with the state could be reduced since the intersection of the state prediction and the sensor measure produces a set with equal or less uncertainty than the sensor measure.

Denoting the measure set as M_{k+1} for each time step of the simulation, the next state X_{k+1} will be defined as just the measurement set $X_{k+1} = M_{k+1}$. The measurement zonotope centre $\mathbf{c}_{\mathcal{M}_{k+1}}$, is given by the MATLAB function to measure the actual nonlinear dynamics, and it will be denoted as $\mathbf{m}_{k+1}$. The generator matrix $\mathbf{G}_{\mathcal{M}_{k+1}}$ is a tunable parameter that considers the uncertainty associated with the measure. This consideration also helps establish a constant uncertainty value for the state, defined by $\mathbf{G}_{\mathcal{M}_{k+1}}$.

When considering a simulation without obstacles, the safety constraints will consider a obstacle wall with a very large value (in the order of 10^{99}) to ensure the compliance with the safety constraints at any point. Since the obstacles limit the maximum and minimum values of the position coordinates, an obstacle at $x = 10^{99}$ metres, for example, will not affect the generated trajectory since the given parameters can not lead to a state close to the obstacle.

Furthermore, this algorithm was implemented in Matlab, and the optimization problems were solved using Gurobi.

6.1 Default simulation

Firstly, we show the simulation results that will be considered the baseline for the parameter variation. The parameter default values are presented in Table 6.1.

Table 6.1: Default simulation parameters

Parameter	Value	Description
$\mathbf{G}_{\mathcal{X}_0}$	$0.01\mathbf{I}_3$ (m, m, rad)	Initial state uncertainty
$\mathbf{x}^*$	$[0, 0, \frac{2\pi}{5}]^\top$ (m, m, rad)	Initial state
$\mathbf{u}^*$	$[0.05, 0]^\top$ (m/s , rad/s)	Initial input
$\delta\mathbf{u}_{max}$	$[1, \frac{\pi}{4}]^\top$ (m/s , rad/s)	Maximum actuation values (define set U)
$\mathbf{p}_{ref}$	$[0.15, 2]^\top$ m	Position reference
h	0.1 s	Time step size
N_T	3 s	Simulation time
N	4	MPC prediction horizon
R	1000	Actuation cost
$\mathbf{Q}$	$10^4\mathbf{I}_2$	Position error weight matrix
$\mathbf{Q}_f$	$10^6\mathbf{I}_2$	Final position error weight matrix
$\mathbf{G}_{\mathcal{M}}$	$0.01\mathbf{I}_3$ (m, m, rad)	Measurement uncertainty matrix
$\mathbf{o}$	$-10^{99}[1, 1]^\top$ m	Obstacle limiting the position minimum
$\mathbf{q}$	$0.22[1, 10^{99}]^\top$ m	Obstacle limiting the position maximum

The simulation based on these parameters is a simple test where the unicycle needs to turn to avoid the wall defined by $\mathbf{q}$ while trying to reach a position reference close to it. This first test aims to clearly show the algorithm's functioning in Fig. 6.1 and Fig. 6.2. It is important to note the difference in the x and y axis scales. It leads to a transformation of the state set, from a square to the small rectangle observed in Fig. 6.1. Although this change is unwanted, the axis are represented as such to better visualize the system trajectory, as considering the same scale for both coordinates would lead to representing the trajectory as a straight line, since the variation is much larger in the y coordinate.

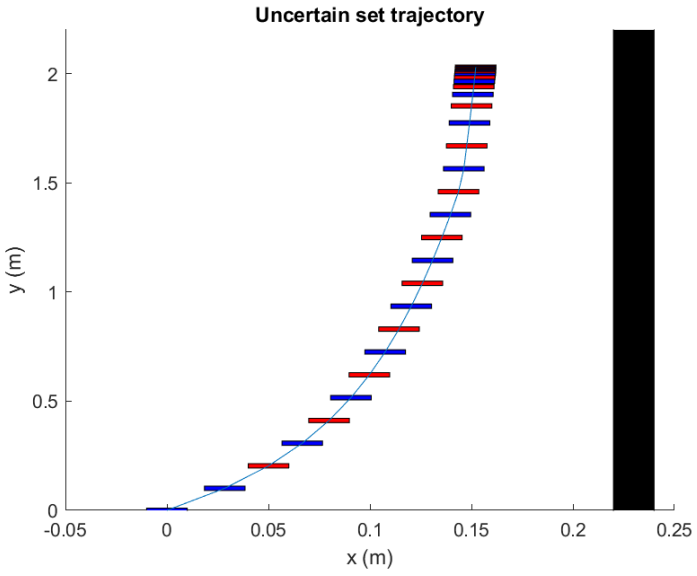


Figure 6.1: Representation of the uncertain state set trajectory. Alternating in red and blue are the consecutive representations for each time step of the simulations. The block in black shows the obstacle at $x = 0.22$ meters.

The following table 6.2, shows the results where $e_{\mathbf{p}}$ refers to the minimum error ($\|\mathbf{p} - \mathbf{p}_{ref}\|$) and the time step $t(e_{\mathbf{p}})$ denotes the step on which $e_{\mathbf{p}}$ is recorded.

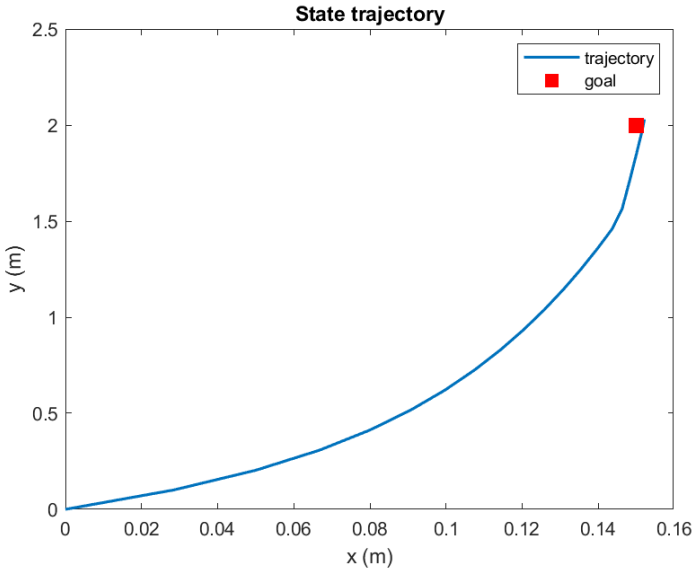


Figure 6.2: Evolution of the system's position in the given reference and obstacle conditions.

Table 6.2: Default simulation results

Time elapsed (s)	$e_{\mathbf{p}}$ (m)	$t(e_{\mathbf{p}})$
67.398	0.0749	25

Analyzing both figures and the table, the algorithm stays clear of the obstacle and reaches close to the reference. While formulating the MPC optimization problem, the algorithm considers the state estimation described in the previous chapter. As referenced in Subsection 5.4.3, it leads to a very conservative obstacle avoidance constraint that produces the slight movement by the end of the simulation noticed in Fig. 6.2, despite not being close to collision in the next time step. The inputs represented in Fig. 6.3 characterize the last movement, increasing the angular velocity for one step in the middle of the simulation. This movement is made clear in the next sections with the comparison made later for simulations with different obstacles.

