Constrained Predictive Control Strategies for Feedback-Linearized Autonomous Wheeled Vehicles

Cristian Tiriolo

A Thesis

in

The Department

of

Concordia Institute for Information and Systems Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy (Information and Systems Engineering) at

Concordia University

Montréal, Québec, Canada

May 2024

© Cristian Tiriolo, 2024

Concordia University

School of Graduate Studies

This is to certify that the thesis

Prepared by:	**Cristian Tiriolo**
Entitled:	**Constrained Predictive Control Strategies for Feedback-Linearized Autonomous Wheeled Vehicles**

and submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy (Information and Systems Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

Dr. Rabindranath Raut	Chair
Dr. Sidney Givigi	External Examiner
Dr. Shahin Hashtrudi Zad	Arm-Length Examiner
Dr. Jun Yan	Examiner
Dr. Rastko Selmic	Examiner
Dr. Walter Lucia	Supervisor

Approved by:

Dr. Jun Yan

Graduate Program Director

June 18th, 2024

Dr. Mourad Debbabi, Dean

Gina Cody School of Engineering and Computer Science

Abstract
Constrained Predictive Control Strategies for Feedback-Linearized Autonomous Wheeled Vehicles

Cristian Tiriolo, Ph.D.
Concordia University, 2024

Autonomous vehicles are becoming increasingly widespread in various real-world applications, ranging from manufacturing and transportation to search and rescue operations. To perform these tasks effectively, it is crucial for the vehicle to be capable of solving trajectory tracking, path following, and obstacle avoidance problems. To improve the accuracy of the performed trajectory, the input constraints acting on the robot's model should be directly included in the control design. Unfortunately, many of the available control algorithms are unable to do so.

In the last two decades of research, Model Predictive Control solutions have been developed to solve the considered control problems for autonomous wheeled vehicles. Nonlinear MPC schemes exploit accurate state predictions, however, the underlying computational demand might not be affordable in strict real-time contexts or when the robot's computation capabilities are limited. Conversely, linearized MPC approaches, have the important advantage of drastically reducing computational burdens at the expense of more conservative control performance.

This research proposes a novel control paradigm to solve trajectory tracking, path following and obstacle avoidance problems for input-constrained wheeled mobile vehicles. The proposed solutions are applicable to both differential-drive and car-like robots, and they are the result of the combination of Model Predictive Control strategies and feedback linearization techniques. First, it is shown that if a feedback linearized model of the robot is exploited for predictions in MPC, then the set of admissible inputs for the linearized model is a nonconvex and state-dependant polyhedron, leading to non-convex and computationally expensive optimization problems with local minima issues. Then, a novel worst-case circular approximation of the state-dependent input constraints set is analytically derived and used to design reference tracking controllers that are, by design, recursively feasible and non-conservative.

The proposed predictive control paradigm has been successfully applied in real-time to solve trajectory tracking, obstacle avoidance, and formation control problems for mobile robots and autonomous cars. The effectiveness and benefits of the proposed control framework are shown with simulations and laboratory experiments involving the Khepera IV differential-drive robots and Quanser Qcar, and its performance contrasted with state-of-the-art alternative control solutions.

Acknowledgments

Completing this doctoral thesis has been a journey filled with challenges, growth, and invaluable support from various individuals and institutions. I extend my deepest gratitude to:

My supervisor, Dr. Walter Lucia, whose guidance, encouragement, and expertise have been instrumental throughout this research endeavor. Your insightful feedback and unwavering support have shaped this thesis in profound ways.

The members of my doctoral committee, Dr. Sidney Givivi, Dr. Shahin Hashtrudi-Zad, Dr. Rabindranath Raut, Dr. Rastko Selmic, and Dr. Jun Yan, for their constructive criticism, scholarly insights, and invaluable suggestions that have greatly enriched the quality of this work.

The faculty and staff of Concordia University, for providing an enriching academic environment and resources essential for carrying out this research.

My family, whose unwavering love, encouragement, and understanding have sustained me through the highs and lows of this journey. Your belief in me has been my greatest motivation.

My friends and colleagues, Elisa, Eleonora, Alessandro, Marco, Mirko, Melo, Pietro, Geovana, and Surya, for their support, encouragement, and occasional distractions that provided much-needed relief during moments of academic intensity.

Finally, I extend my heartfelt gratitude to all the participants and individuals who generously contributed their time, insights, and expertise to this study.

This thesis would not have been possible without the collective support and encouragement of these individuals and institutions. Thank you for being part of this significant milestone in my academic journey.

List of Publications[1]

[J1] C. Tiriolo and W. Lucia, ”A Feedback Linearized Model Predictive Control Strategy for Input-Constrained Self-Driving Cars” submitted to IEEE Transactions on Control Systems Technology (TCST) (under review[2]).

[C1] S.Rajkumar†, C. Tiriolo†, and W. Lucia, ”Collision-Free Platooning of Mobile Robots through a Set-Theoretic Predictive Control Approach” accepted for presentation to IEEE American Control Conference (ACC) 2024

[J2] C. Tiriolo and W. Lucia, ”A Set-Theoretic Control Approach to the Trajectory Tracking Problem for Input-Output Linearized Wheeled Mobile Robots” IEEE Control Systems Letters, vol. 7, pp. 23472352, 2023, presented to Conference on Decision and Control (CDC) 2023.

[J3] C. Tiriolo and W. Lucia, ”On the Design of Control Invariant Regions for Feedback Linearized Car-Like Vehicles”, IEEE Control System Letters vol. 7, pp. 739–744, 2022, presented to IEEE American Control Conference (ACC) 2023.

[C2] C. Tiriolo, G. Franzè, and W. Lucia, ”An Obstacle-Avoidance Receding Horizon Control Scheme for Constrained Differential-Drive Robot via Dynamic Feedback Linearization” in 2023 American Control Conference (ACC), pp. 11161121, IEEE, 2023.

[J4] C. Tiriolo, G. Franzè, and W. Lucia, ”A Receding Horizon Trajectory Tracking Strategy for Input-Constrained Differential-Drive Robots via Feedback-Linearization”, Transactions on Control Systems Technology (TCST), vol. 31, no. 3, pp. 1460–1467, 2023

[1]The publications are ordered chronologically. [J] denotes journal articles, [C] conference papers
[2]Preprint available at `https://arxiv.org/abs/2405.01753`

Contribution of Authors

This thesis is a compilation of manuscripts that have been published and/or submitted for publication. The work presented in these manuscripts is a collaborative effort involving multiple contributors. Below, we outline the specific contributions of each author to the research and writing of these manuscripts.

Manuscript 1: A Feedback Linearized Model Predictive Control Strategy for Input-Constrained Self-Driving Cars

- **Cristian Tiriolo:** Conceptualized and designed the study, developed the predictive paradigm combining Model Predictive Control and Feedback Linearization, conducted the simulations and experiments, analyzed the data, and wrote the manuscript.
- **Dr. Walter Lucia:** Supervised the research project, provided critical feedback and guidance throughout the study, and contributed to the manuscript revisions.

Manuscript 2: Collision-Free Platooning of Mobile Robots through a Set-Theoretic Predictive Control Approach

- **Cristian Tiriolo:** Conceptualized and designed the study, developed the control algorithm, conducted the simulations and experiments, analyzed the data, and wrote the manuscript.
- **Suryaprakash Rajkumar:** Conceptualized and designed the study, developed the control algorithm, conducted the simulations and experiments, analyzed the data, and wrote the manuscript.
- **Dr. Walter Lucia:** Supervised the research project, provided critical feedback and guidance throughout the study, and contributed to the manuscript revisions.

Manuscript 3: A Set-Theoretic Control Approach to the Trajectory Tracking Problem for Input-Output Linearized Wheeled Mobile Robots

- **Cristian Tiriolo:** Conceptualized and designed the study, developed the control algorithm, conducted the simulations and experiments, analyzed the data, and wrote the manuscript.
- **Dr. Walter Lucia:** Supervised the research project, provided critical feedback and guidance throughout the study, and contributed to the manuscript revisions.

Manuscript 4: On the Design of Control Invariant Regions for Feedback Linearized Car-Like Vehicles

- **Cristian Tiriolo:** Conceptualized and designed the study, conducted the simulations, analyzed the data, and wrote the manuscript.

- **Dr. Walter Lucia:** Supervised the research project, provided critical feedback and guidance throughout the study, and contributed to the manuscript revisions.

Manuscript 5: An Obstacle-Avoidance Receding Horizon Control Scheme for Constrained Differential-Drive Robot via Dynamic Feedback Linearization

- **Cristian Tiriolo:** Conceptualized and designed the study, conducted the simulations, analyzed the data, and wrote the manuscript.

- **Dr. Giuseppe Franzè:** Provided guidance on the Feedback Linearization technique, assisted with conceptualization and design of the study, and reviewed and edited the manuscript.

- **Dr. Walter Lucia:** Supervised the research project, provided critical feedback and guidance throughout the study, and contributed to the manuscript revisions.

Manuscript 6: A Receding Horizon Trajectory Tracking Strategy for Input-Constrained Differential-Drive Robots via Feedback-Linearization

- **Cristian Tiriolo:** Conceptualized and designed the study,developed the control algorithm, conducted the experiments, analyzed the data, and wrote the manuscript.

- **Dr. Giuseppe Franzè:** Provided guidance on the Feedback Linearization technique, assisted with conceptualization and design of the study, and reviewed and edited the manuscript.

- **Dr. Walter Lucia:** Supervised the research project, provided critical feedback and guidance throughout the study, and contributed to the manuscript revisions.

Contents

List of Figures

List of Tables

List of Symbols

Symbol	Description
$r(t)$	Reference trajectory Cartesian coordinates.
$\xi(t)$	Cartesian coordinates tracking error.
$\mathcal{U}_d$	Set of admissible control inputs for the differential-drive model.
$z(t)$	Feedback-linearized system's state
$u(t)$	Control inputs
$\mathcal{U}_u$	Unicycle's input constraint set
$\mathcal{U}(\theta)$	Feedback-linearized system's input constraints set
$\underline{\mathcal{U}}$	Worst-case time-invariant input constraint set.
$\mathcal{B}(d_r)$	Ball set defined by the vision radius d_r.
Q_x	Weight matrix for the state in the cost function.
Q_u	Weight matrix for the control input in the cost function.
N_p	Prediction horizon in the receding horizon control problem.
T_s	Sampling time.
$r^z(k)$	Reference signal for the feedback-linearized model.
γ	Upper bound on the cost function in the optimization problem.
$\mathcal{E}_0$	Worst-case ellipsoidal domain of attraction
Y_k	Control gain matrix at time step k.
$H(\theta)$	Matrix defining the state-dependent input constraints.
$\mathcal{E}_k$	Ellipsoidal domain of attraction at time step k.
$x(k+1)$	Discrete-time state equation for the vehicle's position along the x axis.
$y(k+1)$	Discrete-time state equation for the vehicle's position along the y axis.
$\theta(k+1)$	Discrete-time state equation for the vehicle's orientation
$\varphi(k+1)$	Discrete-time state equation for the vehicle's steering angle
$\omega_R^i(k)$	Right wheel angular speed of the i-th robot.
$\omega_L^i(k)$	Left wheel angular speed of the i-th robot.
$v^i(k)$	Linear speed of the i-th robot.
$\omega^i(k)$	Angular speed of the i-th robot.
$q_r^i(k)$	Reference pose for the i-th robot.
$q^0(k)$	Leader robot's state vector.
$\eta^i(k)$	Inter-agent delay for the i-th follower robot.

Continued on next page

continued from previous page

Symbol	**Description**
$\mathcal{R}^i(z^i(k))$	One-step reachable set of the i-th robot.
Q_R	Shape matrix for the reachable set.
$\tilde{r}_u$	Radius of the outer approximation of the input constraint set.
$\overline{d}$	Safe distance to avoid collisions.
$\overline{\Omega}$	Maximum angular speed of the wheels' motors.
$\mathcal{B}(\overline{d}, z^i(k))$	Ball set centered at $z^i(k)$ with radius $\overline{d}$.
r_d^0	Radius of the disturbance set for the leader.
r_u^0	Radius of the worst-case input constraint set for the leader.
N	Number of robots in the formation or prediction horizon.
A_c	Continuous-time state matrix for the linear model.
B_c	Continuous-time input matrix for the linear model.
A	Discrete-time state matrix for the linear model.
B	Discrete-time input matrix for the linear model.
$v(k)$	Linear velocity at time k
$a(k)$	Acceleration at time k
$\eta(k)$	affine system's traslational term
$\hat{u}(k)$	Translated input vector for the affine linearized model.
$\zeta(k)$	Affine term in the norm-bounded uncertainty model.
$\rho(k)$	Structured norm-bounded perturbation in the norm-bounded uncertainty model.
Δ	Matrix describing the structured uncertainty.
$\mathcal{B}^{\xi}(0_4, \bar{d})$	Augmented state constraint mapping in the linearized model.
$p(k)$	Planar position of the vehicle
p_f	Desired target location.
$\mathcal{O}_f(k)$	Obstacle-free region detected by an on-board perception module.
$\mathcal{O}(k)$	Set of detected obstacles.
p_i	Sequence of viable waypoints generated by the path planner.
ε	Tolerance parameter for the safe distance.
$Q_{\mathcal{B}_{\bar{d}}}$	Shaping matrix for the obstacle-free ball.
$\bar{z}_i$	Equilibrium state of the affine model associated with waypoint p_i.
$\xi_i(k)$	Error state defined as $\bar{z}_i - z(k)$.
H_u	Matrix defining the input constraints for the unicycle model.

Continued on next page

continued from previous page

Symbol	**Description**
H_{T_s}	Transformation matrix for the discretization process.
$T_{FL}(\theta, v)$	Transformation matrix for feedback linearization.
$H(\theta, v)$	Shaping matrix for the input constraints after feedback linearization.
$\hat{\mathcal{U}}(\theta, v)$	Transformed input constraint set for the affine linearized model.
γ_k	Upper bound on the cost function in the LMI optimization problem at time step k.
Q_k	State weighting matrix in the LMI optimization problem at time step k.
Y_k	Input weighting matrix in the LMI optimization problem at time step k.
Λ	Matrix related to the structured uncertainty in the LMI optimization problem.
$\mathcal{O}_f^z(k)$	Obstacle-free region in the augmented state space.
$J_\infty(k)$	Linear-Quadratic (LQ) cost function in the optimization problem.
$\bar{Q}_k$	Positive definite matrix used in the LMI optimization problem.
$v_r(t)$	Reference linear velocity for a car-like vehicle.
$\omega_r(t)$	Reference steering angular velocity for a car-like vehicle.
$\theta_r(t)$	Reference orientation for a car-like vehicle.
$\varphi_r(t)$	Reference steering angle for a car-like vehicle.
$\tilde{q}(t)$	Vehicle's tracking error
$w(t)$	Input for the feedback linearized model.
$z_r(t)$	Reference trajectory in the feedback linearized system.
$\tilde{z}(t)$	Tracking error in the feedback linearized system.
$\mathcal{U}(\eta)$	State-dependent input constraint set for feedback linearized car-like vehicle.
$L(\eta)$	Shaping matrix for the polyhedral input constraint set of the car's feedback-linearized model.
$\hat{\mathcal{U}}$	Worst-case circular inner approximation of the input constraint set.
$\hat{r}$	Radius of the worst-case circular inner approximation.
$\tilde{\eta}(t)$	Internal state tracking error.

Continued on next page

continued from previous page

Symbol	**Description**
$O(\eta)$	Matrix for the internal dynamics of the car-like vehicle.
T_{FL}	Transformation matrix for input-output linearization.
S	Shaping matrix for the control invariant region.
$\mathcal{Z}$	Control invariant region.
K	Control gain matrix.
λ	Closed-loop eigenvalues of the system.
$\mathcal{W}(r_d)$	Circular set over-approximating the bounded disturbance for the reference trajectory.
W_{r_d}	Shaping matrix for the disturbance set $\mathcal{W}(r_d)$.
ξ	Parameter related to the matrix inequality for robust invariance.
G	Matrix such that $G^T S^{-1} G = I_{2\times 2}$.
Ψ	State transition matrix in the discrete-time feedback linearized model.
Θ	Control input matrix in the discrete-time feedback linearized model.
H	Hessian matrix in the quadratic programming problem.
p	Linear term in the quadratic programming problem.
$\hat{L}(\eta)$	Augmented shaping matrix for the input constraints.
$\hat{Q}_u$	Augmented quadratic term for the input constraints.
Ψ_N	State transition matrix at the prediction horizon N.
Θ_N	Control input matrix at the prediction horizon N.
$\hat{Q}$	Augmented state weighting matrix.
$\hat{R}$	Augmented input weighting matrix.
$\hat{P}_w$	Polyhedral approximation of the input constraints.
$\hat{p}_w$	Right-hand side vector for the polyhedral input constraints.
$\hat{p}_N$	Polyhedral approximation of the terminal set constraints.

List of Acronyms

APF Artificial Potential Field.

FL-MPC Feedback-Linearized Model Predictive Control.

IPS Indoor Positioning System.

LMI Linear Matrix Inequality.

LQ Linear-Quadratic.

MPC Model Predictive Control.

QCQP Quadratically Constrained Quadratic Programming.

QP Quadratic Programming.

RHC Receding Horizon Control.

RPI Robust Positive Invariant.

SQP Sequential Quadratic Programming.

ST-MPC Set-Theoretic Model Predictive Control.

ST-RHC Set-Theoretic Receding Horizon Control.

UKF Unscented Kalman Filter.

WMR Wheeled Mobile Robots.

1 Introduction

1.1 Motivation

The field of mobile robotics and autonomous vehicles has witnessed remarkable growth and innovation in recent years, driven by advances in computing power, sensor technology, and artificial intelligence. This progression has led to the widespread adoption of robotic systems and self-driving cars across various industries, ranging from manufacturing and logistics to transportation and urban planning [1].

Wheeled mobile robots and self-driving cars represent two prominent examples of autonomous systems that have gained significant attention due to their potential to revolutionize transportation and mobility [2]. Wheeled mobile robots, ranging from small indoor robots to large outdoor platforms, are employed in tasks such

- environmental surveillance [3] (Fig. 1.1-(a));
- military applications [4, 5](Fig. 1.1-(b));
- agriculture [6, 7](Fig. 1.1-(c));
- warehouse automation [8];
- disinfection of contaminated environments [9](Fig. 1.1-(d));
- search and rescue operations [10].

On the other hand, self-driving cars, also known as autonomous vehicles, aim to transform personal and public transportation by enabling vehicles to navigate and operate on roads without human intervention.