All figures specific to a certain simulation will have the format described for this simulation and the performance metrics as in Table 6.2. We will evaluate the system's performance for the parameter variations of most interest to the simulation by analyzing both the results and the figures that characterize the simulations.

6.2 Obstacle Variation

Considering different obstacles that, similarly to the previous simulation, restrict the maximum value of x. Tests were made for the variations in Table 6.3.

Table 6.3: Obstacle variation test parameters

Simulation index	Parameter changed	Value
Default	$\mathbf{q}$	$0.22[1, 10^{99}]^\top$ m
1	$\mathbf{q}$	$0.15[1, 10^{99}]^\top$ m
2	$\mathbf{q}$	$0.19[1, 10^{99}]^\top$ m
3	$\mathbf{q}$	$10^{99}[1, 1]^\top$ m

Consider the plots in Fig. 6.4, where the uncertain state set trajectory is represented. In Fig. 6.5, the

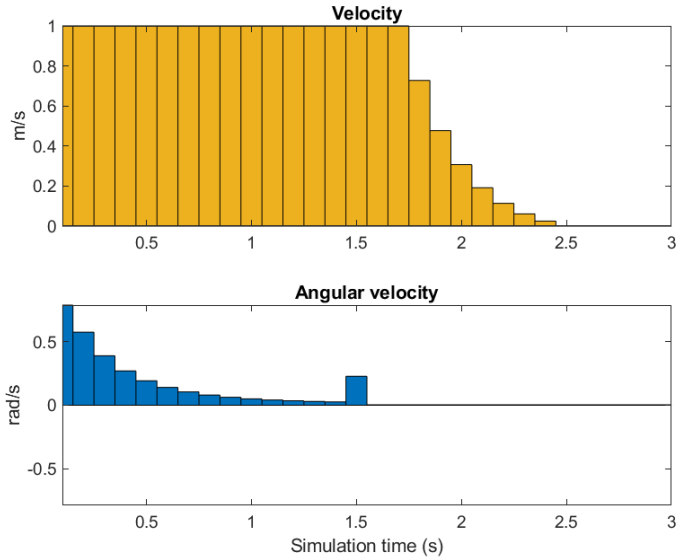
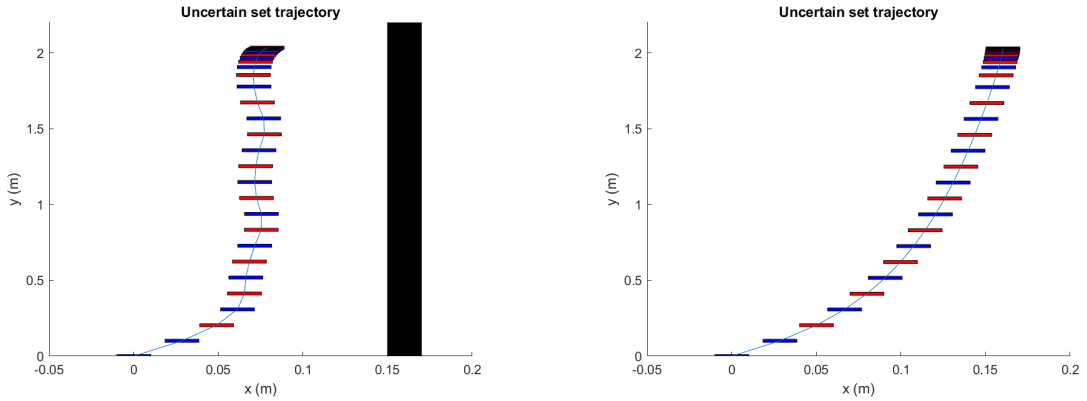


Figure 6.3: Representation of the input v and ω, where the plots are limited to the respective maximum actuation

state trajectory of each of the four simulations is presented.



(a) Representation of the uncertain state set trajectory given the obstacle at $x = 0.15$ meters

(b) Representation of the uncertain state set trajectory given no obstacle restriction

Figure 6.4: Uncertain state evolution for the first and third simulations

Given the values of R, $\mathbf{Q}$ and $\mathbf{Q}_f$ that characterize the MPC optimization problem, the generated trajectory behaves as in Fig. 6.4b. Also shown in Fig. 6.5, the trajectory with no obstacle sacrifices some accuracy in the final state, leading to a smoother and less saturated actuation process than for the default simulation. Considering a longer prediction horizon should improve final accuracy, as seen in the following Section 6.4. Also, the definition of v^* different than zero leads to constant small movements forward in iterations the MPC would consider input 0.

The other simulations produce much different trajectories, impacted by the closer wall. Referring again to the state estimation process of the MPC as the cause of considering states remotely close to the obstacle as unsafe. It is important to note that after the intersection process, the uncertain sets always define the same uncertainty of 0.01 meters or 0.01 radians in the position and orientation angle, respectively. This would mean that, theoretically, the state could go close to 1 centimetre from the wall and still be safe. The algorithm does not consider the state safe given the large disturbances associated

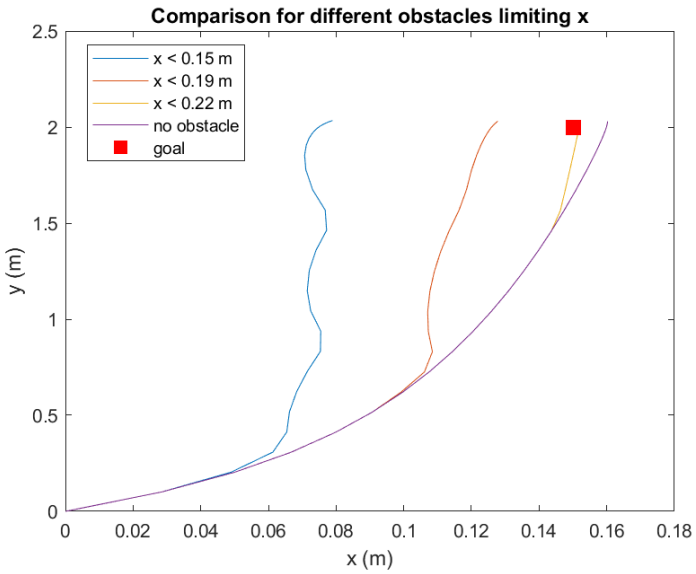


Figure 6.5: Representation of the state trajectory for the multiple simulations of different obstacle values

with the state estimation in the MPC. The robustness of the approximation in the state estimation and consequent safety constraint act as a condition of passive safety, since it performs avoidance manoeuvres some time steps before it is actually necessary.

In Fig. 6.6, the input variation is represented for each simulation considered, denoting clearly the movements of avoidance by increasing angular velocity ω, which means more effort from the actuators. The results of the simulations are shown in Table 6.4, confirming the observations made. The accuracy difference of the first two simulations concentrates mostly on the x value.