Mobile robots and autonomous vehicles are systems generally designed to autonomously operate in known, partially or totally unknown, and unpredictable scenarios [2, 11]. Among the available robot architectures, Wheeled Mobile Robots (WMRs) have attracted a major interest [6] thanks to their relatively low cost, small dimensions, and simple kinematic structure. Three well-known types of WMRs are:

- unicycle, a robot with a single driving wheel;
- differential-drive, consisting of two independently driven wheels;
- and the car-like robot, having the same kinematics as an automobile.

https://smprobotics.com/
(a)
archive of VOP CZ s.p.; author: Radim Horák)
(b)
https://www.nytimes.com/2020/02/13/science/farm-agriculture-robots.html
(c)
https://www.automationworld.com/home/article/21126536
(d)

Figure 1.1: Common applications of Wheeled Mobile Robots

These systems are typical examples of nonholonomic mechanisms i.e., any lateral displacement of the wheels is not allowed by the kinematic model of the robot.

In general, regardless of the specific application, WMRs and autonomous vehicles must be capable of addressing the following control problems [2, 12–14]:

- Localization, i.e, the problem of estimating the robot's position and orientation within an environment.
- Mapping, i.e. building an accurate representation of the environment.
- Trajectory and/or path planning, i.e. the problem of computing an admissible path or trajectory from a start to a final location. Although the terms path and trajectory are often used synonymously, the first is a sequence of points in the workspace, while the latter defines a timing law associated with each point of the trajectory.
- Trajectory and/or path tracking, i.e., the problem of accurately following the planned path or trajectory.
- Obstacle avoidance, i.e., computing a control law that drives the robot toward the goal by avoiding any obstacle present in the environment.
- Formation control, i.e. cooperative control for a fleet of robots or vehicles aimed at following a predefined trajectory while maintaining a desired spatial pattern [15]

1.2 Literature Review

Of particular interest for this thesis are the above control problems when differential-drive, unicycle, and car-like robots are considered. Specifically, from a control perspective, the proposed work addresses four main research topics:

1. trajectory tracking and path following for mobile robots.
2. collision-avoidance control for mobile robots.

3. platooning formation control for mobile robots

4. trajectory tracking for car-like vehicles.

In the following, a review of the existing literature for the above-mentionedcontrol problems is proposed.

1.2.1 Trajectory Tracking and Path Following for mobile robots and car-like vehicles

In most of the application domains, the vehicle's control system should be able to solve either a path-following problem or a trajectory-tracking problem. Although the two problems are similar, the first requires a sequence of points that the robot tracks sequentially, while the second presupposes the knowledge of the trajectory timing law [16]. The main difficulty in accomplishing such tasks relies on the presence of nonholonomic constraints, i.e., the wheels cannot perform any lateral displacements [17]. Furthermore, due to the Brockett conditions, posture stabilization of such systems cannot be achieved by a static state-feedback law [18, 19]. Extensive research has been conducted on the trajectory tracking problem for wheeled mobile robots [12] and most of the proposed solutions range from Lyapunov-based design, [20–22], adaptive laws [23], and feedback linearization-based schemes [19, 24].

In more recent years, thanks to the evolution of processing units and numerical algorithms to address optimization problems, several Model Predictive Control (MPC) solutions have been developed, see [18, 25–29] and references therein. Nonlinear MPC schemes exploit accurate state predictions, however recursive feasibility and stability properties are usually difficult to prove [27, 30]. Besides, the underlying computational demand might not be affordable in strict real-time contexts or when the robot's computation capabilities are limited [28, 31]. As the latter issue is concerned, an alternative solution, based on a neural network, has been presented in [29] in order to obtain a computationally affordable algorithm. Conversely, linearized MPC approaches, e.g., [18, 25, 32], have the important advantage of drastically reducing computational burdens at the expense of more conservative control performance.

1.2.2 Collision-avoidance Control for Mobile Robots

When autonomous robots are required to operate in unknown obstacle environments, tasks such as mission control, communication management, navigation, and motion control must be adequately performed [33]. In [34], a deep analysis of navigation techniques for mobile robots moving in obstacle-populated environments is provided.

Particular attention has been devoted to those solutions leveraging the concepts of Artificial Potential Field (APF), machine learning, and Model Predictive Control. Strategies based on APF typically do not require the introduction of a path planner and address the navigation problem by means of the so-called reactive approaches. Anyway, such solutions might suffer from local minima, sub-optimality of the solution, and the difficulty of ensuring obstacle avoidance capabilities when constrained vehicles are considered [35]. Recently, machine learning-based solutions have been gaining increasing attention for their potential use in heterogeneous real-world obstacle environments. Although interesting, the resulting algorithms suffer from their black-box nature and, as a consequence, it is difficult to provide theoretical guarantees on the absence of collisions [36].

The intrinsic capability of MPC to simultaneously and formally deal with stability requirements, disturbances, and constraints, makes it an appealing solution to the control of mobile robots in obstacle scenarios. Different MPC-based approaches have been proposed in order to address the navigation problem of mobile ground robots [34, 37], where the absence of collisions is embedded into the MPC optimization.

Both nonlinear [38–42] and linear [43–45] MPC formulations are available for collision avoidance purposes. However, such solutions suffer from the same drawbacks highlighted above.

1.2.3 Tracking control of platoons of wheeled mobile robots

Platoon in vehicular technologies is often defined as a group of vehicles traversing in a coordinated manner while communicating with each other and by using autonomous driving technology [46].

In the literature, different solutions have been proposed to control platoons of mobile robots (see [47] and references therein). In [48], the problem is solved in a centralized fashion via nonlinear controllers, while in [49] a collision-free solution is developed; on the other hand, the authors of [50] propose a decentralized constant-time approach where an inter-vehicle delay policy is designed to track a given reference trajectory while maintaining a platoon structure. Although appealing, the above-mentioned control solutions share as a common drawback the incapability to guarantee input constraint fulfillment, i.e., physical limitations on the control signals of the vehicles are not directly considered in the control design. Unfortunately, failing to address input constraints may lead to undesired saturation phenomena, loss of tracking performance, and in the worst case collisions between agents. Model Predictive Control has been successfully applied also to the control of vehicular platoons thanks to its peculiar capability to incorporate stability requirements, tracking performance, and state and input constraints directly in the control design.

In the last decade, nonlinear MPC has been investigated to solve platooning control problems for autonomous vehicles, (see [51] and references therein), however, its high computational burdens represent a major obstacle for real-time implementations. Furthermore, nonlinear formulations are in general nonconvex and they suffer from local minima problems [52]. Conversely, simpler linear MPC formulations are more computationally affordable, and preferred for real-time applications. In [53], the linear dual-mode Set-Theoretic MPC (ST-MPC), first developed in [54] to stabilize a general linear system subject to bounded disturbance, has been applied to the considered platooning control problem for mobile robots. Such a control architecture combines the concept of control invariance and N-step controllability, to formulate a convex and recursively feasible optimization. However, the strategy in [53] deals with agents described by linear time-invariant models. The latter may constitute a restrictive assumption especially when time-varying reference trajectories are considered.

1.2.4 Tracking Control of Self-Driving Cars

Central to the autonomous vehicle's operational integrity is trajectory tracking control, a critical aspect for ensuring the safety, reliability, and comfort of these vehicles [13, 14]. The term reference trajectory refers to a sequence of consecutive waypoints, with associated spatial and temporal information, that the vehicle is required to accurately track [55]. Achieving reliable trajectory tracking enables autonomous vehicles to navigate complex environments with precision, adaptability, and safety, thus accelerating the integration of autonomous technology into everyday transportation systems [12].

Extensive research has been conducted to develop control strategies to solve the trajectory tracking problem for autonomous cars [56], ranging from simple non-model based solutions like the well-established PID controllers [57, 58], to more sophisticated nonlinear control solutions like sliding-mode controllers [59,60], and adaptive backstepping control [61]. In the last decade, deep learning-based approaches have been applied to the control of autonomous vehicles due to their ability to self-optimize their behavior from data and adapt to complex and dynamic environments. [62,63] offer an exhaustive review of the most recent developments in the application of machine learning techniques to autonomous vehicle control. Despite their relatively high tracking performance, the biggest challenge pertaining to this class of algorithms remains their dependence on large, annotated datasets for training, which can be expensive and time-consuming to collect and maintain. Additionally, machine learning models often act as "black boxes," offering limited interpretability regarding how decisions are made, which raises concerns about accountability and safety in critical applications like autonomous driving. One common drawback of the above-discussed tracking strategies is their incapability to address input constraints, i.e., physical limitations of the computed control signal, which may lead to a lack of close-loop stability guarantees [64].

1.2.5 Model Predictive Control and Feedback Linearization

Model Predictive Control has emerged as a premier control strategy in this domain, owing to its ability to anticipate the future behavior of the vehicle and handle multiple constraints simultaneously [14, 65]. In recent years, the application of MPC in autonomous vehicles has been extensively studied, highlighting its potential in managing the complex dynamics and uncertainties inherent to vehicular control [14]. Notable works [66, 67] have underscored the efficacy of MPC in navigating autonomous vehicles through dynamic environments. Despite these advancements, the deployment of MPC in real-world scenarios faces significant difficulties. Nonlinear MPC approaches [68, 69], while robust, are computationally demanding, posing challenges to their real-time implementation [70, 71]. Moreover, given the nonconvex nature of the MPC optimization, the solver algorithms may be characterized by uncertain convergence and suboptimality [70, 72]. Conversely, linear MPC techniques [71, 73, 74] offer computational efficiency but at the expense of model fidelity. As a matter of fact, solutions that exploit error dynamics linearized around the reference trajectory have been found to be suboptimal [75].

Feedback Linearization (FL) is a popular technique used to recast nonlinear dynamical models into equivalent linear descriptions by means of a proper change of variables and control input vector [76]. In [77,78], the FL has been combined with MPC formulations. There, the authors have used FL to obtain exact linear state predictions for an unconstrained vehicle model. Along similar lines is the approach presented in [79], where an unconstrained FL-MPC scheme is derived to address the trajectory tracking problem for a wheeled inverted pendulum. In [80], the nonlinear model of an electric power wheelchair is linearized via transverse feedback linearization. Moreover, the input constraints, arising from nonlinear mapping operations between the vehicle and the linear model, are exploited for input prediction purposes. Consequently, the resulting MPC optimization problem is nonlinear and non-convex. This literature analysis puts to light that the combined use of FL arguments and MPC philosophy leads to computationally affordable convex optimizations only for unconstrained models. On the other hand, input constraints are always converted into nonlinear and state-dependent constraints [81], leading to nonlinear MPC formulations. As a matter of fact, it has been shown that non-convex MPC formulations can be avoided if sequential approximations of the constraint set are performed along the prediction horizon.

Unfortunately, this strategy turns out to be very conservative even for small values of the prediction horizon, see, e.g. [81–83] .

1.3 Contribution of the Thesis

In the last decades, the nonlinear Model Predictive Control scheme outlined in Chapter 2 has been successfully applied to solve trajectory tracking, motion planning, and path-following problems for wheeled mobile robots and autonomous vehicles. However, despite the progress made in these areas, there remains a pressing need for further research, particularly with regard to the local minima problems and high computational complexity that such algorithms suffer from when a nonlinear model is exploited for control design [72].

In light of this need, this thesis presents a novel control paradigm, based on the combination of predictive control and feedback linearization arguments, for solving trajectory-tracking and obstacle-avoidance and formation control problems for autonomous wheeled vehicles, with the ultimate aim of providing a low-computational demanding but highly efficient control solution to the considered problems. To the best of the author's knowledge, it represents the first attempt to design guaranteed predictive control strategies for input-constrained feedback linearized autonomous wheeled vehicles. The main contributions of the thesis can be summarized as follows:

1. Formal characterization of the time-varying and state-dependent input constraint sets to which the feedback-linearized model is consequently subject and analytical characterization of worst-case inner approximation of such time-varying sets.

2. Handling of the nonlinear internal dynamics arising when the vehicle model is approached by partial-state input-output linearization techniques.

3. Formulation of real-time affordable quadratic optimizations that don't introduce any approximations in the model predictions.

4. Guarantee of the existence of a control law that solves the considered problem when time-varying constraints act on the linearized model. The proposed control solutions enjoy the recursive feasibility property, i.e., under mild assumptions, the proposed MPC optimization problems are always feasible.

5. Real-time applicability of the proposed control paradigm. The developed control solutions have been extensively validated via laboratory real-time experiments, involving Khepera IV mobile robots and Quanser Qcars.

1.4 Structure of the Thesis

This thesis follows a manuscript-based format[3] and it is organized as follows.

- Chapter 2 contains all the background material related to the proposed research. Specifically, first, the kinematic modeling of wheeled autonomous vehicles is discussed, then the theory pertaining to feedback linearization is proposed, with particular emphasis on the application of such a technique to the linearization of robots' kinematics. Finally, the theoretical material related to predictive control is presented.

- Chapters 3-4 proposes the work published in [84] ([J4]) and [85] ([J2]), respectively, which address the trajectory tracking control problem for wheeled mobile robots. Specifically, Chapter 3 proposes a receding horizon control solution based on feedback-linearization arguments and linear matrix inequalities. On the other hand, in Chapter 4, such a control solution is extended by means of set-theoretic MPC arguments to improve tracking performance. Among the main contributions, these two works represent the first attempt at the analytical characterization of the time-varying admissible input constraint set to which the linearized model is subject.

- In Chapter 5, the set-theoretic MPC proposed in Chapter 4 is extended to solve the trajectory tracking problem for a platoon of wheeled mobile robots while guaranteeing absence of collisions between agents. The results are published in [86]([C1]).

- Chapter 6, whose results are published in [87] [C2], addresses the obstacle avoidance problem for differential-drive robots via receding horizon control based on full-state dynamic feedback linearization.

- Chapters 7-8 present the work published in [88] ([J3]) and [J1], respectively, which address the trajectory tracking problem for self-driving cars using feedback linearization arguments. Specifically, in Chapter 7 the input-output linearized model of the car kinematics is discussed and its time-varying input constraint set is formally characterized. Moreover, the chapter proposes a novel analytical design of control invariant regions that makes use of the worst-case inner approximation of the input constraint set, which is also analytically characterized. Such a control design is exploited in Chapter 8 to design a novel MPC framework based on feedback linearization capable of ensuring bounded tracking error, guaranteed feasibility, and input constraint fulfillment for feedback-linearized car-like vehicles.

[3] `https://www.concordia.ca/content/dam/sgs/docs/handbooks/thesispreparationguide.pdf`

- Chapter 9 concludes this thesis discussing potential research implications.

2 Background

In this chapter, the fundamental theoretical concepts pertaining to the proposed research are presented. Specifically, in the following, first, an overview of the kinematic model of some wheeled mobile robots is given. Then, different feedback linearization techniques are outlined to derive exact linear models describing the kinematics of the considered wheeled vehicles. Finally, the chapter concludes with a brief overview of Model Predictive Control for the stabilization of discrete-time dynamic systems.

2.1 Wheeled Vehicles Models

The kinematics of a generic nonholonomic Wheeled Mobile Robot (WMR) can be modeled as:

$$\dot{q} = G(q)w, \quad G \in \mathbb{R}^{n \times m} \tag{2.1.1}$$

where $q \in \mathbb{R}^n$ is a vector of generalized coordinates, and $w \in \mathbb{R}^m$ the vector of control inputs [89]. The above nonlinear system describes a general driftless WMR and the matrix G can assume different structures depending on the considered vehicle.

Such a system is nonholonomic since it is characterized by the presence of $n - m$ non-integrable differential constraints in the form:

$$A(q)\dot{q} = 0 \tag{2.1.2}$$

This is due to the fact that the wheels of the vehicle cannot slip but can perform only pure rolling motion. Consequently, the nonlinear kinematics (2.1.1) do not satisfy the so-called Brockett necessary conditions for the smooth stabilizability [90]. One major negative consequence is established by the following proposition.

Proposition 2.1. *Since the number of inputs m is different from the number of states n, asymptotic stability cannot be achieved by static control laws, and a time-varying feedback control is instead required. [89]*

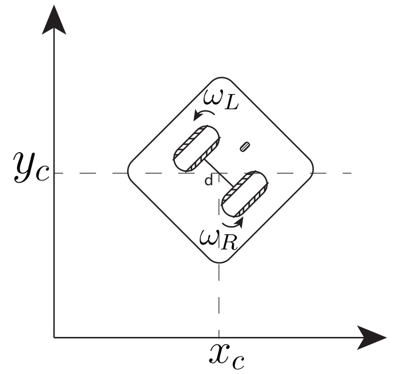
In the following three different examples of nonholonomic WMR are analyzed: differential-drive, unicycle, and car-like robots.

2.1.1 Differential-Drive Robot model

A typical example of WMR is the differential-drive robot, consisting of two independent drive wheels (see. Fig. 2.1). By assuming that the robot moves at relatively low speed, its kinematic is described by

$$\begin{array}{rcl} \dot{x}(t) & = & \frac{r}{2}(\omega_R(t)+\omega_L(t))\cos(\theta(t)) \\ \dot{y}(t) & = & \frac{r}{2}(\omega_R(t)+\omega_L(t))\sin(\theta(t)) \\ \dot{\theta}(t) & = & \frac{r}{d}(\omega_R(t)-\omega_L(t)) \end{array} \tag{2.1.3}$$

where ω_R, ω_L are the control inputs, right and left wheels' angular velocities, respectively,



(a)

Figure 2.1: Differential-drive vehicle

while $q(t) = [x(t), y(t), \theta(t)]^T$ is the state vector of the system, consisting of the robot's center of mass position in the plane, $x(t)$ and $y(t)$, and its orientation $\theta(t)$. The nonholonomic constraint (2.1.2) for such robot is given by:

$$\dot{x}\sin(\theta) = \dot{y}\cos(\theta) \tag{2.1.4}$$

Moreover, the robot's model is subject to input saturation constraints on the wheel angular velocities, i.e.

$$|\omega_R| \leq \overline{\Omega}, \quad |\omega_L| \leq \overline{\Omega} \tag{2.1.5}$$

which can be rewritten in a compact form as:

$$\mathcal{U}_d = \{[\omega_R, \omega_L]^T \in \mathbb{R}^2 : \, H_d \, [\omega_R, \omega_L]^T \leq 1\}, \quad \text{where } H_d = \begin{bmatrix} -\frac{1}{\overline{\Omega}} & 0 \\ 0 & -\frac{1}{\overline{\Omega}} \\ \frac{1}{\overline{\Omega}} & 0 \\ 0 & \frac{1}{\overline{\Omega}} \end{bmatrix} \tag{2.1.6}$$

where $\overline{\Omega}$ is the maximum angular velocity the wheels can perform. It is worth mentioning that (2.1.6) defines an squared input constraint set, as shown in Fig. 2.2

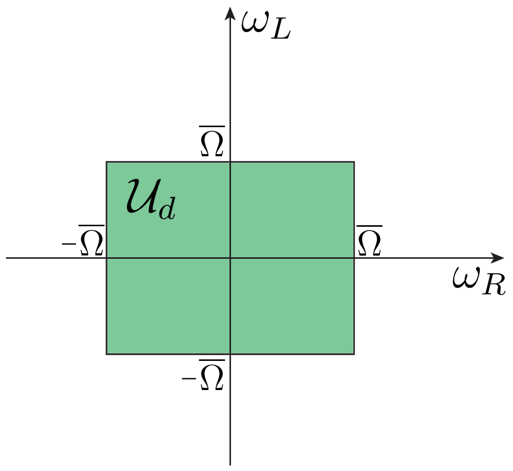


Figure 2.2: Input constraint set of the differential-drive robot

2.1.2 Unicycle Robot model

Another type of nonholonomic WMR is the unicycle, illustrated in Fig. 2.3, consisting of a single drive-wheel. Its kinematic can be obtained starting from the differetial-drive model (2.1.3). Specifically, by defining the following input-transformation:

$$\begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix} = T \begin{bmatrix} \omega_R(t) \\ \omega_L(t) \end{bmatrix}, \quad T = \begin{bmatrix} \frac{R}{2} & \frac{R}{2} \\ \frac{R}{D} & -\frac{R}{D} \end{bmatrix} \tag{2.1.7}$$

where $v(t)$ and $\omega(t)$ are the control inputs, respectively the linear and angular velocity of the vehicle, the unicycle kinematics are described by:

$$\begin{aligned} \dot{x}(t) &= v(t)\cos(\theta(t)) \\ \dot{y}(t) &= v(t)\sin(\theta(t)) \\ \dot{\theta}(t) &= \omega(t) \end{aligned} \tag{2.1.8}$$

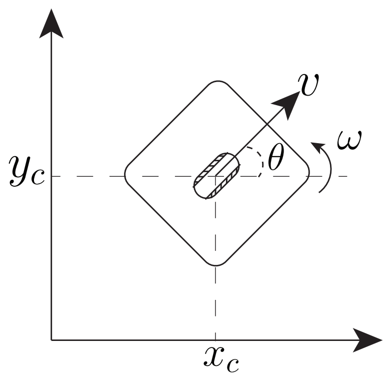


Figure 2.3: Unicycle vehicle

The transformation (2.1.7) defines a new rhombus-like constraint set $\mathcal{U}_u$ (see Fig. 2.4) for the unicycle model (2.1.8), obtained by applying (2.1.7) to (2.1.6), i.e.:

$$\mathcal{U}_u = \{[v,\omega]^T \in \mathbb{R}^2 : H_u [v,\omega]^T \leq 1\},\ H_u = H_d T^{-1} \tag{2.1.9}$$

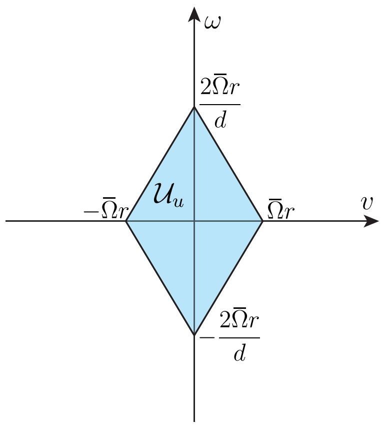


Figure 2.4: Input constraint set of the differential-drive robot

2.1.3 Car-like Vehicle Model

Another example of nonholonomic WMR is the car-like vehicle depicted in Fig. 2.5. Defining the vector of generalized coordinates $q = [x, y, \theta, \varphi]^T$,the kinematic of a car-like vehicle in Fig 2.5 with rear driving-wheels can be described by the following so-called bicycle model [91]:

$$\dot{q}(t) = \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \\ \dot{\varphi}(t) \end{bmatrix} = \begin{bmatrix} \cos\theta(t) \\ \sin\theta(t) \\ \frac{1}{l}\tan(\varphi(t)) \\ 0 \end{bmatrix} v(t) + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \omega_s(t) \tag{2.1.10}$$

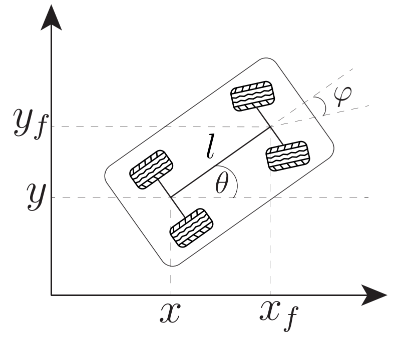


Figure 2.5: Car-like vehicle

where l is the distance between front and rear wheels, x and y represents the position of the center of mass of the vehicle within the Cartesian plane, θ is the orientation of the vehicle, φ is the steering angle, and v, ω_s are the control inputs, i.e., the linear velocity of the vehicle and the steering angular velocity respectively. It is worth to point out that the set of differential equation (2.1.10) admits a solution $\forall \varphi \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$. In other words, $\varphi = \pm\frac{\pi}{2}$ constitutes a singularity for the car-vehicle model, i.e. the car gets jammed whenever the front wheel is perpendicular to the axis of the car. However, as remarked in [91] such a singularity is neglectable since the range of $\varphi(t)$ is restricted in practical applications.
Since $n-m = 2$ for the car like vehicle, the system is subject to two different not-differentiable nonholonomic constraints in the form (2.1.2), i.e.,

$$\begin{bmatrix} \sin(\theta+\varphi) & -\cos(\theta+\varphi) & -l\cos\varphi & 0 \\ \sin\theta & -\cos(\theta) & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \\ \dot{\varphi}(t) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.1.11}$$

The control inputs are assumed subject to the following rectangular constraints, depicted in Fig. 2.6

$$\mathcal{U}_{car} = \{[v, \omega_s]^T \in \mathbb{R}^2 : \, H\,[v, \omega]^T \leq g\}, \tag{2.1.12}$$

$$H = \begin{bmatrix} -1 & 0 \\ 0 & -1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad g = \begin{bmatrix} \overline{v} \\ \overline{\omega} \\ \overline{v} \\ \overline{\omega} \end{bmatrix} \tag{2.1.13}$$

where $\overline{v}, \overline{\omega} > 0$ are positive real numbers.

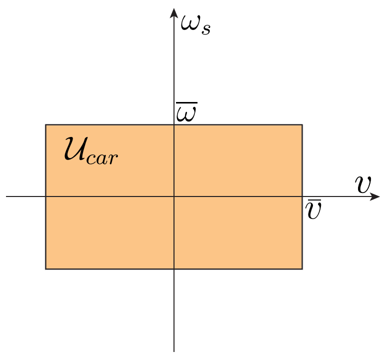


Figure 2.6: Input constraint set of the differential-drive robot

2.2 Feedback Linearization

Feedback linearization is a popular technique to exactly linearize a nonlinear model by means of a proper change of coordinates [92]. Consider a nonlinear system in the form:

$$\begin{aligned} \dot{x} &= f(x) + G(x)u \\ y &= h(x) \end{aligned} \tag{2.2.1}$$

where $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$, $y \in \mathbb{R}^m$ are the state, input, and output vectors, respectively, f and h, smooth vector fields, and $G \in \mathbb{R}^{n\times m}$ is a matrix having smooth vector fields as columns.

The feedback linearization problem consists of finding a state-feedback control

$$u(x) = \alpha(x) + E(x)v \tag{2.2.2}$$

such that if the (2.2.2) is fed into the the nonlinear system (2.2.1), it cancels out (fully or partially) the nonlinearities of (2.2.1), transforming it into an equivalent linear system. Specifically, if the nonlinear system is completely linearized, i.e. all its equations are linearized, then the linearization is said to be full-state, otherwise, if only a subset of the equations is linearized, the linearization is said to be input-output.

2.2.1 Input-Output Feedback Linearization:

To apply the input-output feedback linearization, first of all, the output dynamics must be rewritten as:

$$\begin{bmatrix} y_1^{(r_1)} \\ \vdots \\ y_m^{(r_m)} \end{bmatrix} = \begin{bmatrix} L_f^{r_1} h_1(x) \\ \vdots \\ L_f^{r_m} h_m(x) \end{bmatrix} + E(x)u \tag{2.2.3}$$

with $E(x) \in \mathbb{R}^{m \times m}$ invertible over a certain compact set Ω, and $y_i^{(r_i)}$ denoting the r_i-th derivative of the output y_i, and $L_f h = \nabla h f$ denoting the Lie derivative of h with respect to f. Then, by considering the following input transformation

$$u = E^{-1} \begin{bmatrix} v_1 - L_f^{r_1} h_1 \\ \vdots \\ v_m - L_f^{r_m} h_m \end{bmatrix} \tag{2.2.4}$$

The nonlinear system (2.2.1) is equivalent to

$$y_i^{(r_i)} = v_i \tag{2.2.5}$$

In general, by applying such a technique only a partial linearization is achieved, i.e. an internal nonlinear dynamics decoupled by the linearized system must be considered. When an internal dynamics is not obtained the linearization is said to be full-state.

2.2.2 Dynamic Feedback Linearization

When a static feedback linearization is not applicable, a dynamic feedback linearization approach may be an option [89]. The actual process may vary based on the specific nonlinear system, but in its general formulation, the problem is solved by defining new outputs consisting of a subset of the state variables. Then, the defined outputs are successively differentiated until the inputs appear in the equations. If the obtained input-output relation is singular, i.e. the decoupling matrix relating input and outputs is singular, then a certain number of integrators must be added to a subset of the inputs, which convert the input of the system into the state of a so-called dynamic compensator. The final result of such a procedure is a dynamic feedback compensator in the form:

$$\begin{aligned} \dot{\xi} &= a(x, \xi) + b(q, \xi)u \\ w &= c(q, \xi) + d(q, \xi)u \end{aligned} \tag{2.2.6}$$

such that under a transformation $z = T(x, \xi)$, the nonlinear system (2.2.1) is equivalent to a linear controllable form.

2.2.3 Input-Output Linearization of the Unicycle model

The described linearization method can be applied to the unicycle model (2.1.8) to obtain a linearized description of the vehicle's kinematics, as outlined in [89]. Specifically, by defining a new output:

$$z(t) = \begin{bmatrix} z_1(t) \\ z_2(t) \end{bmatrix} = \begin{bmatrix} x(t) + b\cos\theta(t) \\ y(t) + b\sin\theta(t) \end{bmatrix} \tag{2.2.7}$$

representing the position of a point B external to the robot, translated with respect to the vehicle's center of mass (See Fig. 2.7). Notice that the geometrical parameter $b > 0$ defines the translation of the point B. Let's consider the derivative of the new output, i.e.,

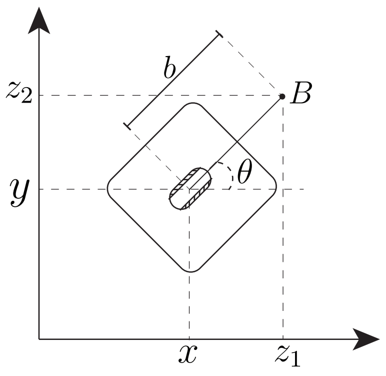


Figure 2.7: Unicycle Representation for Input-Output Linearization

$$\dot{z}(t) = \begin{bmatrix} \dot{z}_1(t) \\ \dot{z}_2(t) \end{bmatrix} = \begin{bmatrix} \dot{x}(t) - b\sin\theta(t)\dot{\theta}(t) \\ \dot{y}(t) + b\cos\theta(t)\dot{\theta}(t) \end{bmatrix} \tag{2.2.8}$$

By substituting the unicycle dynamics (2.1.8), (2.2.8) can be rewritten as:

$$\dot{z}(t) = \begin{bmatrix} v(t)\cos\theta(t) - b\sin\theta(t)\omega(t) \\ v(t)\sin(\theta(t)) + b\cos\theta(t)\omega(t) \end{bmatrix} \tag{2.2.9}$$

The latter can be written in a more compact matrix form as:

$$\dot{z}(t) = T_{NL}(\theta(t)) \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix}, \quad \text{where } T_{NL}(\theta(t)) = \begin{bmatrix} \cos\theta(t) & -b\sin\theta(t) \\ \sin\theta(t) & b\cos\theta(t) \end{bmatrix} \tag{2.2.10}$$

It is worth to mention that if $b > 0$, then the matrix T_{NL} is nonsingular and it admits the following inverse matrix:

$$T_{FL}(\theta(t)) = T_{NL}^{-1}(\theta(t)) = \begin{bmatrix} \cos\theta(t) & \sin\theta(t) \\ \frac{-\sin\theta(t)}{b} & \frac{\cos\theta(t)}{b} \end{bmatrix} \tag{2.2.11}$$

Therefore, by defining the linearizing inputs as:

$$\begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} = T_{FL}^{-1}(\theta(t)) \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix} \implies \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix} = T_{FL}(\theta) \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} \tag{2.2.12}$$

and substituting it in the nonlinear system (2.2.9), it is equivalent to the following two-single integrator model

$$\begin{aligned} \dot{z}_1(t) &= u_1(t) \\ \dot{z}_2(t) &= u_2(t) \end{aligned} \tag{2.2.13}$$

which is a Linear Time-Invariant (LTI) system in the form:

$$\dot{z}(t) = Bu(t), \quad B = I_{2\times 2} \tag{2.2.14}$$

where $u = [u_1, u_2]^T$ is the linearizing input vector. It is worth mentioning that the used linearization method consists of a partial linearization. In fact, the dynamics of the orientation $\theta(t)$ have been excluded by the linearized description. Such dynamics can be recovered by applying the input transformation (2.2.12) to $\dot{\theta} = \omega$, obtaining;

$$\dot{\theta}(t) = \frac{1}{b}(u_2(t)\cos\theta(t) - u_1(t)\sin\theta(t)) \tag{2.2.15}$$

The above equation is known as *internal dynamics* representing the nonlinear internal evolution of $\theta(t)$ which cannot be directly controlled. However, the following lemma holds:

Lemma 2.2. *If a control law $u(t)$ stabilizes the linearized system* (2.2.13)*, its internal dynamics remain bounded [89].*

Although it is not possible to apriori characterize the evolution of $\theta(t)$ with respect to its equilibrium, the carried out experiments have highlighted that under a proper choice of the control law, the internal dynamics are confined within a reasonably small neighborhood of the equilibrium. The interested reader shall refer to Chapter 3 for further details.

2.2.4 Input-Output Linearization of the Car-like Model

Similarly to the previous case, the input-output linearization can be applied to the car-like vehicle model (2.1.10). By defining a new output vector

$$z(t) = \begin{bmatrix} z_1(t) \\ z_2(t) \end{bmatrix} = \begin{bmatrix} x(t) + l\cos\theta(t) + \Delta\cos(\theta(t)+\varphi(t)) \\ y(t) + l\sin\theta(t) + \Delta\sin(\theta(t)+\varphi(t)) \end{bmatrix} \tag{2.2.16}$$

where $\Delta > 0$ defines a point P displaced with respect to the midpoint of the rear axle of the car (See Fig. 2.8). Let's consider the time derivative of the new output

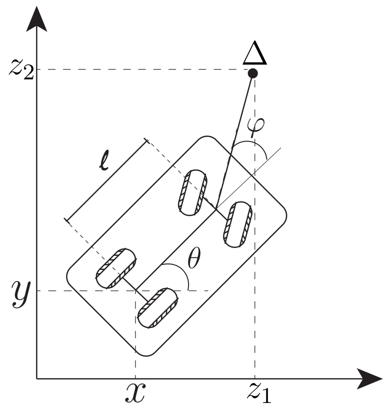


Figure 2.8: Car-like Vehicle Representation for Input-Output Linearization

$$\dot{z}(t) = \begin{bmatrix} \dot{z}_1(t) \\ \dot{z}_2(t) \end{bmatrix} = \begin{bmatrix} \dot{x}(t) + l\sin\theta(t)\dot{\theta}(t) - \Delta\sin(\theta(t)+\varphi(t))(\dot{\theta}(t)+\dot{\varphi}(t)) \\ \dot{y}(t) + l\cos\theta(t)\dot{\theta}(t) + \Delta\cos(\theta(t)+\varphi(t))(\dot{\theta}(t)+\dot{\varphi}(t)) \end{bmatrix} \tag{2.2.17}$$

By substituting the bicycle dynamics (2.1.10) in (2.2.17), it can be rewritten in a matrix form as follows:

$$\begin{gathered} \dot{z}(t) = T_{NL}(\theta(t),\varphi(t)) \begin{bmatrix} v(t) \\ \omega_s(t) \end{bmatrix}, \\ T_{NL}(\theta(t),\varphi(t)) = \begin{bmatrix} \cos\theta - \tan\varphi(\sin\theta + \frac{\Delta}{l}\sin(\theta+\varphi)) & -\Delta\sin(\theta+\varphi) \\ \sin\theta - \tan\varphi(\cos\theta + \frac{\Delta}{l}\cos(\theta+\varphi)) & \Delta\cos(\theta+\varphi) \end{bmatrix} \end{gathered} \tag{2.2.18}$$

It is worth mentioning that the matrix $T_{NL}(\theta,\varphi)$ is not singular since

$$det(T_{NL}(\theta,\varphi)) = \frac{\Delta}{\cos\varphi} \neq 0$$

The latter holds true since $\Delta > 0$ by definition. Thus, T_{NL} admits the following inverse matrix $T_{FL}(\theta,\varphi) = T_{NL}^{-1}(\theta,\varphi)$:

$$T_{FL}(\theta,\varphi) = \begin{bmatrix} \frac{\cos\theta+\cos(2\varphi+\theta)}{2} & \frac{\sin\theta+\sin(2\varphi+\theta)}{2} \\ \frac{\Delta\sin(\theta)-\Delta\sin(2\varphi+\theta)-2l\sin(\theta+\varphi)}{2l\Delta} & \frac{-\Delta\cos(\theta)+\Delta\cos(2\varphi+\theta)+2l\cos(\theta+\varphi)}{2l\Delta} \end{bmatrix} \tag{2.2.19}$$

Therefore, by defining the input transformation

$$w = T_{NL}(\theta,\varphi)\,[v,\omega_s]^T \tag{2.2.20}$$

the nonlinear system (2.2.18) is equivalent to the following two-single integrator model

$$\begin{aligned} \dot{z}_1(t) &= w_1(t) \\ \dot{z}_2(t) &= w_2(t) \end{aligned} \tag{2.2.21}$$

which is a Linear Time-Invariant (LTI) system in the form:

$$\dot{z}(t) = Bw(t), \quad B = I_{2\times 2} \tag{2.2.22}$$

where $w = [w_1, w_2]^T$ is the vector of linearizing inputs. Similarly to the unicycle case, the input-output linearization method generates nonlinear internal dynamics that can be obtained by applying the transformation (2.2.20) to the dynamics of $\dot{\theta}$ and $\dot{\varphi}$ described in (2.1.10), obtaining

$$\begin{gathered} \begin{bmatrix} \dot{\theta} \\ \dot{\varphi} \end{bmatrix} = O(\theta,\varphi)_{FL} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \\ O_{FL}(\theta,\varphi) = \begin{bmatrix} \frac{\sin\varphi\cos(\theta+\varphi)}{l} & \frac{\sin(\varphi)\sin(\theta+\varphi)}{l} \\ -\frac{\cos(\theta+\varphi)\sin\varphi}{l} - \frac{\sin(\theta+\varphi)}{\Delta} & -\frac{\sin(\theta+\varphi)\sin\varphi}{l} + \frac{\cos(\theta+\varphi)}{\Delta} \end{bmatrix} \end{gathered} \tag{2.2.23}$$

However, differently from the unicycle case, the internal dynamics are 2-dimensional and the stability of the overall control system depends on its boundedness. The stability of the internal dynamics around a reference trajectory has been deeply analyzed in [93], and further details will be given in chapter 7.