Table 6.4: Obstacle variation simulation results

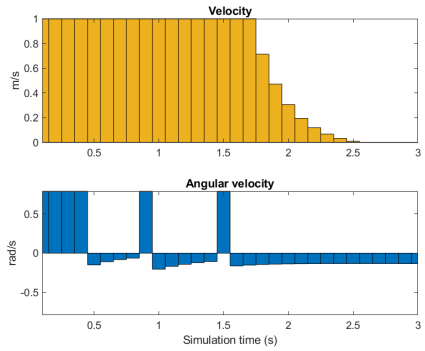
Simulation index	Time elapsed (s)	$e_{\mathbf{p}}$ (m)	$t(e_{\mathbf{p}})$
Default	66.509	0.00184	24
1	67.398	0.0749	25
2	66.389	0.0240	24
3	67.122	0.00991	24

6.3 MPC state estimation

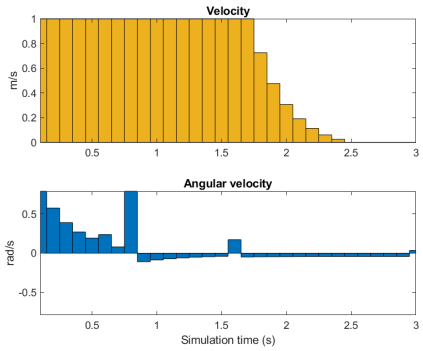
In this section, we illustrate the steps of the state estimation process that lead to the safety constraints in the MPC and its solution for the optimal control sequence. These representations are done based on the default simulation parameters in 6.1, with the only change being the obstacle $\mathbf{q}$ to $\mathbf{q} = 0.15[1, 10^{99}]^\top$.

Firstly, the algorithm defines the zonotopes L that over-approximate the model error for estimates of the state considering the evolution for the whole set of possible inputs. This process, described in Algorithm 1, is illustrated in Fig. 6.7a, where the sequential state estimates are represented. The final estimate of the state, with the addition of L as disturbance, is illustrated in Fig. 6.7b. Both representations are obtained for time step $k = 2$.

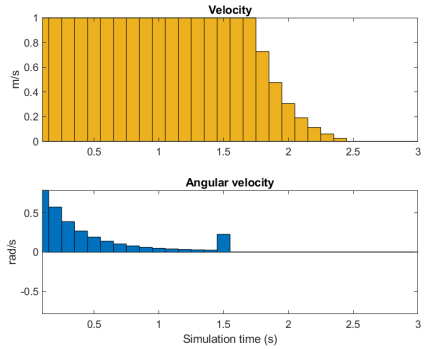
Given the time step of this simulation, $k = 2$ and the initial orientation angle of the system being $\theta = \frac{2\pi}{5}$, the estimates expand more in that similar direction. This is why, in Fig. 6.7a, the estimates show a much larger interval for y than for x. Similarly, the expansion in the opposite direction towards the origin is significantly smaller.



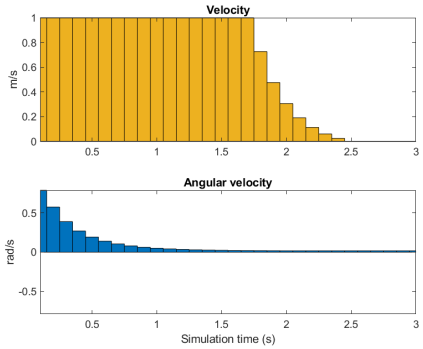
(a) Simulation 1, with $x < 0.15$m



(b) Simulation 2, with $x < 0.19$m

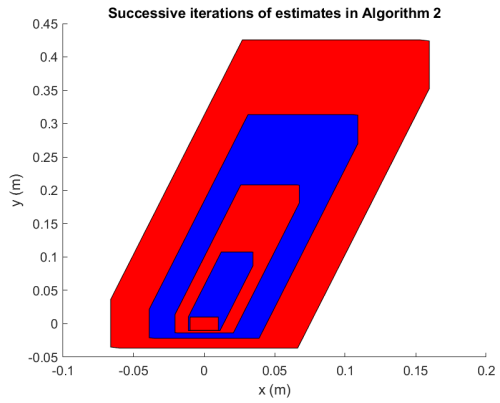


(c) Default simulation, with $x < 0.22$m

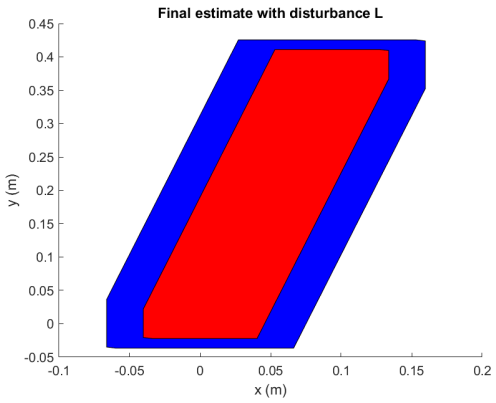


(d) Simulation 3, with no obstacle

Figure 6.6: Input values variation for all four simulations with different obstacles



(a) Representation of consecutive estimates. The sets in red and blue define different time steps of fastly growing estimates. The initial state is represented by the smallest zonotope in the middle, which is used to compute the next smallest estimate in blue. The rest of the estimates follow the pattern.



(b) Representation of the final estimate used to compute L. The blue set highlights the disturbance set added to the previous estimate to form this final set

Figure 6.7: Representations of the process of over-approximating the model error

The results of this process are represented as zonotopes L for each time step $k+j$, with $j = 0, \ldots, N$. Shown in Table 6.5 are the uncertainty values of the generator matrix of L. The values are denoted as l_x, l_y and l_θ for the uncertainty of state components, x, y and θ, respectively.

Table 6.5: Model error estimation results

Time step $k+j$	l_x (m)	l_y (m)	l_θ (rad)
k	$9.759e-04$	$2.480e-04$	0
$k+1$	0.00888	0.00321	0
$k+2$	0.0172	0.00794	0
$k+3$	0.0260	0.0144	0
$k+4$	0.0352	0.0225	0

These zonotopes are inserted into the MPC formulation. We represent the control sequence result in Fig. 6.8 for the iteration that considers the previous model error computations in Table 6.5.

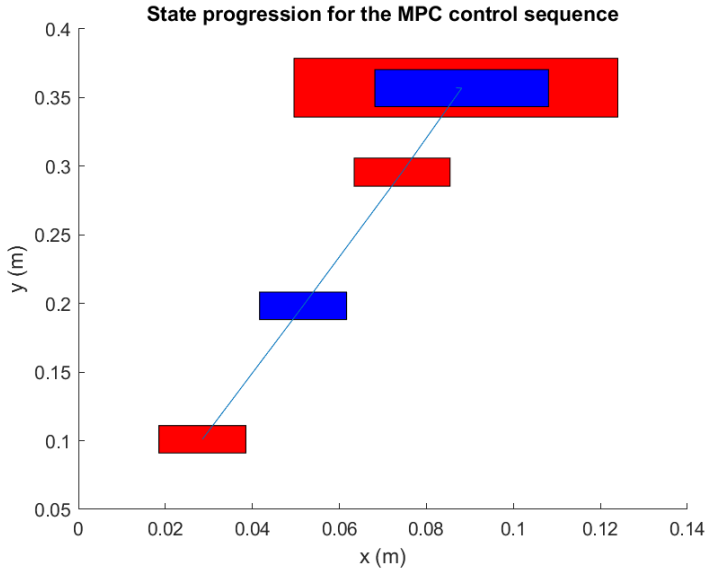


Figure 6.8: Representation of the estimated uncertain state trajectory given by the optimal control input sequence.