2.2.5 Dynamic Feedback Linearization of the Unicycle

In order to obtain a full-state linearization of the unicycle, here a dynamic feedback linearization method is discussed. Such a method has been successfully applied to the unicycle model (2.1.8) in [89]. Similarly to the input-output linearization, the process start by defining a new output, i.e., $z = [x, y]^T$. Then, by taking the first-order time derivative and considering the first two equations of the unicycle model we obtain:

$$\dot{z}(t) = \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \end{bmatrix} = \begin{bmatrix} v(t)\cos\theta(t) \\ v(t)\sin\theta(t) \end{bmatrix}.$$

The latter can be further differentiated with respect to the time t obtaining:

$$\ddot{z}(t) = \begin{bmatrix} \ddot{x}(t) \\ \ddot{y}(t) \end{bmatrix} = \begin{bmatrix} \dot{v}(t)\cos(\theta(t)) - \dot{\theta}v(t)sin(\theta(t)) \\ \dot{v}(t)\sin(\theta(t)) + \dot{\theta}v(t)cos(\theta(t)) \end{bmatrix} \tag{2.2.24}$$

which can be rewritten in a matrix form as:

$$\ddot{z}(t) = \begin{bmatrix} \cos\theta(t) & -v(t)\sin\theta(t) \\ \sin\theta(t) & v(t)\cos\theta(t) \end{bmatrix} \begin{bmatrix} \dot{v}(t) \\ \dot{\theta}(t) \end{bmatrix}.$$

Then, by substituting the dynamics of $\theta(t)$ from the unicycle model (2.1.8), and defining a new input variable $a(t) = \dot{v}(t)$, the latter can be rewritten as

$$\begin{bmatrix} \ddot{z}_1(t) \\ \ddot{z}_2(t) \end{bmatrix} = T_{FL}(\theta, v) \begin{bmatrix} a(t) \\ \omega(t) \end{bmatrix}, \quad T_{FL}(v, \theta) = \begin{bmatrix} \cos\theta(t) & -v(t)\sin\theta(t) \\ \sin\theta(t) & v(t)\cos\theta(t) \end{bmatrix} \tag{2.2.25}$$

The above-described approach is known in the literature as dynamic extension [89, 91] since it consists of adding an integrator on the input $v(t)$ that transforms the input variable into the state of the following so-called dynamic compensator

$$\begin{array}{rcl} \dot{v}(t) & = & a(t) \\ \omega(t) & = & \frac{u_2(t)\cos\theta(t) - u_1(t)\cos\theta(t)}{v(t)} \end{array} \tag{2.2.26}$$

Finally, by applying the state-dependent input transformation

$$\begin{bmatrix} a(t) \\ \omega(t) \end{bmatrix} = T_{FL}^{-1}(\theta(t), v(t)) \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} \tag{2.2.27}$$

to (2.2.25), the unicycle dynamics can be described by a two-double-integrator

$$\begin{aligned} \ddot{z}_1(t) &= u_1(t) \\ \ddot{z}_2(t) &= u_2(t) \end{aligned} \tag{2.2.28}$$

which can be rewritten in a matrix form as follows:

$$\dot{z}(t) = Az(t) + Bu(t), \quad A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{2.2.29}$$

where $z = [x, y, \dot{x}, \dot{y}]^T = [z_1, z_2, z_3, z_4]^T$, is the state vector of the feedback linearized system and $u = [u_1, u_2]^T$ its input vector.

2.3 Model Predictive Control

Model Predictive Control, also commonly known as Receding Horizon Control (RHC), is a popular technique to control dynamic systems. The control law is computed through the solution, at each sampling time, of finite horizon control problems [94]. Such technique is composed of three main ingredients:

- model of the plant;
- a performance-based cost function;
- a receding horizon logic.

Basically, RHC uses the mathematical model describing the plant to predict the future evolution of the system's trajectory and determines the optimal control by solving an optimization problem that minimizes a given performance cost [95].

Formally, here is assumed that the plant is modeled as a discrete-time nonlinear system in the form:

$$x(k+1) \ = \ \varphi(x(k), u(k)) \tag{2.3.1}$$

where $\varphi(\cdot, \cdot) : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n$ is a continuous function containing the origin, $x(k) \in \mathbb{R}^n$ denotes the state vector, $u(k) \in \mathbb{R}^m$ the control inputs, and $y(k) \in \mathbb{R}^p$ the output of the system. In addition, it's assumed that the system is subject to state and input constraints defined as:

$$\begin{aligned} x(k) &\in \mathcal{X} \subseteq \mathbb{R}^n, \quad \forall k \in \mathbb{Z}_+, \quad 0_x \in \mathcal{X} \\ u(k) &\in \mathcal{U} \subseteq \mathbb{R}^m, \quad \forall k \in \mathbb{Z}_+, \quad 0_u \in \mathcal{U} \end{aligned} \tag{2.3.2}$$

where $\mathcal{X}$ and $\mathcal{U}$ are compact and convex sets containing the origin.

In its general formulation [95], the objective of an RHC is to compute a state-feedback control law $u(\cdot) = g(x(\cdot))$ such that:

- The closed-loop trajectory of the system (2.3.1) asymptotically converges to the equilibrium 0_x.
- $x(k) \in \mathcal{X}$, $u(k) \in \mathcal{U}$, $\forall k \in \mathbb{Z}_+$ i.e., the prescribed constraints are fulfilled.

As mentioned before, the solution is obtained by exploiting a prediction of the system state-trajectory over a so-called prediction horizon N_p i.e.,

$$\begin{array}{rcl} x(k+1|k) &=& \varphi(x(k), u(k)) \\ x(k+2|k) &=& \varphi(x(k+1|k), u(k+1)) \\ &\vdots& \\ x(k+N_p|k) &=& \varphi(x(k+N_p-1|k), u(k+N_p-1)) \end{array} \tag{2.3.3}$$

Then, given an optimality criterion

$$J(k, x, u) = \sum_{i=0}^{k+N_p-1} l(x(i), u(i)) + L(x(k+N_p)) \tag{2.3.4}$$

an optimal sequence of current and future control inputs of length N_p

$$\mathbf{u}^* = \{u^*(k), u^*(k+1), \dots, u^*(k+N_p-1)\} \tag{2.3.5}$$

is computed by solving the following optimization problem:

$$\min_{\mathbf{u}} J(t, x, \mathbf{u}) \text{s.t.} \tag{2.3.6}$$

$$x(k+i) = \varphi(x(k+i-1), u(k+i-1)), \qquad i = k, \dots k+N_p-1 \tag{2.3.7}$$

$$u(i) \in \mathcal{U}, \qquad i = k, \dots k+N_p-1 \tag{2.3.8}$$

$$x(i) \in \mathcal{X}, \qquad i = k, \dots k+N_p-1 \tag{2.3.9}$$

$$x(k+N_p) = 0_x \tag{2.3.10}$$

where $\mathbf{u} = \{u(k), u(k+1), \dots, u(k+N_p-1)\}$ is the sequence of decision variables. In the

above optimization, (2.3.7) are the model predictions, (2.3.8) and (2.3.9) are the prescribed input and state constraints respectively, and (2.3.10) is the terminal constraint imposing that the final sample of the predicted state must be the equilibrium 0_x. If the optimization (2.3.6)-(2.3.10) admits a solution, the sequence of inputs $u^* = \{u^*(k), u^*(k+1), \ldots, u^*(k+N_p-1)\}$ brings the closed-loop trajectory to 0_x in N_p time steps.
The generic RHC algorithm is thereafter reported:

General *RHC* Algorithm

```
1: Compute the sequence $\mathbf{u}^*$ by solving (2.3.6)-(2.3.10)
2: Apply $u^*(0)$ to the system (2.3.1)
3: measure $x(k+1)$
4: $k \leftarrow k+1$, go to 1
```

Remark 2.1. (2.3.10) is a hard constraint that, although guarantees asymptotic stability, might compromise the feasibility of the optimization i.e., the problem might not admit a solution [94]. To avoid such a drawback, constraint (2.3.10) can be relaxed. Different solutions are possible:

- The RHC can be formulated as an infinite horizon optimal control.
- The terminal state can be constrained to lie in a terminal control invariant region containing the origin, instead of imposing it to reach 0_x.

In the following, the second solution is analyzed.
First, it is worth formally defining the concept of control invariance [96].

Definition 2.1. A set $\Sigma \subset \mathcal{X}$ is said to be control invariant with respect to the system (2.3.1), subject to constraints (2.3.2), if

$$\forall x(k) \in \Sigma, \quad \exists u \in \mathcal{U} \text{ s.t. } \quad x(k+1) \in \Sigma, \quad \forall k \in \mathbb{Z}_+ \tag{2.3.11}$$

In other words, if the trajectory of the system starts in the control invariant region Σ, then there surely exists an admissible control law input that keeps the trajectory within Σ. Such a region can be exploited to relax the constraint (2.3.10), as follows:

$$x(k+N_p) \in \Sigma \tag{2.3.12}$$

In the literature, different techniques have been proposed to design a state-feedback controller and the associated control invariant region Σ [97]. By combining the RHC algorithm and a state-feedback controller associated to Σ, it is possible to obtain a different control scheme, known in the literature as Dual-Mode MPC [54]. The computed control law in the first N_p time instants is open-loop, while after the N_p-th time step, a feedback control law associated with the terminal region is used. The stability properties of such an algorithm have been formally proven in the literature. A deep analysis pertaining to the closed-loop stability of the RHC scheme can be found in [94].

Although formulated for a general class of dynamic systems, the MPC algorithm can be successfully applied to solve practical problems related to the control of unmanned wheeled vehicles.

2.4 Set-Theoretic Control

2.4.1 Preliminary Definitions

Definition 2.2. Given two sets $\mathcal{A}$, $\mathcal{B} \subset \mathbb{R}^n$, their Minkowski/Pontryagin sum ($\oplus$) and difference ($\ominus$) are [96]:

$$\begin{aligned}\mathcal{A} \oplus \mathcal{B} &:= \{a + b : \, a \in \mathcal{A}, \, b \in \mathcal{B}\} \\ \mathcal{A} \ominus \mathcal{B} &:= \{a \in \mathcal{A} : \, a + b \in \mathcal{A}, \, \forall b \in \mathcal{B}\}.\end{aligned}$$

Definition 2.3. Given the ellipsoidal set

$$\mathcal{E} := \{z \in \mathbb{R}^n : \, z^T E^{-1} z \leq 1\}, \, E = E^T > 0, \, E \in \mathbb{R}^{n \times n}$$

and a matrix $M \in \mathbb{R}^{n \times n}$, then [98]

$$M\mathcal{E} := \{z \in \mathbb{R}^n : z^T (MEM^T)^{-1} z \leq 1\}$$

Property 2.1. Given two ball sets $\mathcal{C}_1$ and $\mathcal{C}_2$ in the form:

$$\begin{aligned}\mathcal{C}_1 &:= \{z \in \mathbb{R}^2 : \, z^T Q_1^{-1} z \leq 1\}, \quad Q_1 = r_{\mathcal{C}_1}^2 I \\ \mathcal{C}_2 &:= \{z \in \mathbb{R}^2 : \, z^T Q_2^{-1} z \leq 1\}, \quad Q_2 = r_{\mathcal{C}_2}^2 I\end{aligned} \tag{2.4.1}$$

where $r_{\mathcal{C}_1}$ and $r_{\mathcal{C}_2}$ are the radii of $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively, the Minkowski sum of $\mathcal{C}_1$ and $\mathcal{C}_2$ is defined as follows:

$$\mathcal{C}_1 \oplus \mathcal{C}_2 = \{z \in \mathbb{R}^2 : z^T Q_s^{-1} z\},\ Q_s = r_s^2 I,\ \ r_s = r_{\mathcal{C}_1} + r_{\mathcal{C}_2}$$

while, assuming $r_{\mathcal{C}_1} > r_{\mathcal{C}_2}$ the Minkowski difference can be computed as follows:

$$\mathcal{C}_1 \ominus \mathcal{C}_2 = \{z \in \mathbb{R}^2 : z^T Q_d^{-1} z\},\ Q_d = r_d^2 I,\ r_d = r_{\mathcal{C}_1} - r_{\mathcal{C}_2}$$

and given $M \in \mathbb{R}^{2\times 2} = mI$ the affine transformation $M\mathcal{C}_1$ is given by:

$$M\mathcal{C}_1 = \{z \in \mathbb{R}^2 :\ z^T Q_a^{-1} z \le 1\},\ Q_a = r_a^2 I,\ r_a = m r_{\mathcal{C}_1}$$

Definition 2.4. A function $f(t) : \mathbb{R} \to \mathbb{R}^n$ is said uniformly bounded and smooth if $\forall t,\ \exists \varGamma > 0 :\ \|f(t)\| < \varGamma$ and $f(t) \in \mathcal{C}^2$.

Let's consider the following discrete-time autonomous nonlinear system subject to bounded disturbance:

$$z(k+1) = f(z(k), d(k)), \quad d(k) \in \mathcal{D} \tag{2.4.2}$$

where $k \in \mathbb{Z} := \{0, 1, \ldots\}$, $z \in \mathbb{R}^n$ is the system's state, $d \in \mathbb{R}^n$ is the bounded disturbance. Moreover, $\mathcal{D} \subset \mathbb{R}^n$ is a compact and convex set containing the origin.

Definition 2.5 ([96])**.** The set $\mathcal{S} \subset \mathbb{R}^n$. is Robust Positive Invariant (RPI) for (2.4.2) if

$$\forall z(0) \in \mathcal{S} \quad \Longrightarrow \quad z(k) \in \mathcal{S}, \quad \forall d \in \mathcal{D}, \quad \forall k \in \mathbb{Z}_+ \tag{2.4.3}$$

If $d(k) = 0_m$, $\forall k \in \mathbb{Z}_+$, then S is said to be Positive Invariant (PI).

Let's now consider a generic discrete-time nonlinear system subject to bounded disturbance:

$$z(k+1) = f(z(k), u(k), d(k)), \quad u(k) \in \mathcal{U},\ d(k) \in \mathcal{D} \tag{2.4.4}$$

where $k \in \mathbb{Z} := \{0, 1, \ldots\}$, $z \in \mathbb{R}^n$ is the system's state, $u \in \mathbb{R}^m$ the control inputs, $d \in \mathbb{R}^n$ is the bounded disturbance. Moreover, $\mathcal{U} \subset \mathbb{R}^m$, $\mathcal{D} \subset \mathbb{R}^n$ are compact and convex sets containing the origin.

Definition 2.6 ([96]). The set $\mathcal{S} \subset \mathbb{R}^n$. is Robust Control Invariant (RCI) for (2.4.4) if

$$\forall z(k) \in \mathcal{S} \quad \Longrightarrow \quad \exists u(k) \in \mathcal{U}, \text{ s.t. } f(z(k), u(k), d(k)) \in \mathcal{S}, \quad , \forall d(k) \in \mathcal{D}, \quad \forall k \in \mathbb{Z}_+ \tag{2.4.5}$$

If $d(k) = 0_m$, $\forall k \in \mathbb{Z}_+$, then S is said to be Control Invariant (CI).

Definition 2.7 ([96]). Consider the constrained system (2.4.4) and a target set $\mathcal{Z} \subset \mathbb{R}^n$. The set of states $\mathcal{T}^c \subset \mathbb{R}^n$ Robust One-Step Controllable (ROSC) to $\mathcal{Z}$ for (2.4.4) is defined as:

$$\mathcal{T} := \{z \in \mathbb{R}^n : \exists u \in \mathcal{U} \text{ s.t. } f(z, u, d) \in \mathcal{Z}, \forall d \in \mathcal{D}\} \tag{2.4.6}$$

Definition 2.8 ([96]). Consider the constrained system (2.4.4) and a set $\mathcal{Z} \subset \mathbb{R}^n$. The set of states $\mathcal{T}^r \subset \mathbb{R}^n$ Robust One-Step Reachable (ROSR) from $\mathcal{Z}$ for (2.4.4) is defined as:

$$\mathcal{T}^r := \{z \in \mathbb{R}^n : \exists z(0) \in \mathcal{Z}, \exists u \in \mathcal{U}, \exists d \in \mathcal{D} \text{ s.t. } z = f(z(0), u, d)\} \tag{2.4.7}$$

Let's consider a discrete-time linear system subject to bounded disturbance

$$z(k+1) = Az(k) + Bu(k) + d(k), \ \ u(k) \in \mathcal{U}, \ d(k) \in \mathcal{D} \tag{2.4.8}$$

Definition 2.9 ([96]). The set $\mathcal{S} \subset \mathbb{R}^n$ is said to be Robust Control Invariant (RCI) for (2.4.8) if

$$\forall z(k) \in \mathcal{S}, \implies \exists u(k) \in \mathcal{U} \text{ s.t. } Az(k) + Bu(k) + d(k) \in \mathcal{S}, \forall d(k) \in \mathcal{D}, \forall k \in \mathbb{Z}_+ \tag{2.4.9}$$

Definition 2.10. Consider the constrained system (2.4.8) and a target set $\mathcal{Z} \subset \mathbb{R}^n$. The set of states $\mathcal{T}^c \subset \mathbb{R}^n$ Robust One-Step Controllable (ROSC) to $\mathcal{Z}$ for (2.4.8) is defined as:

$$\mathcal{T}^c := \{z \in \mathbb{R}^n : \exists u \in \mathcal{U} \text{ s.t. } Az + Bu + d \in \mathcal{Z}, \forall d \in \mathcal{D}\} \tag{2.4.10}$$

Such a set can be computed as follows:

$$\mathcal{T}^c = ((\mathcal{Z} \ominus \mathcal{D}) \oplus (-B \circ \mathcal{U})) \circ A \tag{2.4.11}$$

where the operator $\circ$ denotes an affine operation on sets.

Definition 2.11 ([96])**.** Consider the constrained system (2.4.8) and a set $\mathcal{Z} \subset \mathbb{R}^n$. The set of states $\mathcal{T}^r \subset \mathbb{R}^n$ Robust One-Step Reachable (ROSR) from $\mathcal{Z}$ for (2.4.8) is defined as:

$$\mathcal{T}^r := \{z \in \mathbb{R}^n : \exists z(0) \in \mathcal{Z},\ \exists u \in \mathcal{U},\ \exists d \in \mathcal{D} \text{ s.t. } z = Az(0) + Bu + d\} \tag{2.4.12}$$

Such a set can be computed as follows:

$$\mathcal{T}^r = ((A \circ \mathcal{Z}) \oplus (B \circ \mathcal{U})) \oplus \mathcal{D} \tag{2.4.13}$$

where the operator $\circ$ denotes an affine operation on sets.

2.4.2 Set-Theoretic Receding Horizon Control Scheme:

The constrained system (2.4.8) can be stabilized using the ST-RHC scheme proposed in [54]. Such a dual-mode receding horizon control strategy can be summarized as follows:
- *Offline:* First, by considering model (2.4.8) in a disturbance-free scenario (i.e., $d(k) = 0, \forall k$), design a state-feedback controller $u(k) = -Kz(k)$ such that $A-BK$ is asymptotically stable. Then, compute the smallest RCI region, namely $\mathcal{T}^0$, associated to the controlled system. Finally starting from $\mathcal{T}^0$, recursively apply Definition 2.10 to build a family of ROSC sets $\mathcal{T}^i$ until the set growth saturates (i.e., $\mathcal{T}^{i+1} = \mathcal{T}^i$) or the desired state-space region is covered. Store the computed family $\{\mathcal{T}^i\}_{i=1}^N$, where N is the number of computed ROSC sets.
- *Online ($\forall k$):* Determine the smallest set-membership index $i(k)$ of the ROSC set $\mathcal{T}^{i(k)}$ containing $z(k)$. Then

- **if** $i(k) = 0$, **then** apply $u(k) = -Kz(k)$
- **else** solve the following convex optimization problem:

$$\begin{gathered} u(k) = \arg\min_u J(z(k), u) \ s.t. \\ Az(k) + Bu \in (\mathcal{T}^{i(k)-1} \ominus \mathcal{D}) \\ u \in \mathcal{U} \end{gathered} \tag{2.4.14}$$

where $J(z(k), u)$ is a convex cost function.

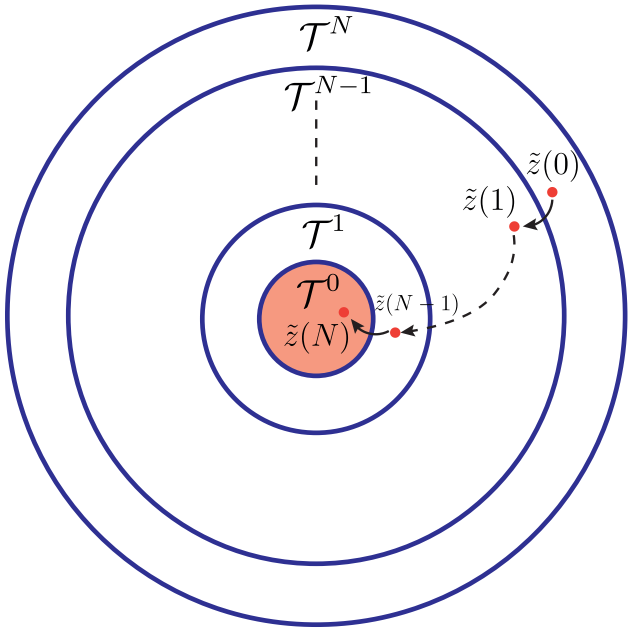


Figure 2.9: Modus operandi of Set-Theoretic Control algorithm

Property 2.2. The ST-RHC controller, whose modus operandi is depicted in Fig. 2.9, enjoys the following properties [54]:

- The optimization (2.4.14) enjoys recursive feasibility.
- The state trajectory is uniformly ultimately bounded in $\mathcal{T}_0$. In particular, starting from any $z(0) \in \bigcup_{i=0}^{N} \mathcal{T}^i$, the terminal RCI region T^0 is reached in at most N steps regardless of any disturbance realization $d(k) \in \mathcal{D}$.

3 A Receding Horizon Trajectory Tracking Strategy for Input Constrained Differential Drive Robots via Feedback Linearization

In this chapter, a novel approach is proposed to solve the trajectory tracking control problem for input-constrained differential-drive robots. In particular, a robust set-based receding horizon tracking scheme is developed, which is capable of dealing with state-dependent input constraints arising when the vehicle's dynamics are approached by a standard feedback linearization technique. First, the worst-case input constraint set is offline characterized and an admissible, although not optimal, controller is proposed. Then, online, by leveraging the knowledge of the robot's orientation, the offline constraint set is enlarged in a receding-horizon fashion, with a consequent improvement of the tracking performance. Recursive feasibility and constraint fulfillment are formally proven. Finally, the approach's effectiveness is experimentally validated on a Khepera IV differential-drive robot by comparing the control performance with several competitor schemes.

3.1 Problem Formulation

Let's consider a differential-drive robot whose kinematics is described by the nonlinear model (2.1.3), subject to the wheels' angular velocity constraints (2.1.6), and a bounded reference trajectory $r(t) = [x_r(t), y_r(t)]^T$ defining the desired x and y position of the center of mass of robot in the plane. The problem addressed in this chapter is known as *Constrained Trajectory Tracking problem* and can be formally stated as follows:

Problem 1. *Design a feedback control law* $[\omega_R(t), \omega_L(t)]^T = \phi(\left[p_c(t)^T, \theta(t)\right]^T, r(t))$ *such that the tracking error* $\xi(t) = r(t) - p_c(t)$ *is bounded and* $[\omega_R(t), \omega_L(t)]^T \in \mathcal{U}_d,\ \forall t \geq 0.$

To address such a problem, in the following the robot model is recast into a unicycle model and linearized via the input-output feedback linearization presented in Chapter 2.2.3. The time-varying input constraint set associated with the feedback linearized model is characterized and its worst-case realization is analytically derived. Then, the trajectory tracking problem is formulated as a discrete-time Receding Horizon Control (RHC) tracking problem using the constrained input-output linearized description.

3.2 Linearized Vehicle Model

In Sections 2.1.1-2.1.2 it has been shown how the differential-drive kinematics (2.1.3) can be recast into the unicycle one (2.1.8) by using the input transformation (2.1.7). On the other hand, the unicycle kinematics can be input-output linearized by another input transformation (2.2.12) obtaining the following model:

$$\dot{z}(t) = Bu(t), \qquad B = I_{2\times 2} \tag{3.2.1a}$$

$$\dot{\theta}(t) = D(\theta)u(t), \quad D(\theta) = \begin{bmatrix} -\frac{sin(\theta(t))}{b} & \frac{\cos(\theta(t))}{b} \end{bmatrix} \tag{3.2.1b}$$

where $z(t) = [x(t) + b\cos\theta(t), y(t) + b\sin\theta(t)]^T$ is the cartesian position of a point B translated with respect to the center of mass of the robot (See Fig. 2.7), and $u(t) = [u_1(t), u_2(t)]^T$ is the vector of linearizing control inputs. As discussed in chapter 2.2.3, the unicycle is equivalent to a two-single integrator model (3.2.1a) with a decoupled stable internal dynamics (3.2.1b).

3.3 Input constraints characterization

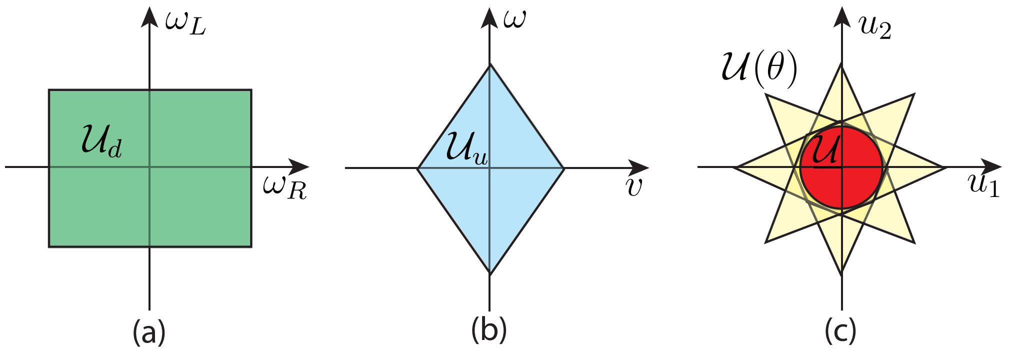


Figure 3.1: Input constraints: *(a)* differential-drive, *(b)* unicycle, (c) feedback linearized model.

As outlined in 2.1.2, under the transformation (2.1.7) and the unicycle model (2.1.8), the input constraint set (2.1.6) is mapped into a rhombus-like set $\mathcal{U}_u$ on the liner angular velocities of the robot (see Fig 2.4.b), i.e., $[v, \omega]^T \in \mathcal{U}_u$, described by equation (2.1.9). Moreover, by applying (2.2.12) to (2.1.9), the set $\mathcal{U}_u$ transforms according to the robot's orientation θ. As a consequence, the linearized model (3.2.1) is subject to state-dependent input constraints defined as rhombus-like sets $\mathcal{U}(\theta)$, where

$$\mathcal{U}(\theta) = \{[u_1, u_2]^T \in \mathbb{R}^2 : \, H(\theta)\,[u_1, u_2]^T \leq 1\},$$
$$H(\theta) = H_u T_{FL}(\theta) = \begin{bmatrix} \frac{D\sin\theta - 2\cos\theta b}{2\bar{\Omega}Rb} & \frac{-D\cos\theta - 2\sin\theta b}{2\bar{\Omega}Rb} \\ \frac{-D\sin\theta - 2\cos\theta b}{2\bar{\Omega}Rb} & \frac{D\cos\theta - 2\sin\theta b}{2\bar{\Omega}Rb} \\ \frac{-D\sin\theta + 2\cos\theta b}{2\bar{\Omega}Rb} & \frac{D\cos\theta + 2\sin\theta b}{2\bar{\Omega}Rb} \\ \frac{D\sin\theta + 2\cos\theta b}{2\bar{\Omega}Rb} & \frac{-D\cos\theta + 2\sin\theta b}{2\bar{\Omega}Rb} \end{bmatrix} \tag{3.3.1}$$

The above-mentioned input set transformations are depicted in Fig. 3.1. Specifically, in the differential-drive input set $\mathcal{U}_{dd}$ is shown in Fig. 3.1-(a), the unicycle input set $\mathcal{U}_u$ in Fig. 3.1-(b), and the time-varying input set $\mathcal{U}(\theta)$ of the input-output linearized system is shown in Fig. 3.1-(c) Notice that, although (3.3.1) is time-varying, a worst-case time-invariant input constraint $\underline{\mathcal{U}}$ set such that $\underline{\mathcal{U}} \subset \mathcal{U}(\theta)$, $\forall\theta$, can be defined as:

$$\underline{\mathcal{U}} := \bigcap_{\theta=0}^{2\pi} \mathcal{U}(\theta) \tag{3.3.2}$$

i.e. as the greatest circle inscribed in every-rotating polyhedron. Indeed, it will be shown that the rotation of the polyhedron does not affect the diameter of such an inner circular approximation.

3.4 Worst-case Inner Approximation of the Input Constraint Set

The following lemma analytically characterizes the worst-case inner approximation of the time-varying input constraint set $\mathcal{U}(\theta)$.

Lemma 3.1. *The worst-case input constraint set* (3.3.2) *is equal to the following ball set*

$$\underline{\mathcal{U}}(r_u) = \{u \in \mathbb{R}^2 \,|u^T u \leq r_u^2\}, \; r_u = \frac{2\bar{\Omega}Rb}{\sqrt{4b^2 + D^2}} \tag{3.4.1}$$

Proof. The vertices $V_i(\theta)$, $i = 1, \ldots, 4$ of the polyhedron (3.3.1) can be analytically computed intersecting the four hyperplanes shaped by $H_{[i,:]}(\theta)[u_1, \, u_2]^T = 1$, $i = 1, \ldots 4$, (with $H_{[i,:]}(\theta)$ denoting the $i-th$ row of $H(\theta)$), obtaining

$$\begin{aligned} V_1(\theta) &= \left[-\bar{\Omega}R\cos\theta, -\bar{\Omega}R\sin\theta\right]^T, \; V_3(\theta) = -V_1(\theta) \\ V_2(\theta) &= \left[\tfrac{2\bar{\Omega}Rb\sin\theta}{D}, -\tfrac{2\bar{\Omega}Rb\cos\theta}{D}\right]^T, \; V_4(\theta) = -V_2(\theta) \end{aligned} \tag{3.4.2}$$

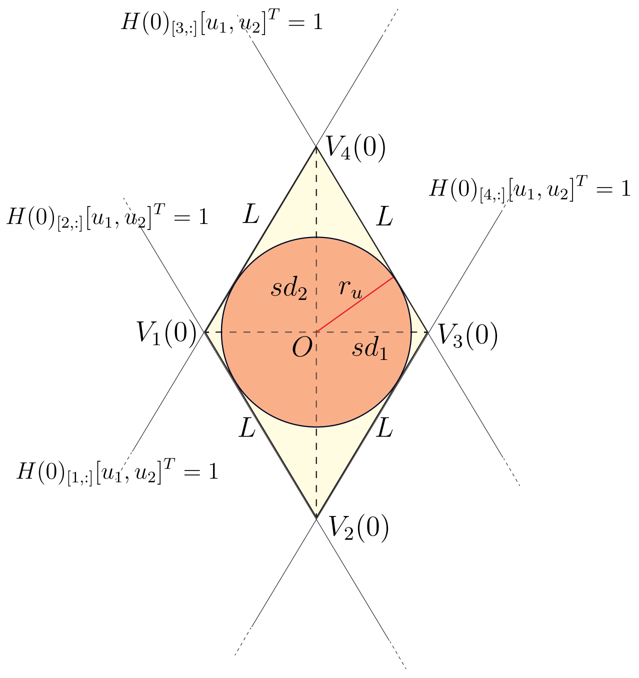

$H(0)_{[3,:]}[u_1, u_2]^T = 1$
$V_4(0)$
$H(0)_{[4,:]}[u_1, u_2]^T = 1$
$H(0)_{[2,:]}[u_1, u_2]^T = 1$
L
L
sd_2
r_u
$V_1(0)$
O
sd_1
$V_3(0)$
L
L
$H(0)_{[1,:]}[u_1, u_2]^T = 1$
$V_2(0)$

Figure 3.2: Input constraint set $\mathcal{U}(0)$ and inscribed circle $\underline{\mathcal{U}}(r_u)$.

Then, by computing from (3.4.2) the lengths L_i, $i = 1, \ldots, 4$ of each side of $\mathcal{U}(\theta)$, it is straightforward to verify that they are all equal and independent of θ, i.e.,

$$L_i = L = \frac{\bar{\Omega} R}{D}\sqrt{4b^2 + D^2}, \quad i = 1, \ldots, 4 \tag{3.4.3}$$

Therefore, the transformation (2.2.12) defines a rhombus set $\mathcal{U}(\theta)$ centered in the origin and rigidly rotating into the plane. As a consequence, the set $\underline{\mathcal{U}}$ is equivalent to the biggest circle inscribed into $\mathcal{U}(\theta)$ for any θ. By considering for simplicity and w.l.o.g. only $\theta = 0$ (see Fig. 3.2), the radius $r_u \in \mathbb{R}$ of the biggest circle within $\mathcal{U}(0)$ can be computed resorting to simple geometric arguments:

$$r_u = \frac{sd_1 sd_2}{\sqrt{sd_1^2 + sd_2^2}} = \frac{2\bar{\Omega} R b}{\sqrt{4b^2 + D^2}} \tag{3.4.4}$$

where $sd_1 = [1,\, 0]V_3(0)$ and $sd_2 = [0,\, 1]V_4(0)$ are the length of the two semi-diagonals of $\mathcal{U}(0)$. $\square$

3.5 Proposed receding Horizon Tracking Controller

In the following, the input-output linearized system (3.2.1) will be used to design a receding horizon controller that solved the tracking problem 1. Although such a system includes the nonlinear internal dynamics describing the orientation θ, it has been proved that the evolution of such internal dynamics remains bounded if the system (3.2.1a) is stable. The following remark formally states such a property.

Remark 3.1. A linear tracking controller for (3.2.1a) allows the point **B** to track any reference trajectory with stable internal dynamics for $\theta(t)$ [89, 93, Section 3.3].

Let $T_s > 0$ be a sufficiently small sampling time, the model (3.2.1a) can be discretized by resorting to the forward Euler method, obtaining:

$$z(k+1) = Az(k) + Bu(k) \tag{3.5.1}$$

where, $k \in \mathbb{Z}_+ := \{0, 1, \ldots\}$ denotes the discrete-time instants, $A = I_{2\times 2}$, and $B = T_s I_{2\times 2}$. Note that the same state-space description is achieved when the zero-order hold discretization method is used.

Property 3.1. The input-output feedback linearization (2.2.7),(2.2.12),(2.2.13) and forward Euler discretization commute when the unicycle robot model (2.1.8) is considered.