Table 6.6 represents the optimized control sequence, of $\delta\mathbf{u}_{k+j}$ with $j = 0, \ldots, N-1$. Of this sequence, only the input $\mathbf{u}_k$, given by $\mathbf{u}_k = \delta\mathbf{u}_k + \mathbf{u}^*$ is used for the consequent state measurement.

Table 6.6: MPC optimal control sequence, for iteration $k = 2$.

Time step $k+j$	δv (m/s)	$\delta\omega$ (rad/s)
k	1.000	0.785
$k+1$	1.000	0.785
$k+2$	0.627	0.785
$k+3$	$2.429e-11$	0.785

Observing the results of the MPC iteration and the model error disturbances, the conservativeness of the safety condition is clearly visible. Since the model error disturbances are increasingly larger, and there is no update to the state with state measures, the uncertainty associated with the estimated state rapidly grows. This leads to the low values of actuation δv in the last estimates to precisely avoid the obstacle at $x = 0.15$ meters. The robustness associated with this method can serve as a condition of passive safety since the MPC will always consider this overly conservative state progression of exponential uncertainty growth. An important note to these results is that they were obtained for a small MPC

horizon. Increasing the horizon N by a small value will lead to a substantial increase in the uncertainty for the already conservative solution.

6.4 MPC horizon variation

Considering now, the simulation of the situations with different obstacles in Table 6.3 for an increased MPC horizon of 6 time steps. The previous section clearly shows the results of increased uncertainty during the state estimation of the MPC and Fig. 6.9 represents its effects on the state trajectory generated in the default situation for horizon $N = 6$. Fig. 6.10 describes the state trajectory of all four of the different obstacles considered.

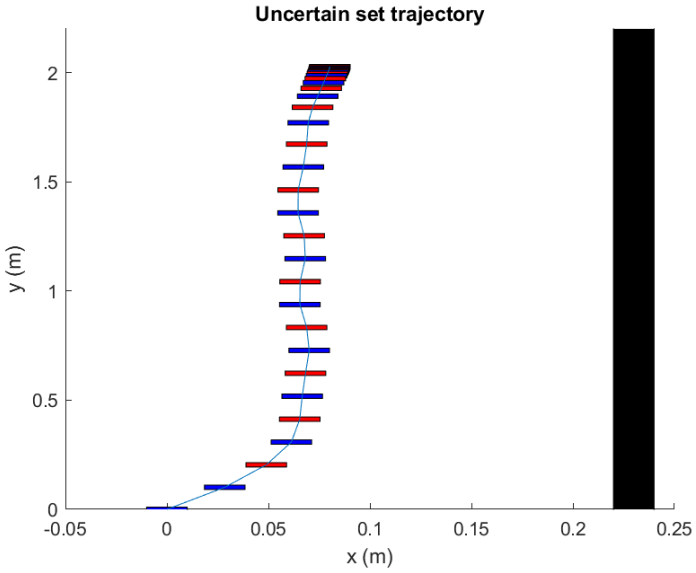


Figure 6.9: Trajectory of the uncertain state set, with an obstacle $x < 0.22$ m

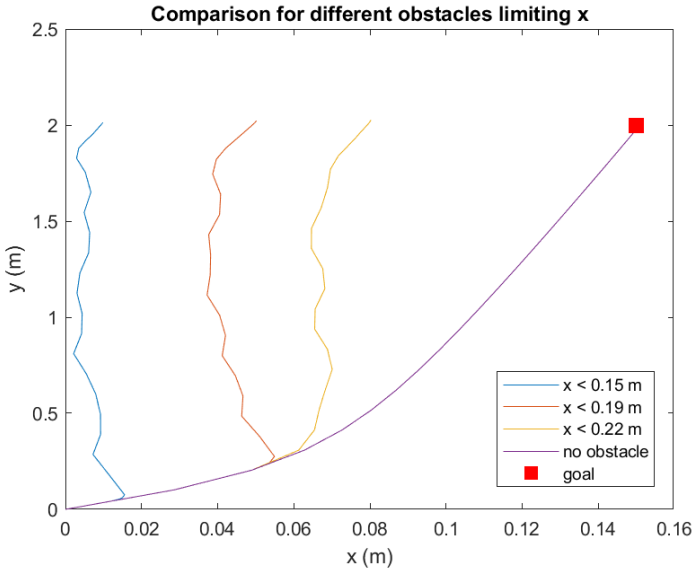


Figure 6.10: State trajectory simulation for different obstacles and increased horizon N

As expected, the generated trajectory stays much farther from the obstacle since the safety constraint

is based on 6 state estimates computed with the disturbances L and no updates to reduce uncertainty. The overly conservative safety condition leads to the unwanted result of not accurately following the reference.

The results of these simulations are shown in Table 6.7.

Table 6.7: Test results for increased horizon and different obstacles

Simulation obstacle	Time elapsed (s)	$e_{\mathbf{p}}$ (m)	$t(e_{\mathbf{p}})$
$x < 0.15$ m	92.657	0.141	29
$x < 0.19$ m	92.244	0.101	27
$x < 0.22$ m	92.441	0.0706	25
$x < 10^{99}$ m	97.896	0.00102	25

It is possible to notice, in the Fig. 6.10 and Table 6.7, that the position error for the simulations with obstacles is concentrated in the x value. This happens because of the safety conditions. In fact, we can note the difference in the minimum position error $e_{\mathbf{p}}$ between the first three simulations as closely related to the difference between the x limits given by the obstacle. The distance of obstacles of the first two simulations is 4 centimetres as well as their difference $e_{\mathbf{p}}$. The same occurs with the other simulations. Another aspect to note is the increase in time elapsed while running the simulation, when compared to the other simulations with $N = 4$, in Table 6.2. The variation almost matches the 50% increase in the horizon, which points to the optimization problems in computing the zonotopes L, as one of the main sources of computational complexity. The simulation time and step size h did not change, so the increase is based on the L calculations and MPC construction.

Using a smaller horizon is also possible, but it can limit the prediction capacity of the system to react to obstacles and anticipate future violations of the constraints. The default simulation horizon is already considerably small at $N = 4$, so lowering it could lead to many complications, especially in a real problem and not in a simulation environment.

6.5 State uncertainty variation

One of the parameters with more implications for the algorithm's performance is the initial uncertainty of the state and the uncertainty of the measures. It follows the simulation results when considering the change of the uncertainty values from the default simulation. The parameters are shown in Table 6.8.