Proof. By resorting to the forward Euler discretization method, the discrete-time characterization of (2.1.8) is

$$\begin{aligned} x_c(k+1) &= x_c(k) + T_s v(k)\cos(\theta(k)) \\ y_c(k+1) &= y_c(k) + T_s v(k)\sin(\theta(k)) \\ \theta(k+1) &= \theta(k) + T_s\omega(k) \end{aligned} \tag{3.5.2}$$

By differentiating the outputs $z_1(t)$ and $z_2(t)$ in (2.2.7)

$$\begin{aligned} \dot{z}_1(t) &= \dot{x}_c(t) - b\sin(\theta(t))\dot{\theta}(t) \\ \dot{z}_2(t) &= \dot{y}_c(t) + b\cos(\theta(t))\dot{\theta}(t) \end{aligned}$$

one obtains under discretization arguments

$$\begin{aligned} \tfrac{z_1(k+1)-z_1(k)}{T_s} &= \tfrac{x_c(k+1)-x_c(k)}{T_s} - b\sin(\theta(k))\tfrac{\theta(k+1)-\theta(k)}{T_s} \\ \tfrac{z_2(k+1)-z_2(k)}{T_s} &= \tfrac{y_c(k+1)-y_c(k)}{T_s} + b\cos(\theta(k))\tfrac{\theta(k+1)-\theta(k)}{T_s} \end{aligned}$$

Then, by substituting $x_c(k+1)$, $y_c(k+1)$ and $\theta(k+1)$ with the right-hand sides of (3.5.2), the resulting discrete-time evolutions of z_1 and z_2 are

$$\begin{aligned} z_1(k+1) &= z_1(k) + T_s v(k)\cos(\theta(k)) - T_s b \sin(\theta(k))\omega(k) \\ z_2(k+1) &= z_2(k) + T_s v(k)\sin(\theta(k)) + T_s b \cos(\theta(k))\omega(k) \end{aligned}$$

whose compact form is

$$\begin{bmatrix} z_1(k+1) \\ z_2(k+1) \end{bmatrix} = \begin{bmatrix} z_1(k) \\ z_2(k) \end{bmatrix} + T_s T_{FL}^{-1}(\theta(k)) \begin{bmatrix} v(k) \\ \omega(k) \end{bmatrix} \tag{3.5.3}$$

Finally, by using the input transformation (2.2.12), the discrete-time model (3.5.3) becomes equal to (3.5.1). Hence, input-output linearization and discretization commute. As a consequence, for a sufficiently small value of the sampling time T_s, the discrete-time linear system (3.5.1) well approximates the input-output linearized continuos-time system. □

In the following By denoting with $r^z(k)$ the reference signal for the **B** point model (i.e., obtained applying the transformation (2.2.7) to $r(k)$), and by considering (3.3.1) and (3.5.1), a solution to Problem 1 can be, in principle, achieved via the following constrained optimization problem over a prediction horizon N_p :

$$\begin{gathered} \min_{\{u(k+j|k)\}} \sum_{j=0}^{N_p} ||\xi(k+j|k)||_{Q_x} + ||u(k+j|k)||_{Q_u} \ s.t. \\ z(k+j+1|k) = Az(k+j|k) + Bu(k+j), \\ u(k+j|k) \in \mathcal{U}(\theta(k+j|k)), \\ j = 0, \ldots, N_p - 1, \end{gathered} \tag{3.5.4}$$

where $\xi(k) = r^z(k) - z(k) = r(k) - p_c(k)$ is the tracking error, and $Q_x = Q_x^T > 0$ and $Q_u = Q_u^T > 0$ two weight matrices. Moreover, $z(k+j|k)$, $\theta(k+j|k)$ are $j-$th step ahead state and orientation prediction, respectively. Similarly, $u(k+j|k)$ is the control move at $k+j$.

Remark 3.2. Notice that the optimization (3.5.4) is nonlinear and non-convex. The latter is due to the presence of the state-dependent input constraint $\mathcal{U}(\theta(k+j|k))$ whose prediction involves, at each step, the nonlinear transformation (2.2.12). As a consequence, although appealing, the optimization (3.5.4) does not present any tangible advantage when compared to nonlinear MPC counterparts [27,28,30]. On the other hand, a viable approach would require robustly approximating the state-dependent input constraint set over the prediction horizon N_p. Unfortunately, this leads to a computationally demanding solution that becomes very conservative even for small values of the prediction horizon, see, e.g., [82, 83] and references therein. Moreover, for a small horizon, closed-loop stability might not be ensured [94]. $\square$

In the sequel, we tackle the constrained trajectory tracking problem by properly adapting the infinite horizon ($N_p \to \infty$) receding horizon controller developed in [99, 100], where an upper-bound of the cost function in (3.5.4) is minimized by means of constant state-feedback control law.

First, by considering the worst-case realization (3.4.1) of the state-dependent input constraint set $\mathcal{U}(\theta)$, we offline derive a guaranteed, although conservative, solution. Then, during the online phase, the offline controller is updated by exploiting the knowledge of the current robot orientations $\theta(k)$ and proper set-inclusions. This allows to drastically reduce its conservativeness and preserve the recursive feasibility property.

Assumption 3.1. The robot is equipped with an onboard vision module and an online trajectory planner providing a reference signal $r(k)$ within a vision radius $d_r > 0$. The latter implies that $r(k) - p_c(k) \in \mathcal{B}(d_r), \forall\, k,$ with $\mathcal{B}(d_r) = \{\xi \in \mathbb{R}^2 : \xi^T B^{-1}(d_r)\xi \leq 1\}$, $B(d_r) = \begin{bmatrix} d_r^2 & 0 \\ 0 & d_r^2 \end{bmatrix}$.

Definition 3.1. *A set $\Sigma \subset \mathbb{R}^n$ is said Control Invariant (CI) for* (3.5.1) *under and admissible control law $u(k) \in \mathcal{U}(\theta)$ if $\forall\, z(k) \in \Sigma \Rightarrow Az(k) + Bu(k) \in \Sigma$ [96].*

3.5.1 Offline solution:

Proposition 3.2. *Consider the vehicle linearized model* (3.5.1) *and the worst-case input constraint* (3.4.1)*. If the following semidefinite programming problem*

$$[Q_0^*,\, Y_0^*, \gamma_0^*] = \arg \min_{\gamma, Q, Y} \quad \gamma \quad s.t. \tag{3.5.5a}$$

$$\begin{bmatrix} 1 & \xi^T(0) \\ \xi(0) & Q \end{bmatrix} \geq 0, \quad Q = Q^T > 0 \tag{3.5.5b}$$

$$\begin{bmatrix} Q & QA^T + Y^T B^T & QQ_x^{\frac{1}{2}} & Y^T Q_u^{\frac{1}{2}} \\ AQ + BY & Q & 0 & 0 \\ Q_x^{\frac{1}{2}} Q & 0 & \gamma I & 0 \\ Q_u^{\frac{1}{2}} Y & 0 & 0 & \gamma I \end{bmatrix} \geq 0 \tag{3.5.5c}$$

$$\begin{bmatrix} r_u^2 I & Y \\ Y^T & Q \end{bmatrix} \geq 0 \tag{3.5.5d}$$

$$Q \geq B(d_r) \tag{3.5.5e}$$

admits a solution and $r(k)$ is a reference complying with Assumption 3.1, then the state-feedback control law

$$[\omega_r(k), \omega_l(k)] = -T^{-1} T_{FL}(\theta(k)) Y_0^* Q_0^{*-1} \xi(k) \tag{3.5.6}$$

solves the constrained Problem 1. Moreover, $\forall\, k \geq 0$, $\xi(k)$ belongs to the ellipsoidal control invariant set $\mathcal{E}_0 = \{\xi \in \mathbb{R}^2 : \xi^T Q_0^{*-1} \xi \leq 1\}$.

Proof. Initially, consider the scenario $r^z(k) = 0, \forall\, k$ (i.e., a regulation problem). As shown in [99], the state-feedback control law

$$u(k) = -Y_0^* Q_0^{*-1} \xi(k) \tag{3.5.7}$$

obtained solving the optimization problem (3.5.5a)-(3.5.5d), guarantees closed-loop asymptotic stability of (3.5.1) and worst-case input-constraint (3.4.1) fulfillment for any $\xi(0) \in \mathcal{E}_0$, with $\mathcal{E}_0$ CI. By noticing that $(\bar{r}^z, 0_2)$ is the equilibrium pair for (3.5.1) under any constant admissible reference $r^z(k) = \bar{r}^z$ [101, Section 2], the tracking error dynamics have the same structure of (3.5.1): $\xi(k+1) = A\xi(k) + Bu(k)$. Moreover, since the equilibrium input is always zero and no state constraints act on (3.5.1), then the tracking error dynamics are also subject to the same worst-case input constraint (3.4.1). Consequently, an admissible setpoint tracking control law for (3.5.1) is given by the solution of (3.5.5a)-(3.5.5d) if $\xi(k) = \bar{r}^z - z(k) \in \mathcal{E}_0$ (i.e., the error signal remains confined within the controller's domain). By generalizing such a result to a generic reference signal $r(k)$ complying with Assumption 3.1, the admissibility condition $\xi(k) \in \mathcal{E}_0$ is fulfilled $\forall\, k$ if $\mathcal{E}_0 \supseteq \mathcal{B}(d_r)$. The latter translates into a further Linear Matrix Inequalities (LMI) constraint, i.e., $Q \geq B(d_r)$ (see (3.5.5e)) that must be added to (3.5.5). Finally, since the control law (3.5.7) ensures bounded tracking error for the linearized model (3.5.1) and the internal dynamics of $\theta(t)$ is stable (see Remark 3.1), the nonlinear feedback control law (3.5.6), obtained applying the transformations (2.1.7) and (2.2.12) to (3.5.7), solves Problem 1. $\square$

3.5.2 Online receding horizon controller:

Proposition 3.3. *If* (3.5.5) *admits a solution at* $k = 0$ *and* $r(k)$ *is an admissible reference complying with Assumption 3.1, then the following semidefinite programming problem*

$$[Q_k^*, Y_k^*, \gamma_k^*] = \arg \min_{\gamma_k, Q_k, Y_k} \quad \gamma_k \quad s.t. \tag{3.5.8a}$$

$$\begin{bmatrix} 1 & \xi(k)^T \\ \xi(k) & Q_k \end{bmatrix} \geq 0, \quad Q = Q^T > 0 \tag{3.5.8b}$$

$$\begin{bmatrix} Q_k & Q_k A^T + Y_k^T B^T & Q_k Q_x^{\frac{1}{2}} & Y_k^T Q_u^{\frac{1}{2}} \\ AQ_k + BY_k & Q_k & 0 & 0 \\ Q_x^{\frac{1}{2}} Q_k & 0 & \gamma_k I & 0 \\ Q_u^{\frac{1}{2}} Y_k & 0 & 0 & \gamma_k I \end{bmatrix} \geq 0 \tag{3.5.8c}$$

$$\begin{bmatrix} Q_k & Y_k^T H^T(\theta(k))_{[i,:]} \\ H(\theta(k))_{[i,:]} Y_k & 1 \end{bmatrix} \geq 0, \; i = 1, \ldots, 4 \tag{3.5.8d}$$

$$Q_k \leq \mathcal{Q}_0^* \tag{3.5.8e}$$

admits a solution $\forall \, k > 0$ *and the receding horizon state-feedback control law*

$$[\omega_R(k), \omega_L(k)]^T = -T^{-1} T_{FL}(\theta(k)) Y_k^* Q_k^{*-1} \xi(k); \tag{3.5.9}$$

solves the constrained Problem 1.

Proof. First, the LMI (3.5.8d) is equivalent to the polyhedral input constraint $\mathcal{U}(\theta(k))$ [99, 100]. Therefore, the receding-horizon control law $u(k) = -Y_k^* Q_k^{*-1} \xi(k)$, obtained solving the optimization problem (3.5.8a)-(3.5.8d) with a constant input constraint set $\mathcal{U}(\theta(k)) = \mathcal{U}$, $\forall \, k$, solves the regulation problem for (3.5.1), with $\mathcal{E}_k = \{\xi \in \mathbb{R}^2 : \xi^T Q_k^{*-1} \xi \leq 1\}$ the CI region [100]. Conversely, if $\theta(k)$ varies at each time instant, it is no longer ensured that the one-step evolution $\xi(k+1)$ represents an admissible initial condition for (3.5.8a)-(3.5.8d) at $k+1$. However, such a drawback can be overcome by forcing the invariant region $\mathcal{E}_k$ to be included in the worst-case domain of attraction $\mathcal{E}_0$ (obtained solving (3.5.5)). This translates into the additional LMI constraint (3.5.8e) ensuring recursive feasibility of (3.5.8) $\forall \, k$, i.e., the offline solution $[Q_0^*, Y_0^*, \gamma_0^*]$ is always an admissible, although not optimal, solution of the problem. Finally, by resorting to the same arguments used in the proof of Proposition 3.2, the control law (3.5.9) solves Problem 1 $\square$

Remark 3.3. At each k, the control law (3.5.9) exploits the input constraint set $\mathcal{U}(\theta) \supset \underline{\mathcal{U}}$ corresponding to the mapping of the differential-drive input constraint $\mathcal{U}_d$ into the feedback linearized robot model for the current orientation $\theta(k)$. Consequently $\forall\, k$, the input constraint set (3.5.8d) is a subset of (3.5.5d). $\square$

Finally, the following algorithm summarizes all the above developments.

```
Algorithm 1 Reference Tracking - RHC Strategy
Configuration:
    • Robot parameters and planner radius: $\bar{\Omega}, R, D$ and $d_r$
    • LQ cost matrices and B point distance: $Q_x, Q_u, b > 0$
Offline:
    1. Compute $\underline{\mathcal{U}}(r_u)$ as in (3.4.1) and solve opt. (3.5.5) with $\xi(0) = r(0) - p_c(0)$. Store $Q_0^*$
Online ($\forall\, k$):
    1. Obtain $r(k)$ from the trajectory planner, compute $T_{FL}(\theta)$ using (2.2.11), and $H(\theta)$ using (3.3.1)
    2. Solve opt. (3.5.8) with $\xi(k) = r(k) - p_c(k)$
    3. Compute $\omega_R(k), \omega_L(k)$ using (3.5.9)
```

3.6 Experimental Results

Here, the experimental results, obtained using a Khepera IV robot, are presented to show the effectiveness of the proposed tracking controller and compare it with the solutions developed in [18, 19, 22, 25, 102]. In particular, [18] and [25] are two popular linear MPC-based controllers developed on the constrained and unconstrained robot's model, respectively; [19]

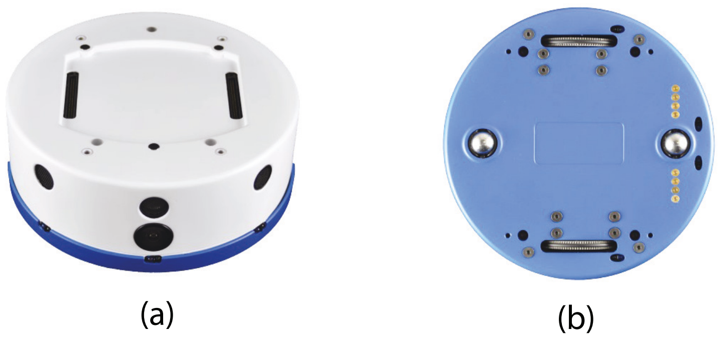


Figure 3.3: Khepera IV differential-drive robot: top (a) and bottom (b) view.

is the landmark control solution for unconstrained robots controlled via feedback linearization; [22] is a popular unconstrained Lyapunov-based solution; [102] is the preliminary RHC constrained controller developed by the same authors.

3.6.1 Setup:

The Khepera IV (Fig. 3.3) is a differential-drive robot produced by the K-Team Corporation. This wheeled robot presents two independent wheels driven by two DC brushed motors equipped with incremental encoders. Each wheel has a radius $R = 0.021\,m$, and the rear axis length is $D = 0.0884\,m$. Each wheel has a maximum speed of 1200 $steps/s$ which corresponds to a linear velocity of $0.813\,m/s$, and an angular velocity of 38.7143 rad/s. Nevertheless, in all the performed experiments, the maximum speed of the robot's wheels has been limited to $|\omega_R| = |\omega_L| \leq 250\,steps/s$ $(8.0655\,rad/s)$ to avoid unmodeled dynamic effects and wheel slips. A sampling time $T_s = 0.15\,s$ is considered and the robot's position has been estimated resorting to odometric calculations [89]. Moreover, Bluetooth has been used to communicate between the robot and a Linux Ubuntu 16 computer, equipped with an Intel Core I7 8750H processor, 16Gb of RAM, and running Matlab R2020a.

3.6.2 Configuration of the proposed controller:

To implement the above algorithm, the following parameters have been used: $b = 0.1m$, $Q_x = I_2$ and $Q_u = 0.001 I_2$. Moreover, the LMIs optimizations (3.5.5) and (3.5.8) have been solved using the Matlab built-in LMI solver *"mincx"*.

3.6.3 Configuration of the competitor schemes:

Each competitor scheme has been tuned to obtain the best tracking results in the conducted experiments. Since the algorithms in [19, 22] do not take into account the wheels' velocity limitations, a saturation is enforced whenever necessary. For further details on the meaning of the following parameters one car refer to [18, 19, 22, 25, 102]:

- the MPC solutions in [18], [25] have been tuned to use a prediction horizon $N_p = 10$ with a cost function as in (3.5.4), where $Q_x = \begin{bmatrix} I_2 & 0 \\ 0 & 0.05 \end{bmatrix}$ and $Q_u = 0.001 I_2$. The optimization problem resulting from [18] has been solved in Matlab using the built-in quadratic programming solver *"quadprog"*. Whereas, the unconstrained optimization in [25] has been analytically solved. In addition, a reference matrix $A_r = 0.65 I_3$ has been selected for the algorithm of [25].

- The proportional-derivative controller in [19] has been implemented with $k_{p1} = k_{p2} = 1$, and $k_{d1} = k_{d2} = 0.7$.

- The Lyapunov-based control law in [22] has been implemented with $k = 5$, $k_x = 0.5$, $k_s = 5$, $n = 2$, $a = 7$.

- The receding-horizon controller in [102] has been implemented using the same parameters of the proposed controller.

3.6.4 Evaluation of the tracking performance:

The tracking performance have been evaluated by resorting to four error-based indices described in [26]. By defining the instantaneous tracking error as

$$e(t) = \sqrt{(x_r(t) - x_c(t))^2 + (y_r(t) - y_c(t))^2},$$

and by denoting with T_f the duration of the experiment, the used indices are: (i) integral absolute error (IAE) ($\int_0^{T_f} |e(t)| dt$), (ii) integral square error (ISE) ($\int_0^{T_f} e(t)^2 dt$), (iii) integral time-weighted absolute error (ITAE) ($\int_0^{T_f} t|e(t)| dt$), (iv) integral time squared error (ITSE) ($\int_0^{T_f} t e(t)^2 dt$).

Trajectories: Two trajectories $r(k)$ have been considered:
(i)- An eight-shaped trajectory, where

$$r(t) = \begin{bmatrix} x_r(t) \\ y_r(t) \end{bmatrix} = \begin{bmatrix} 0.6 \sin(\frac{t}{3.5}) \\ 0.6 \sin(\frac{t}{7}) \end{bmatrix}, \quad t \in [0, 44]$$

(ii)- A circular trajectory, where

$$r(t) = \begin{bmatrix} x_r(t) \\ y_r(t) \end{bmatrix} = \begin{bmatrix} 0.3 \sin(\frac{t}{3.8}) \\ 0.3 \cos(\frac{t}{3.8}) \end{bmatrix}, \quad t \in [0, 24]$$

Moreover, the vehicle initial's state $[p_c^T(0),\, \theta(0)]^T$ is $[0.6,\, 0,\, \pi]^T$ for the eight trajectory, and $[0.3,\, 0.3,\, \pi]^T$ for the circular reference.