Table 6.8: Variation of uncertainty parameters from the default simulation

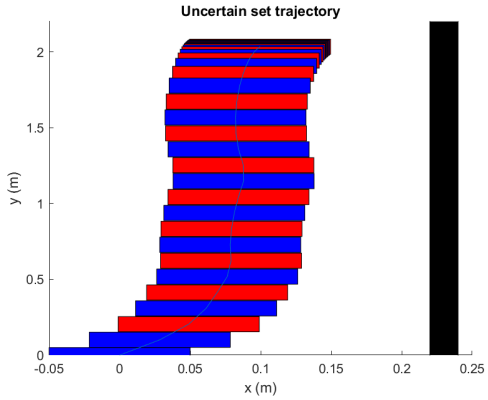
Simulation id	Parameter	New value	Parameter	New value
1	$\mathbf{G}_{\mathcal{X}_0}$	$0.05\mathbf{I}_3$	$\mathbf{G}_{\mathcal{M}}$	$0.05\mathbf{I}_3$
2	$\mathbf{G}_{\mathcal{X}_0}$	$0.005\mathbf{I}_3$	$\mathbf{G}_{\mathcal{M}}$	$0.005\mathbf{I}_3$

The simulations in Fig. 6.11, Fig. 6.12 and Fig. 6.13 illustrate the performance of the solution for these new uncertainty values. Table 6.9, shows the performance metrics of both simulations

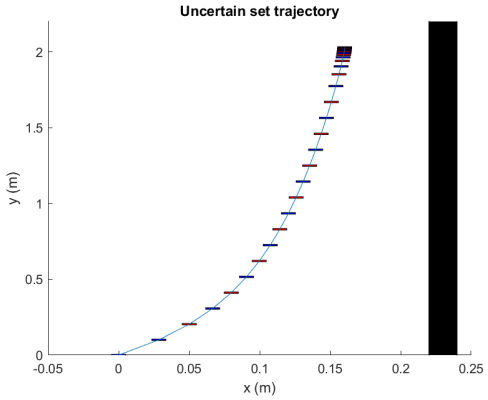
Table 6.9: Test results for different uncertainty values

Uncertainty value	Time elapsed (s)	$e_{\mathbf{p}}$ (m)	$t(e_{\mathbf{p}})$
$0.05\mathbf{I}_3$	68.779	0.0543	24
$0.005\mathbf{I}_3$	66.688	0.00991	24

The difference in the uncertainty values leads to very different system behaviour. The trajectory of the state with larger uncertainty values resembles the trajectory of those that consider a closer obstacle.

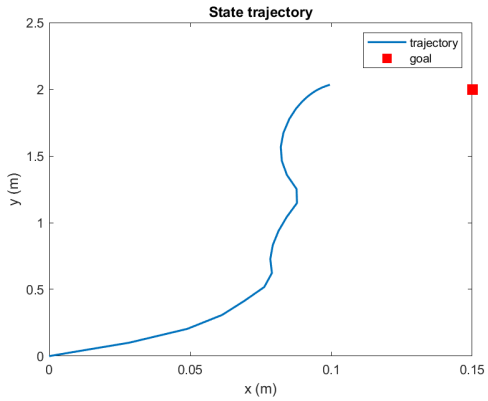


(a) Uncertain state trajectory for sets with 5 cm of uncertainty.

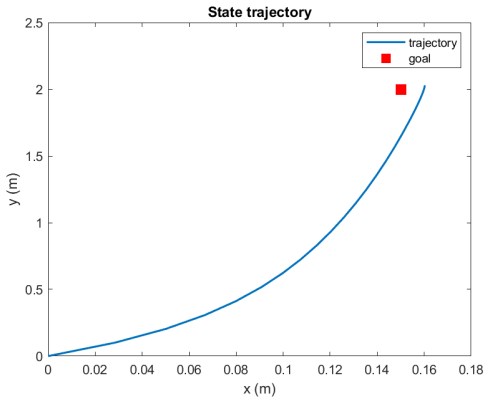


(b) Uncertain state trajectory for sets with 5 mm of uncertainty

Figure 6.11: Side-by-side representations of the uncertain state trajectories for both simulations

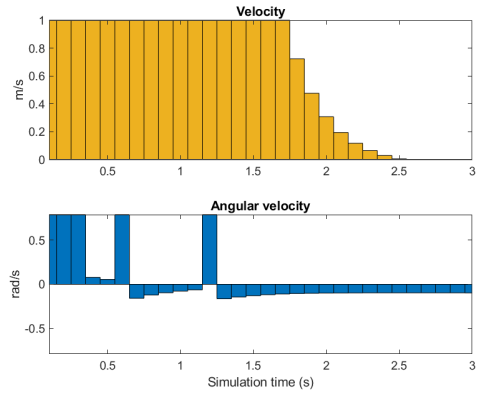


(a) State evolution for sets with 5 cm of uncertainty.

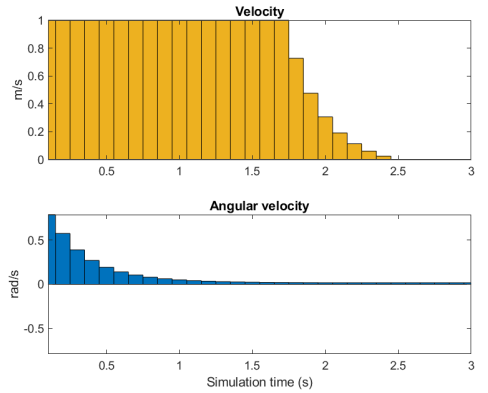


(b) State evolution for sets with 5 mm of uncertainty

Figure 6.12: Representations of the state trajectories for both simulations



(a) Input values for sets with 5 cm of uncertainty.



(b) Input values for sets with 5 mm of uncertainty

Figure 6.13: Representation of the input sequence for both simulations

The increased state uncertainty leads to a more conservative safety constraint since the size of the model error disturbances L are computed from the initial state, adding to its uncertainty. In turn, the simulation with a smaller uncertainty matches the behaviour of a simulation without obstacle, comparing Fig. 6.12b to the line of no obstacles in Fig. 6.5. The state uncertainty is small enough that the state can be closer to the obstacle without violating the safety constraint.

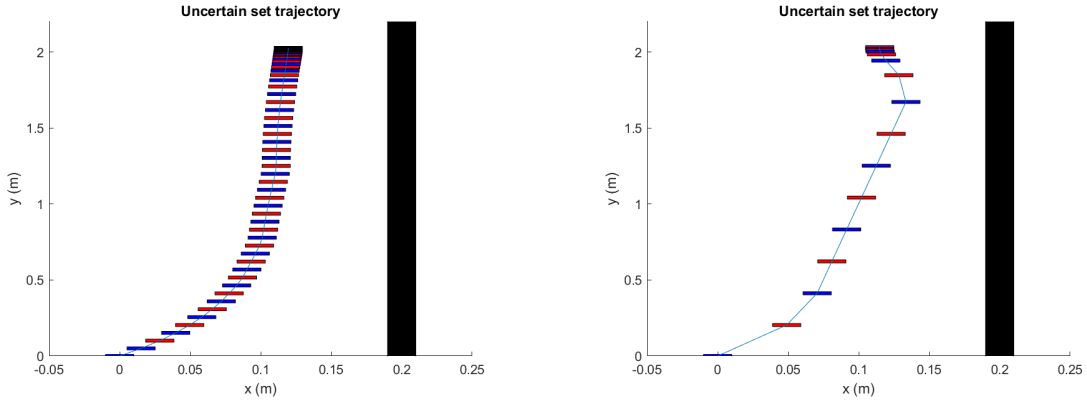
6.6 Time step size variation

We can record the following simulations to infer the performance change for variations in the time step size in Table 6.10. The goal is to keep the simulation time and the horizon N parameters, altering N to correspond to the same fraction of simulation time N_T. The simulations consider a closer obstacle than the default simulation to increase its influence in the representation and better analyze the influence of the time step size h.

Table 6.10: Variation of parameters from the default simulation

Simulation id	Parameter	New value	Parameter	New value	Parameter	New value
1	h	0.05 s	N	8	x max limit (q_x)	0.19 m
2	h	0.2 s	N	2	x max limit (q_x)	0.19 m

The figures that represent each simulation are shown side by side in Fig. 6.14, Fig. 6.15 and Fig. 6.16



(a) Uncertain state trajectory for $h = 0.05$ s.