3.6.5 Results:

The obtained results are collected in Figs. 3.4-3.6 and Tables 3.1-3.4. Moreover, the additional Table 3.5 shows an averaged comparison based on all the performed experiments. For the interested reader, videos of the performed experiments can be found at the following web-link: https://tinyurl.com/mhsejcny
Two experiments have been performed:
Experiment 1: The operating scenario prescribes that the reference trajectory and its first and second derivatives are available for any desired prediction horizon. Results and comparisons are shown in Fig. 3.4 and Table 3.1 (for the eight-shaped trajectory) and in Fig. 3.5 and Table 3.2 (for the circular trajectory). The obtained results show that,

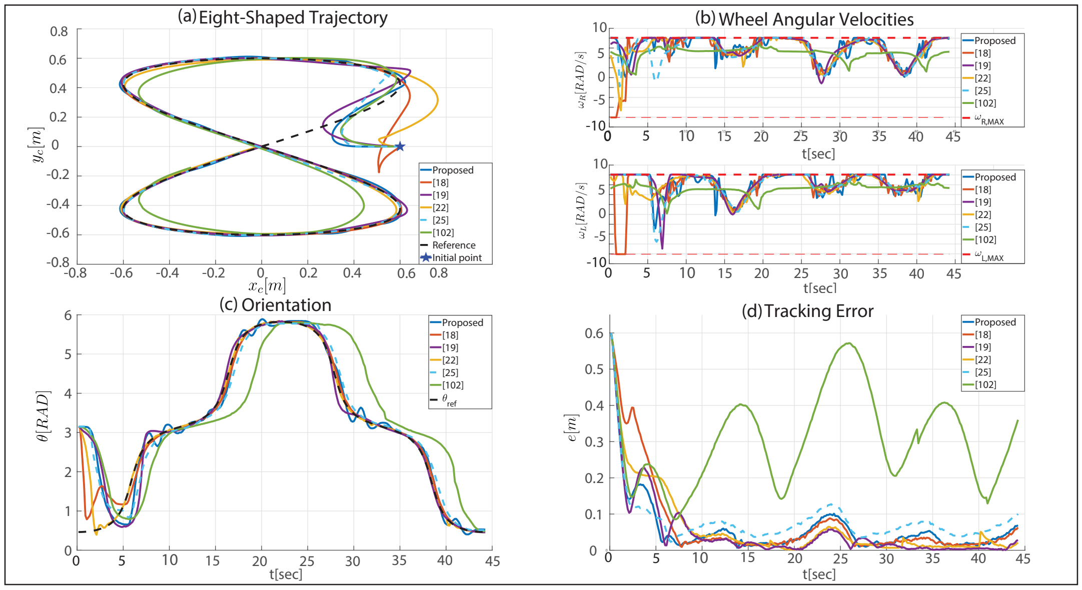


Figure 3.4: Eight-shaped trajectory - Experiment 1

for both trajectories, the proposed controller achieves better performance when compared to [18, 25, 102]. On the other hand, the solutions [19, 22] show slightly better performance with respect to some indices (i.e., IAE, ITAE, ITSE) when the eight trajectory is under investigation. Indeed, the proposed controller only uses the current reference, while the competitors exploit trajectory predictions or its derivatives.
Experiment 2: We consider a scenario where an onboard path planner computes the trajectory in real-time. In this setup, only the current reference point $r(k)$ is available, and there is no information on the trajectory derivatives The obtained results are shown in

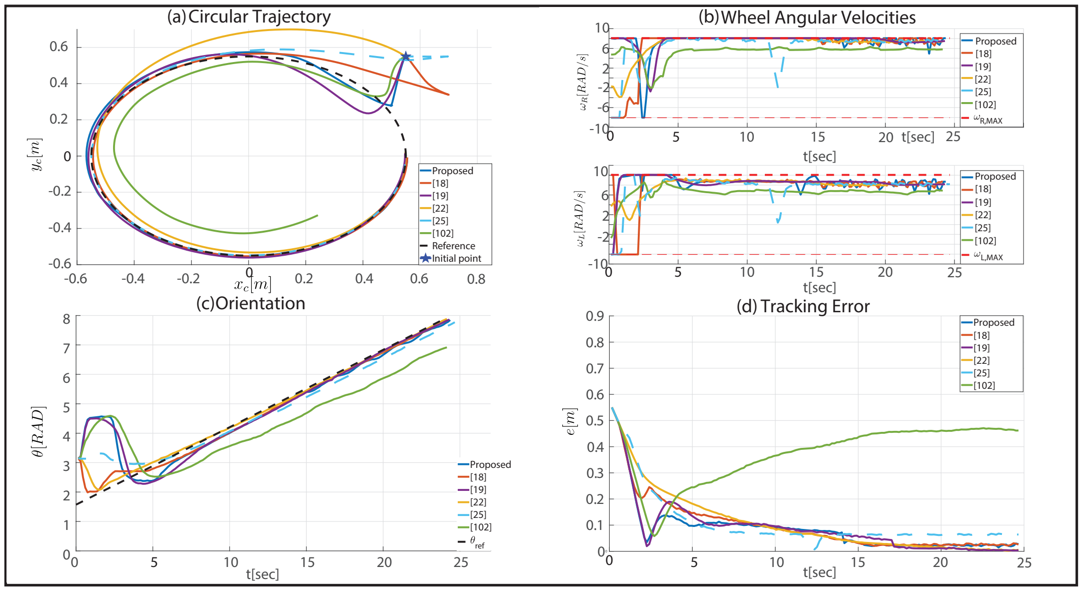


Figure 3.5: Circular trajectory - Experiment 1

Fig. 3.6 and Table 3.3 (for the eight-shaped trajectory) and in Fig. 3.7 and Table 3.4 (for the circular trajectory). First, the proposed solution and the strategies in [22,102] perform as in

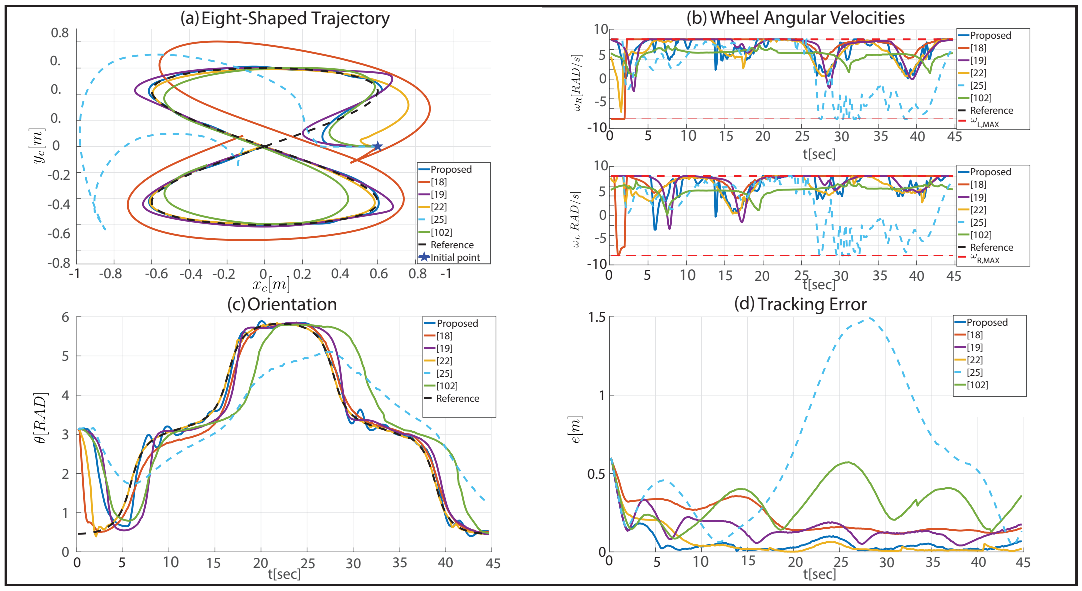


Figure 3.6: Eight-shaped trajectory - Experiment 2

Experiment 1. This because such strategies do not use a preview of the reference trajectory nor its derivatives. Moreover, the proposed approach outperforms all the competitor schemes, except [22] that shows slightly better tracking performance (according to the indices IAE, ITAE, ITSE) when the eight trajectory is considered. With respect to Experiment 1, the predictive approaches in [18], [25] worsen because only one-step predictions can be exploited.

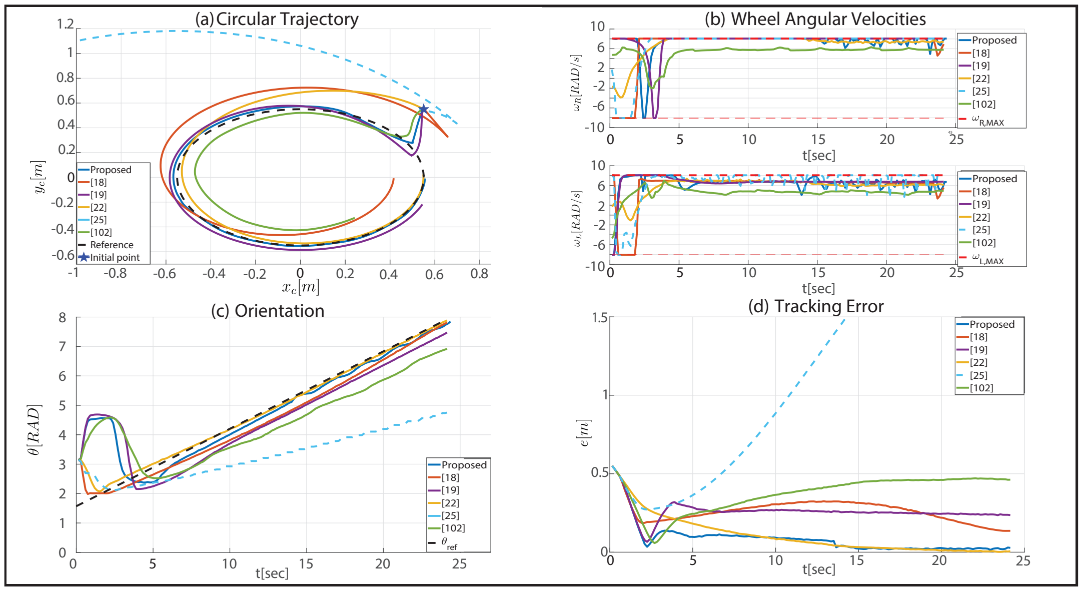


Figure 3.7: Circular trajectory - Experiment 2

Moreover, the solution in [19] is affected by the absence of information about the trajectory's derivatives.

3.6.6 Overall performance:

In both experiments, the proposed controller can correctly track both trajectories while fulfilling the vehicle's velocity constraints. Moreover, the average CPU time required by the proposed control algorithm is $4.1\,ms$, which is much lower than the sampling time. Note that the proposed controller outperforms, in any tracking performance index, the preliminary solution [102]. The latter can be explained by looking at, e.g., the angular velocities plotted in Fig. 3.4.b, where it is evident that the proposed solution can make use of the maximum robot's velocity, while [102] always remains very conservative. Finally, from the summary Table 3.5, obtained averaging the results in all the experiments, it is interesting to note that for almost all the indices the proposed strategy performs better than the competitor schemes.

3.7 Conclusions

In this chapter, a novel receding-horizon reference tracking controller for input constrained differential-drive robots has been presented. The proposed strategy has been designed by properly leveraging two main ingredients: a feedback linearization technique and a receding-horizon control framework. The obtained tracking controller has the peculiar capability of

Table 3.1: Experiment 1 (Eight Trajectory): tracking performance.

	IAE	**ISE**	**ITAE**	**ITSE**
Proposed	2,502	0,390	38,026	2,248
[**18**]	3,084	0,761	36,805	3,436
[**19**]	2,002	0,394	16,952	1,042
[**22**]	2,163	0,454	17,454	1,145
[**25**]	3,467	0,467	63,512	4,711
[**102**]	13,505	4,765	317,567	113,990

Table 3.2: Experiment 1 (Circular Trajectory): tracking performance.

	IAE	**ISE**	**ITAE**	**ITSE**
Proposed	2,160	0,390	15,913	1,305
[**18**]	2,298	0,462	14,397	1,330
[**19**]	2,121	0,408	13,831	1,269
[**22**]	3,692	0,781	30,919	4,149
[**25**]	2,634	0,542	21,046	1,855
[**102**]	8,989	3,625	123,516	53,107

Table 3.3: Experiment 2 (Eight Trajectory): tracking performance.

	IAE	**ISE**	**ITAE**	**ITSE**
Proposed	2,502	0,390	38,026	2,248
[**18**]	9,452	2,458	162,222	31,162
[**19**]	6,631	1,247	122,173	17,206
[**22**]	2,163	0,454	17,454	1,145
[**25**]	27,730	26,152	707,647	707,830
[**102**]	13,505	4,765	317,567	113,990

Table 3.4: Experiment 2 (Circular Trajectory): tracking performance.

	IAE	**ISE**	**ITAE**	**ITSE**
Proposed	2,160	0,390	15,913	1,305
[**18**]	6,298	1,768	72,313	19,100
[**19**]	6,220	1,691	73,415	18,607
[**22**]	3,692	0,781	30,919	4,149
[**25**]	29,949	50,840	486,219	933,627
[**102**]	8,989	3,625	123,516	53,107

efficiently dealing with state-dependent input constraints acting on the feedback linearized

Table 3.5: Average Tracking performance Indices.

	IAE	**ISE**	**ITAE**	**ITSE**
Proposed	2,331	0,390	26,970	1,776
[**18**]	5,283	1,362	71,434	13,757
[**19**]	4,243	0,935	56,593	9,531
[**22**]	2,927	0,617	24,186	2,647
[**25**]	15,945	19,500	319,606	412,006
[**102**]	11,247	4,195	220,541	83,549

vehicle model while ensuring recursive feasibility, stability, and velocity constraints fulfillment. Extensive experimental results and comparisons have been carried out to highlight the features and advantages of the proposed tracking controller.

4 A Set-Theoretic Control Approach to the Trajectory Tracking Problem for Input-Output Linearized Wheeled Mobile Robots

In this chapter, an extension of trajectory tracking RHC strategy, presented in Chapter 3, for input-constrained differential-drive robots. The proposed solution adapts the Set-Theoretic Receding Horizon Control (ST-RHC) algorithm introduced in [54] to deal with the time-varying constraints acting on the input-output feedback linearized model of the robot. In particular, we characterize the linearized error dynamics as a constrained linear system subject to a bounded disturbance depending on the reference trajectory. Worst-case arguments on the disturbance and input constraint sets are leveraged to offline design a stabilizing feedback controller associated with the smallest robust control invariant region. Also, a family of robust one-step controllable sets is computed to enlarge the controller's tracking domain and allow large initial tracking errors. Online, the conservativeness of the offline solution is mitigated by exploiting the knowledge of the current robot's orientation and trajectory-dependent disturbance. The effectiveness of the resulting RHC strategy is validated by means of laboratory experiments. Although the proposed solution borrows from [84] (Chapter 3) the worst-case characterization of the input constraint set acting on the feedback linearized differential-drive robot (see Sec.3), there are some key differences between the two strategies. The approach in [84] deals with a waypoint tracking problem, while here a more challenging and general trajectory tracking problem is considered. The consequence of the above is that unlike [84], the feedback-linearized vehicle's error dynamics are now subject to an additional bounded disturbance term related to the desired reference trajectory. Moreover, differently from [84], the proposed solution is capable of exploiting a larger set of information about the reference trajectory (e.g., reference timing law and its derivatives), which results in improved tracking performance (see Table 3.1). In [84], the waypoint tracking controller is developed by extending the LMI-based receding horizon control framework developed in [99]. On the other hand, the proposed robust controller is developed by extending the robust set-theoretic model predictive control paradigm developed in [54]. Finally, unlike [84], the proposed solution ensures that the tracking error trajectory is uniformly ultimately bounded, in a finite number of steps, in the smallest robust control invariant region.

4.1 Preliminaries and Problem Formulation

4.1.1 Discrete-time Robot's Modeling:

Let's consider a discretization of the nonlinear differential-drive kinematic model (2.1.3).

$$\begin{array}{rcl} x(k+1) & = & x(k) + T_s \frac{R}{2} \left(\omega_R(k) + \omega_L(k)\right) \cos(\theta(k)) \\ y(k+1) & = & y(k) + T_s \frac{R}{2} \left(\omega_R(k) + \omega_L(k)\right) \sin(\theta(k)) \\ \theta(k+1) & = & \theta(k) + T_s \frac{R}{D} (\omega_R(k) - \omega_L(k)) \end{array} \quad (4.1.1)$$

subject to the input-constraints (2.1.6).

Using transformation (2.1.7), the differential-drive discrete kinematics(4.1.1) can be recast into a discrete-time unicycle model.

$$\begin{array}{rcl} x(k+1) & = & x(k) + T_s v(k) \cos(\theta(k)) \\ y(k+1) & = & y(k) + T_s v(k) \sin(\theta(k)) \\ \theta(k+1) & = & \theta(k) + T_s \omega(k) \end{array} \quad (4.1.2)$$

which is subject to the rhombus-like constraints (2.4).

4.1.2 Problem formulation

Consider a bounded and smooth 2D-trajectory described in terms of Cartesian position $(x_r(t), y_r(t))$, velocity $(\dot{x}_r(t), \dot{y}_r(t)$, and acceleration $(\ddot{x}_r(t), \ddot{y}_r(t))$, where $t \in \mathbb{R}^+$. Then, the robot's reference orientation $\theta_r(t)$, longitudinal velocity $v_r(t)$ and angular velocity $\omega_r(t)$ are [19]:

$$\begin{array}{rcl} \left[\begin{array}{c} v_r(t) \\ \omega_r(t) \end{array} \right] & = & \left[\begin{array}{c} \sqrt{\dot{x}_r(t)^2 + \dot{y}_r(t)^2} \\ \frac{\ddot{y}_r(t)\dot{x}_r(t) - \ddot{x}_r(t)\dot{y}_r(t)}{\dot{x}_r(t)^2 + \dot{y}_r(t)^2} \end{array} \right] \\ \theta_r(t) & = & \text{ATAN}_2\left(\dot{y}_r(t), \dot{x}_r(t)\right) \end{array} \quad (4.1.3)$$

Remark 4.1. It is easy to show that the forward Euler discretization of (4.1.3) represents a solution for the discrete-time unicycle model (4.1.2). Moreover, When the tangent velocity $v_r(k)$ is null for some $t \geq 0$, neither the nominal angular velocity nor the nominal orientation can be computed via (4.1.3). Different solutions to such a drawback already exist in literature. For example, [19] proposes to use higher-order differential information about $x_r(k)$ and $y_r(k)$ to determine a consistent reference orientation and angular velocity command. Alternatively, one can keep the same orientation $\theta_r(k) = \theta_r(k-1)$, and use de L'Hôpital analysis to compute $\omega_r(k)$. $\square$

Problem 2. *Consider the input-constrained differential-drive robot model* (4.1.1) *and a bounded and smooth trajectory* $q_r(k) = [x_r(k), y_r(k), \theta_r(k)]^T$ *obtained by means of a forward Euler discretization of* (4.1.3)*, with* $k \in \{0, 1, \dots, k_f\}$*. Design a trajectory tracking control law* $[\omega_R(k), \omega_L(k)]^T = \phi(k, q(k), q_r(k)) \in \mathcal{U}_d$ *such that the tracking error* $\tilde{q}(k) = q(k) - q_r(k)$ *remains bounded* $\forall k \geq 0$.