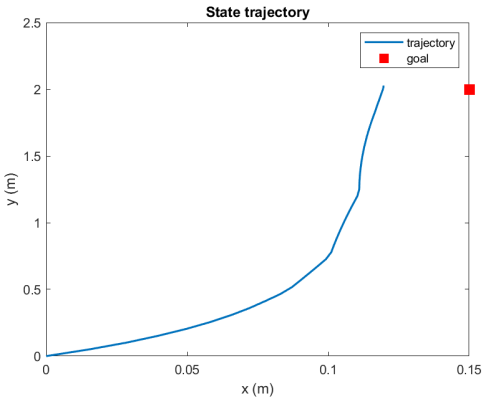
(b) Uncertain state trajectory for $h = 0.2$ s.

Figure 6.14: Side-by-side representations of the uncertain state trajectories for both simulations

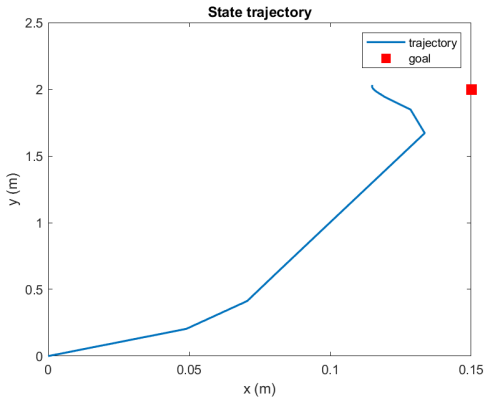
Observing the figures, the algorithm successfully guarantees safety for both cases, showing again a conservative trajectory that does not move close to the obstacle. The simulation with a larger step size in Fig. 6.14b leads to a single extreme avoidance manoeuvre, changing considerably the orientation angle and moving away from the reference in the x axis. The movement is expected given the longer step size h, which leads to more aggressive actions, when compared to the results for $h = 0.05$ seconds, that reveal more and smaller manoeuvres to not violate the system constraints.

6.7 Actuation variation

The final aspect of the simulation environment to be tested here is the change of actuation. A major factor of the simulations made is the actuation, as such, the last simulations will change the parameters

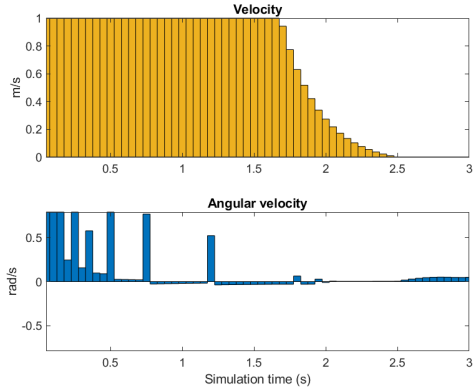


(a) State evolution for $h = 0.05$ s.

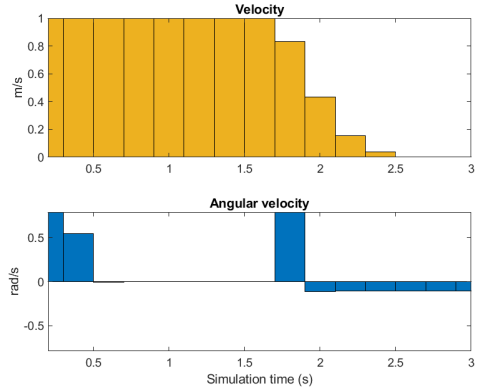


(b) State evolution for $h = 0.2$ s.

Figure 6.15: Representations of the state trajectories for both simulations



(a) Input values for $h = 0.05$ s.



(b) Input values for $h = 0.2$ s.

Figure 6.16: Representation of the input sequence for both simulations

of maximum actuation to record the changes in the performance. It should be noted that these values are associated with the physical limitations of the actuators.

Here we will consider the parameters change in Table 6.11, for four simulations with the different obstacles in Table 6.12.

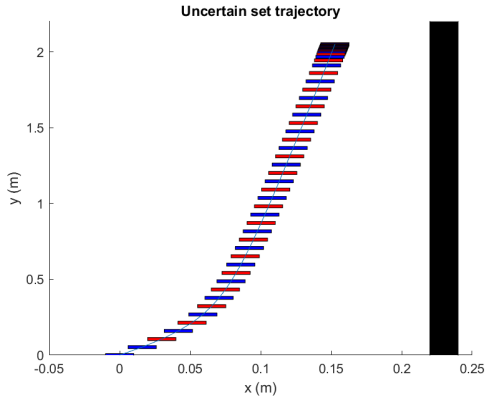
Table 6.11: Variation of parameters from the default simulation

Parameter	New value	Parameter	New value	Parameter	New value
δv_{max}	0.5 m/s	$\delta \omega_{max}$	$\frac{\pi}{8}$ rad/s	N_T	5 s

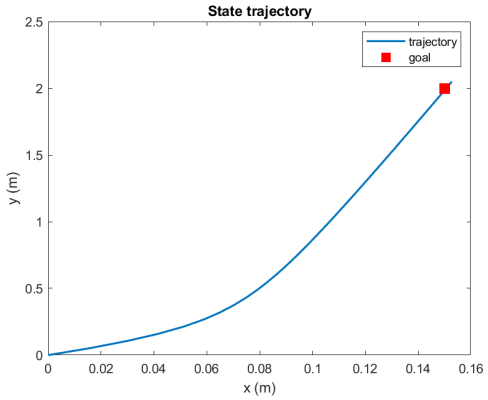
Table 6.12: Obstacle variation test parameters

Simulation id	obstacle wall	Value
1	q_x	0.15 m
2	q_x	0.19 m
3	q_x	0.22 m
4	q_x	10^{99} m

The representation of the uncertain state trajectory for simulation 3 is in Fig. 6.17a. Fig. 6.17b shows the state trajectory of the same simulation, and finally, in Fig. 6.18, the comparison between each trajectory



(a) Representation of the uncertain state set trajectory given the obstacle at $x = 0.22$ meters



(b) Representation of the state trajectory given the obstacle at $x = 0.22$ meters

Figure 6.17: Result of simulation 3, with the same obstacle as the default simulation

Fig. 6.17a and Fig. 6.17b, compared to the default simulation, show better reference following. This is because the set of inputs U is present in the disturbances calculations, affecting the safety constraints. With a smaller actuation, the system can move closer to the wall without violating the constraints. The test considers a different simulation time, allowing the system with less actuation to reach the goal. With more actuation, the constraints are more conservative, but the response is faster. The results from Table 6.13 confirm what was written, since the obstacles that previously led to an avoidance manoeuvre in 6.5 do not influence the generated trajectory. Only the simulation with $x < 0.15$ m does not have the same trajectory as the one without obstacles.

6.8 Summary

Through the multiple tests made, the algorithm successfully establishes safety for the generated trajectory. However, the safety constraints are overly conservative in many situations since the approach of

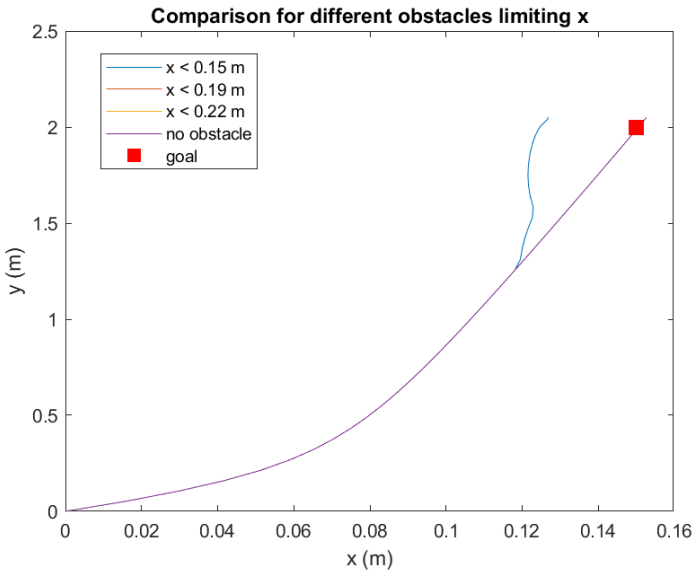


Figure 6.18: Representation of the state trajectory for the multiple simulations of different obstacle values

Table 6.13: Obstacle variation simulation results for less actuation

Simulation index	Time elapsed (s)	$e_{\mathbf{p}}$ (m)	$t(e_{\mathbf{p}})$
1	116.017	0.00184	40
2	121.0267	0.000755	40
3	127.694	0.000755	40
4	138.122	0.000755	40

state estimation in the MPC disturbances L, which, as mentioned several times, is computed from a state estimate set that considers the dynamics evolution for the whole set of inputs U. Throughout this chapter, many simulations are made to understand the implications of changing the algorithm's parameters. The safety conditions the algorithm imposes are, in certain cases, very restrictive and does not allow the system to reach its goal, which is undesirable. However, the robustness of the safety condition can serve as a condition of passive safety, preemptively avoiding states that, given extreme disturbances, lead to collision or unsafe states. During the tests, and in more extreme cases, the optimization problem was either unfeasible or the algorithm was able to generate a trajectory that respected the safety conditions. Some cases of unfeasibility consider large time step sizes and a large prediction horizon, which leads to the explosion of state estimate of MPC considering immediately every actuation as violating the safety constraints.

Chapter 7

Conclusion

The significant progress in autonomous systems in fields like aerospace, automotive, and robotics has highlighted the need for safe, reliable, and efficient control strategies. This thesis aimed to address this demand by developing a framework for generating passively safe trajectories using Model Predictive Control under uncertain conditions.

The work considered the unicycle model as the dynamics of the system to be controlled. Firstly, an approximate model was considered, which was used for an alternative contribution of this thesis: a minimal-conservativeness estimation algorithm of the progression of a unicycle's uncertain set for a given input. The unicycle model was properly linearized and discretized as a LTI model, which makes the approach considered in this thesis possible, unlike the previous approximate model in the set propagation algorithm.

The proposed method combines MPC with a state estimation approach that is critical in defining safety constraints. This state estimation, achieved through zonotopes, plays a vital role in accurately predicting the system's future states while accounting for control inputs and environmental uncertainties. An algorithm was developed to define estimates of the state by overapproximating model error and other uncertainties. The constraints derived from this state estimation consider the worst-case scenario effects of the uncertainty on the state's progression, which ensures safety. This integration of state estimation allows the MPC to define constraints that characterize the problem as inherently safe. Suppose the problem is feasible, meaning there is a possible trajectory the model can take to avoid danger. In that case, the solution, by construction, is also safe since the trajectory generation algorithm we developed defines the safety constraints, assuming an overapproximation of the uncertainty.

The results presented in Chapter 6 confirm the objective of the thesis of guaranteeing the safety of the generated trajectories. Furthermore, the robustness of the safety constraint helps the passive safety of the system by maintaining a conservative interval between the state and the obstacle. The results are satisfactory, albeit limited. Further testing could be made to better comprehend the solution performance.

This research contributes to advancing control strategies for autonomous systems, highlighting the balance between computational efficiency and reliability in dynamic and unpredictable environments.

7.1 Future Work

While the proposed framework demonstrates promising results, several areas remain for further development and optimization. Key directions for future work include:

- **Improvement of the used model:** The chosen model performs best for small variations around

a fixed operating point. Ideally, the possibility of using a more complex, more accurate model improves the overall accuracy of this work. This would require several adaptions to the algorithm that should reveal more positive results;

- **Obstacle avoidance:** The only obstacles considered in this thesis are in the form of a wall. Theoretically, the proposed method could be designed to consider other, more diverse obstacles, effectively validating further the performance of this framework;
- **Less Conservative State Estimation:** The state estimation processes defined in this work lead to considerably conservative estimates. Implementing a less conservative approach could significantly improve the results. Refining the CCG set propagation algorithm for real-time MPC applications is an option and other alternative set representations could be explored. Reducing conservativeness in the state estimation could provide more accurate constraints, minimizing the overapproximation and better optimizing the generated trajectories;
- **Computational optimization:** A recurrent challenge in MPC is the computational complexity of the strategy. Further research to reduce the computational load of the framework effectively improves real-time applicability.

Bibliography

[1] Findeisen, R., and Allgöwer, F., "An introduction to nonlinear model predictive control," *21st Benelux meeting on systems and control (Vol. 11, pp. 119-141)*, 2002.

[2] UNOOSA, *Annual number of objects launched into space – UNOOSA (with major processing by Our World in Data)*, 2024.

[3] Colvin, T. J., Karcz, J., and Wusk, G., "Cost and Benefit Analysis of Orbital Debris Remediation," tech. rep., NASA, 2023.

[4] "World health organization - who." `https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries`. Accessed: 2024-09-18.

[5] Botelho, A., Parreira, B., Rosa, P. N., and Lemos, J. M, *Predictive Control for Spacecraft Rendezvous*. Springer, 2021.

[6] Di Cairano, S., and Kolmanovsky, I. V., "Real-time optimization and model predictive control for aerospace and automotive applications," in *Annual American control conference (ACC). (pp. 2392-2409)IEEE.* , 2018.

[7] Di Cairano, S., Park, H., and Kolmanovsky, I., "Model predictive control approach for guidance of spacecraft rendezvous and proximity maneuvering," *International Journal of Robust and Nonlinear Control, 22(12), 1398-1427*, 2012.

[8] Hartley, E. N., Trodden, P. A., Richards, A. G., and Maciejowski, J. M. , "Model predictive control system design and implementation for spacecraft rendezvous," *Control Engineering Practice, 20(7), 695-713*, 2012.

[9] Frasch, J. V., Gray, A., Zanon, M., Ferreau, H. J., Sager, S., Borrelli, F., and Diehl, M., "An auto-generated nonlinear MPC algorithm for real-time obstacle avoidance of ground vehicles.," in *European Control Conference (ECC) (pp. 4136-4141). IEEE*, 2013.

[10] Klančar, G., and Škrjanc, I. , "Tracking-error model-based predictive control for mobile robots in real time.," *Robotics and autonomous systems, 55(6), 460-469*, 2007.

[11] Mayne, D., "Nonlinear Model Predictive Control: Challenges and Opportunities," *Nonlinear model predictive control, 23-44*, 2000.

[12] Mayne, D. Q., Seron, M. M., and Raković, S. V. , "Robust model predictive control of constrained linear systems with bounded disturbances," *Automatica, 41(2), 219-224*, 2005.

[13] Mayne, D. Q., Kerrigan, E. C., Van Wyk, E. J., and Falugi, P., "Tube-based robust nonlinear model predictive control," *International journal of robust and nonlinear control, 21(11), 1341-1353*, 2011.

[14] Mayne, D. Q., Kerrigan, E. C., and Falugi, P., “Robust model predictive control: advantages and disadvantages of tube-based methods,” *IFAC Proceedings Volumes, 44(1), 191-196*, 2011.

[15] Rubinstein, R. Y., and Kroese, D. P., *Simulation and the Monte Carlo method*. John Wiley & Sons, 2016.

[16] Williams, G., Goldfain, B., Drews, P., Saigol, K., Rehg, J. M., and Theodorou, E. A., “Robust Sampling Based Model Predictive Control with Sparse Objective Information,” *Robotics: Science and Systems (Vol. 14, p. 2018)*, 2018.

[17] Gandhi, M. S., Vlahov, B., Gibson, J., Williams, G., and Theodorou, E. A., “Robust model predictive path integral control: Analysis and performance guarantees,” *IEEE Robotics and Automation Letters, 6(2), 1423-1430*, 2021.

[18] Roque, P., Cortez, W. S., Lindemann, L., and Dimarogonas, D. V., “Corridor MPC: Towards optimal and safe trajectory tracking,” *2022 American Control Conference (ACC) (pp. 2025-2032). IEEE.*, 2022.

[19] Ames, A. D., Coogan, S., Egerstedt, M., Notomista, G., Sreenath, K., and Tabuada, P., “Control Barrier Functions: Theory and Applications,” *2019 18th European control conference (ECC)*, June 2019.

[20] de Groot, O., Brito, B., Ferranti, L., Gavrila, D., and Alonso-Mora, J., “Scenario-based trajectory optimization in uncertain dynamic environments,” *IEEE Robotics and Automation Letters, 6(3), 5389-5396*, 2021.

[21] Trym, T., Brekke, E. F., and Johansen, T. A., “On collision risk assessment for autonomous ships using scenario-based MPC,” *IFAC-PapersOnLine, 53(2), 14509-14516*, 2020.

[22] Althoff, D., Kuffner, J. J., Wollherr, D., and Buss, M., “Safety assessment of robot trajectories for navigation in uncertain and dynamic environments,” *Autonomous Robots, 32, 285-302*, 2012.

[23] Janson, L., Schmerling, E., and Pavone, M., *Monte Carlo motion planning for robot trajectory optimization under uncertainty*. Springer International Publishing, 2016.

[24] Eidehall, A., and Petersson, L., “Threat assessment for general road scenes using monte carlo sampling,” in *IEEE Intelligent Transportation Systems Conference (pp. 1173-1178). IEEE*, 2006.

[25] Eidehall, A., and Petersson, L., “Statistical threat assessment for general road scenes using Monte Carlo sampling,” *IEEE Transactions on intelligent transportation systems, 9(1), 137-147*, 2008.

[26] Jaulin, L., “Range-only slam with occupancy maps: A set-membership approach,” *IEEE Transactions on Robotics, 27(5), 1004-1010*, 2011.

[27] Marçal, J., Jouffroy, J., and Fossen, T. I., “An extended set-valued observer for position estimation using single range measurements,” *Proceedings International Symposium on Unmanned Untethered Submersible Technology*, 2005.

[28] Silvestre, D., Rosa, P., Hespanha, J. P., and Silvestre, C., “Stochastic and deterministic fault detection for randomized gossip algorithms,” *Automatica, 78, 46-60*, 2017.

[29] Scott, J. K., Raimondo, D. M., Marseglia, G. R., and Braatz, R. D., “Constrained zonotopes: A new tool for set-based estimation and fault detection,” *Automatica, 69, 126-136*, 2016.

[30] Daniel Silvestre , “Constrained Convex Generators: A Tool Suitable for Set-Based Estimation With Range and Bearing Measurements,” *IEEE Control Systems Letters, 6, 1610-1615*, 2022.

[31] Thabet, R. E. H., Raissi, T., Combastel, C., Efimov, D., and Zolghadri, A., “An effective method to interval observer design for time-varying systems,” *Automatica, 50(10), 2677-2684*, 2014.

[32] Kühn, W., “Rigorously computed orbits of dynamical systems without the wrapping effect,” *Computing, 61, 47-67*, 1998.

[33] C. Combastel, “A state bounding observer based on zonotopes,” in *2003 European control conference (ECC)*, pp. 2589–2594, IEEE, 2003.

[34] F. Chernousko, “Ellipsoidal state estimation for dynamical systems,” *Nonlinear Analysis: Theory, Methods & Applications*, vol. 63, no. 5-7, pp. 872–879, 2005.

[35] D. Silvestre, P. Rosa, J. Hespanha, and C. Silvestre, “Set-based fault detection and isolation for detectable linear parameter-varying systems,” *International Journal of Robust and Nonlinear Control*, vol. 27, no. 18, pp. 4381–4397, 2017.

[36] F. Abdallah, A. Gning, and P. Bonnifait, “Box particle filtering for nonlinear state estimation using interval analysis,” *Automatica*, vol. 44, no. 3, pp. 807–815, 2008.

[37] Alamo, T., Bravo, J. M., and Camacho, E. F. , “Guaranteed state estimation by zonotopes,” *Automatica, 41(6), 1035-1043*, 2005.

[38] A. Julius and G. Pappas, “Trajectory based verification using local finite-time invariance,” in *International Workshop on Hybrid Systems: Computation and Control*, pp. 223–236, Springer, 2009.

[39] B. Rego, D. Raimondo, and G. Raffo, “Set-based state estimation of nonlinear systems using constrained zonotopes and interval arithmetic,” in *2018 European Control Conference (ECC)*, pp. 1584–1589, IEEE, 2018.

[40] J. Wan, S. Sharma, and R. Sutton, “Guaranteed state estimation for nonlinear discrete-time systems via indirectly implemented polytopic set computation,” *IEEE Transactions on Automatic Control*, vol. 63, no. 12, pp. 4317–4322, 2018.

[41] A. S. Gadre and D. J. Stilwell, “A complete solution to underwater navigation in the presence of unknown currents based on range measurements from a single location,” in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1420–1425, IEEE, 2005.

[42] P.-M. Lee, B.-H. Jun, K. Kim, J. Lee, T. Aoki, and T. Hyakudome, “Simulation of an inertial acoustic navigation system with range aiding for an autonomous underwater vehicle,” *IEEE journal of oceanic engineering*, vol. 32, no. 2, pp. 327–345, 2007.

[43] P. Batista, C. Silvestre, and P. Oliveira, “Single range aided navigation and source localization: Observability and filter design,” *Systems & Control Letters*, vol. 60, no. 8, pp. 665–673, 2011.

[44] Scott, J. K., Findeisen, R., Braatz, R. D., and Raimondo, D. M., “Input design for guaranteed fault diagnosis using zonotopes,” *Automatica, 50(6), 1580-1589*, 2014.

[45] Althoff, M., *Reachability Analysis and its Application to the Safety Assessment of Autonomous Cars*. PhD thesis, Doctoral dissertation, Technische Universität München, 2010.