4.2 Proposed Solution

In this chapter, first, the feedback-linearized tracking-error dynamics are derived, and its time-varying input constraints are discussed. Then, the ST-RHC scheme (see Chapter 4.1) is tailored to solve the considered problem.

Linearized Vehicle Model via Feedback Linearization:
In 3, it has been shown that the discrete-time unicycle model (4.1.2) is feedback linearizable. Consider a scalar $b > 0$ and two new outputs

$$z(k) = \begin{bmatrix} x(k) + b\cos\theta(k) \\ y(k) + b\sin\theta(k) \end{bmatrix} \in \mathbb{R}^2 \tag{4.2.1}$$

representing the coordinates of an external point $\mathbf{B}$ displaced at a distance b from the robot's center of mass. Then, the following state-feedback law

$$\begin{bmatrix} v(k) \\ \omega(k) \end{bmatrix} = T_{FL}(\theta) \begin{bmatrix} u_1(k) \\ u_2(k) \end{bmatrix}, \quad T_{FL}(\theta) = \begin{bmatrix} \cos\theta & \sin\theta \\ \frac{-\sin\theta}{b} & \frac{\cos\theta}{b} \end{bmatrix} \tag{4.2.2}$$

recasts the unicycle model (4.1.2) into the following two-single discrete-time integrator model

$$z(k+1) = Az(k) + Bu(k), \; A = I_{2\times 2}, \; B = T_s I_{2\times 2} \tag{4.2.3a}$$

$$\theta(k+1) = \theta(k) + T_s \frac{-\sin\theta(k)u_1(k) + \cos\theta(k)u_2(k)}{b} \tag{4.2.3b}$$

where $u(k) = [u_1(k), \, u_2(k)]^T \in \mathbb{R}^2$ are the control inputs of the feedback-linearized robot model. Note that (4.2.3b) defines a nonlinear internal dynamics decoupled from (4.2.3a).

4.2.1 Input-Ouput Linearized Error Dynamics:

Given $q_r(k)$, the reference trajectory for feedback linearized robot's model can be computed by means of the transformation (4.2.1) i.e.,

$$z_r = \begin{bmatrix} x_r + b\cos\theta_r \\ y_r + b\sin\theta_r \end{bmatrix} \tag{4.2.4}$$

Thus, the feedback-linearized tracking-error dynamics is given by:

$$\begin{aligned} \tilde{z}(k+1) &= z(k+1) - z_r(k+1) = \\ &= Az(k) + Bu(k) - Az_r(k) - Bu_r(k) \\ &= A\tilde{z}(k) + Bu(k) + d(k) \end{aligned} \tag{4.2.5}$$

where $\tilde{z}(k) = z(k) - z_r(k)$, is the linearized system tracking error,

$$u_r(k) = T_{FL}^{-1}(\theta_r(k))[v_r(k), \omega_r(k)]^T \tag{4.2.6}$$

is the reference input associated to the trajectory, and $d(k) = -Bu_r(k)$.

Remark 4.2. If the reference trajectory is bounded, then $d(k)$ is a bounded disturbance with $d(k) \in \mathcal{D} \subset \mathbb{R}^2$. Moreover, if $d(k)$ is a-priori known, then $\mathcal{D}$ can be over-approximated with a ball of radius r_d, i.e,

$$\mathcal{D} = \{d \in \mathbb{R}^2 : d^T Q_d^{-1} d \leq 1\},\ Q_d = r_d^2 I_{2\times 2} \tag{4.2.7}$$

□

Lemma 4.1. *[93] If a control law $u(k)$ stabilizes* (4.2.5)*, the point* **B** *tracks any reference trajectory with a bounded internal dynamics. Consequently, also the tracking error $\tilde{q}(k)$ is bounded.* □

Lemma 4.2. *[84, Section III.B] The set of admissible inputs for the feedback linearized error dynamics* (4.2.5) *is the following time-varying and orientation-dependent polyhedral* *set*

$$\mathcal{U}(\theta) = \{[u_1, u_2]^T \in \mathbb{R}^2 : \, H(\theta)\,[u_1, u_2]^T \leq 1\},$$

$$H(\theta) = H_d T^{-1} T_{FL}(\theta) =$$

$$= \begin{bmatrix} \frac{D\sin\theta - 2\cos\theta b}{2\overline{\Omega}Rb} & \frac{-D\cos\theta - 2\sin\theta b}{2\overline{\Omega}Rb} \\ \frac{-D\sin\theta - 2\cos\theta b}{2\overline{\Omega}Rb} & \frac{D\cos\theta - 2\sin\theta b}{2\overline{\Omega}Rb} \\ \frac{-D\sin\theta + 2\cos\theta b}{2\overline{\Omega}Rb} & \frac{D\cos\theta + 2\sin\theta b}{2\overline{\Omega}Rb} \\ \frac{D\sin\theta + 2\cos\theta b}{2\overline{\Omega}Rb} & \frac{-D\cos\theta + 2\sin\theta b}{2\overline{\Omega}Rb} \end{bmatrix} \tag{4.2.8}$$

which admits the following worst-case internal and circular approximation:

$$\hat{\mathcal{U}} = \bigcap_{\forall\theta} \mathcal{U}(\theta) = \{u \in \mathbb{R}^2 \,| u^T Q_u^{-1} u \leq 1\}, \; Q_u = r_u^2 I_{2\times 2} \tag{4.2.9}$$

where $r_u = \frac{2\overline{\Omega}Rb}{\sqrt{4b^2 + D^2}}$.

4.2.2 Proposed Receding Horizon Controller:

Here, the control scheme presented in Chapter 4.1 is customized to solve constrained trajectory tracking problem starting from the tracking error dynamics (4.2.5) and its worst-case input constraint set (4.2.9).

Assumption 4.1. The set $B\hat{\mathcal{U}}$ contains $\mathcal{D}$, i.e., the difference set $B\hat{\mathcal{U}} \ominus \mathcal{D} \neq \emptyset$. □

Remark 4.3. Assumption 4.1 ensures that the controller has sufficient authority over the disturbance caused by the reference trajectory. It can be offline verified, and it imposes a feasibility condition for the reference trajectory. □

The linearized robot error dynamics (4.2.5) are subject to the time-varying input constraint $\mathcal{U}(\theta(k))$ and bounded disturbance $d(k) = -Bu_r(k) \in \mathcal{D}$. Consequently, to perform the offline phase of the ST-MPC scheme, the only possibility is to consider the worst-case input constraint $\hat{\mathcal{U}} \subset \mathcal{U}(\theta(k)), \forall\theta(k)$ (see (4.2.9)) to compute the RCI region $\mathcal{T}^0$ and a family of ROSC sets $\mathcal{T}^i$ that are valid $\forall\theta(k)$. Nevertheless, online, such a source of conservativeness will be mitigated exploiting the knowledge of $\theta(k)$ to determine the actual input constraint $\mathcal{U}(\theta(k)), \forall\, k$.

The following propositions show that for the linearized robot error dynamics, the sets $\mathcal{T}^0$ and ROSC sets $\mathcal{T}^i$ can be analytically computed.

Proposition 4.3. *Consider the model* (4.2.5) *under the constraint* $u(k) \in \hat{\mathcal{U}}$ *and disturbance* $d(k) \in \mathcal{D}$. *The terminal set* $\mathcal{T}^0 = \mathcal{D}$ *is the smallest RCI under the control law*

$$u(k) = -B^{-1}\tilde{z}(k) \tag{4.2.10}$$

Proof. First, under Assumption 4.1, $\forall\, \tilde{z} \in \mathcal{D}$, the control law $u = -B^{-1}\tilde{z} \in \hat{\mathcal{U}}$, or equivalently that $B^{-1}\mathcal{D} \subset \hat{\mathcal{U}}$. Indeed, by noticing that B is invertible, $B^{-1}\mathcal{D} \subset \hat{\mathcal{U}} \iff -B^{-1}Q_d(-B^{-1})^T \leq Q_u \iff \frac{r_d}{T_s} \leq r_u$. By cross multiplying the last inequality by B^{-1} on the left and $(B^{-1})^T$ on the right, we obtain $BB^{-1}Q_d(B^{-1})^T B^T \leq BQ_uB^T \iff Q_d \leq BQ_uB^T \iff r_d \leq T_s r_u \iff \mathcal{D} \subseteq B\hat{\mathcal{U}}$. Now, if $d(k) = 0, \forall\, k$, and $u(k) = -B^{-1}\tilde{z}(k)$ we have that $\tilde{z}(k+1) = A\tilde{z}(k) + B(-B^{-1})\tilde{z}(k) = \tilde{z}(k) - \tilde{z}(k) = 0_2$. Consequently, for any disturbance realization $d(k) \in \mathcal{D}$, it is also true that the one-step evolution is bounded by $\mathcal{D}$ and that $\mathcal{T}^0 = \mathcal{D}$ is the smallest RCI set. □

Proposition 4.4. *Consider the model* (4.2.5) *under the constraint* $u(k) \in \hat{\mathcal{U}}$ *and disturbance* $d(k) \in \mathcal{D}$. *Given a target ball set* $\mathcal{T}^{i-1} \subset \mathbb{R}^2$ *of radius* $r_{i-1} > 0$, *the set of states ROSC to* $\mathcal{T}^{i-1}$ *is*

$$\mathcal{T}^i = \{\tilde{z} \in \mathbb{R}^2 : \tilde{z}^T Q_i^{-1}\tilde{z} \leq 1\}, \quad Q_i = r_i^2 I_{2\times 2} \tag{4.2.11}$$

$$r_i = r_{i-1} - r_d + T_s r_u \tag{4.2.12}$$

Proof. The set $\mathcal{T}^i$ ROSC to $\mathcal{T}^{i-1}$ for (4.2.5) can be computed as $\mathcal{T}^i = ((\mathcal{T}^{i-1} \ominus \mathcal{D}) \oplus (-B\hat{\mathcal{U}}))A$, see [96, Sec. 11.3.2]. Since $\mathcal{D}$, $\hat{\mathcal{U}}$, $\mathcal{T}_{i-1}$ are ball sets and $A = I_{2\times 2}$, $B = T_s I_{2\times 2}$, then also $\mathcal{T}^i$ is a ball of radius r_i computed as in (4.2.12) (see Property 2.1), concluding the proof. □

Remark 4.4. *Given the results of Propositions 4.3-4.4, it is possible to solve Problem 2 by implementing the ST-RHC controller detailed in Chapter 4.1, where*

- (2.4.8) *is replaced by* (4.2.5)*;*
- $K = B^{-1}$, $\mathcal{T}^0 = \mathcal{D}$ *as in Proposition 4.3;*
- $\mathcal{U} = \hat{\mathcal{U}}$ *(i.e., the worst-case input constraint set* (4.2.9)*);*

- $\{\mathcal{T}^i\}_{i=1}^N$ *are recursively computed as in Proposition 4.4.*

- *the set-membership index $i(k)$ is computed as*

$$i(k) := \min\{i : \tilde{z}(k)^T Q_i^{-1} \tilde{z}(k) \leq 1\} \tag{4.2.13}$$

□

The solution described in Remark 4.4 is conservative because it uses the worst-case input constraint set $\hat{\mathcal{U}} \subset \mathcal{U}(\theta)$, $\forall\theta$ and it assumes that $d(k) = -Bu_r(k)$ is an unknown disturbance. However, online and for any k, both $d(k)$ and $\mathcal{U}(\theta)$ can be determined starting from the reference trajectory $q_r(k)$ and robot's orientation $\theta(k)$, respectively. By taking advantage of such information, the following proposition describes a non-conservative control strategy solving Problem 2.

Theorem 4.5. *For any $\tilde{z}(0) \in \bigcup_{i=0}^N \mathcal{T}^i$, the tracking ST-RHC strategy described in Algorithm 2 provides a solution to Problem 2.*

Proof. The proof can be divided in four parts:
(I) - Opt (4.2.14) *always admits a solution.* First, by construction, the optimization (5.4.9) is feasible for any $d \in \mathcal{D}$ (see Property 2.1). Consequently, (4.2.14) is admissible because the input constraint set is enlarged (i.e., $\mathcal{U}(\theta) \supset \hat{\mathcal{U}}$, $\forall\theta(k)$) and the conservative Minkowski difference is replaced by the actual value of $d(k)$.
(II) - Opt (4.2.16) *is always feasible and $\mathcal{T}^0$ is RCI under $u(k) = -B^{-1}\tilde{z}(k) + \hat{u}_r(k)$.* Indeed, $\hat{u}_r(k) = 0$ is always a feasible solution that corresponds to the terminal control law for which $\mathcal{T}^0$ is RCI for any $d(k) \in \mathcal{D}$, see Proposition 4.3. On the other hand, the opt. (4.2.16) selects the optimal $\hat{u}_r(k)$, compatible with the input constraint $\mathcal{U}(\theta)$, that compensates (totally or partially) for the disturbance realization $d(k) = -B\hat{u}_r(k)$. Consequently, if $\tilde{z}(k) \in \mathcal{T}^0$, then $\tilde{z}(k+j) \in \mathcal{T}^0$, $\forall\, j \geq 1$ and $u(k+j) \in \mathcal{U}(\theta), \forall j \geq 0$.
(III) - Feasibility and Uniformly Ultimately Boundedness (UUB). Recursive feasibility trivially holds since, by construction, both (4.2.14) and (4.2.16) always admits a feasible solution compatible with the given constraints and worst-case disturbance realization. Consequently, starting from any admissible initial tracking error $\tilde{z}(0) \in \bigcup_{i=0}^N \mathcal{T}^i$, the set-membership index $i(k)$ monotonically decreases, at each k, until $i = 0$ is reached. Consequently, the tracking

error of the feedback linearized model reaches the RCI set $\mathcal{T}_0$ in at most N steps where it is UUB under the effect of (4.2.15).

(IV) - Bounded tracking error. First, $u(k)$ computed by Algorithm 2 stabilizes the feedback linearized error dynamics. Therefore, given the result of Lemma 4.1, the input transformation (4.2.2), the control law (4.2.17) solves the considered reference tracking problem with a bounded tracking error $\tilde{q}(k)$, concluding the proof. $\square$

Remark 4.5. *As prescribed by Algorithm 2, the quadratically constrained quadratic program* (4.2.14) *must be solved, at most, for the first N steps (until $\mathcal{T}^0$ is reached). Afterwards, the linearly constrained quadratic program* (4.2.16) *is solved.*

Algorithm 2 Tracking Set-Theoretic Receding Horizon Controller (T-ST-RHC)

```
Offline:
1: Set $\mathcal{U} = \hat{\mathcal{U}}$, $K = B^{-1}$, $\mathcal{T}^0 = \mathcal{D}$; Build $\{\mathcal{T}^i\}_{i=1}^N$ using (4.2.11); Store $\{\mathcal{T}^i\}_{i=0}^N$.
Online:
1: Measure $x(k)$, $y(k)$, and $\theta(k)$
2: Compute $\tilde{z}(k) = z(k) - z_r(k)$, with $z(k)$ as in (4.2.1), $z_r(k)$ as in (4.2.4);
3: Compute $\mathcal{U}(\theta)$ as in (4.2.8) and $u_r(k)$ as in (4.2.6);
4: Find $i(k)$ as in (4.2.13);
5: if $i(k) > 0$, then
       $u(k) = \arg\min_{u} J(x,u) \; s.t.$    (4.2.14a)
       $A\tilde{z}(k) + Bu - Bu_r(k) \in \mathcal{T}^{i(k)-1}$    (4.2.14b)
       $u \in \mathcal{U}(\theta)$    (4.2.14c)
6: else
       $u(k) = -B^{-1}\tilde{z}(k) + \hat{u}_r(k), \quad \text{where}$    (4.2.15)
       $\hat{u}_r(k) = \arg\min_{\hat{u}_r} \|\hat{u}_r - u_r(k)\|_2^2 \quad s.t.$    (4.2.16a)
       $-B^{-1}\tilde{z}(k) + \hat{u}_r \in \mathcal{U}(\theta)$    (4.2.16b)
7: end if
8: Compute
       $[\omega_R(k), \omega_L(k)]^T = T^{-1}T_{FL}u(k)$    (4.2.17)
   and apply it to the robot; $k \leftarrow k+1$, go to 1;
```

4.3 Experimental Results

The proposed trajectory tracking control has been validated by means of hardware-in-the-loop laboratory experiments carried out using a Khepera IV differential-drive robot. A demo of the hereafter presented experiment can be found at the following web link: `https://youtu.be/A0Tlbgr08tY`. The robot parameters are $R = 0.021\ [m]$, $D = 0.0884\ [m]$, the maximum velocity is set to $\overline{\Omega} = 10\ [rad/sec]$, and $T_s = 0.15\ [sec]$ The robot's pose vector has been estimated using the wheels encoder's measurements and odometry calculations as outlined in [89]. Algorithm 2 has been implemented on a Windows 10 computer equipped with an *Intel i7-8750H* processor and Matlab R2022b. The optimizations (4.2.14) and (4.2.16) have been solved using the Matlab's functions *fmincon* and *qaudprog*, respectively. Moreover, a wireless TCP channel has been used for communicating with the robot, see Fig. 4.1.

The performance of the proposed tracking algorithm has been compared with four alternative strategies: (i) the RHC strategy developed by the same authors in [84], (ii) the Lyapunov-based controller in [22], (iii) the unconstrained linear MPC solution in [25], (iv) the dynamic feedback-linearization controller in [89]. All the competitor schemes have been configured using the same parameters described in [84, Sec. IV.A]. By denoting with $e(t) = \sqrt{(x_r(t) - x(t))^2 + (y_r(t) - y(t))^2}$ the tracking error, the tracking performance have been evaluated using four different indices: (a) integral absolute error (IAE) ($\int_0^{k_f} |e(t)|dt$), (b) integral square error (ISE) ($\int_0^{k_f} e(t)^2 dt$), (c) integral time-weighted absolute error (ITAE) ($\int_0^{k_f} t|e(t)|dt$), (d) integral time squared error (ITSE) ($\int_0^{k_f} te(t)^2 dt$).

In the the performed experiments the following lemniscate trajectory (see Fig. 4.2) has been considered:

$$\begin{bmatrix} x_r(t) \\ y_r(t) \end{bmatrix} = \begin{bmatrix} 0.6\sin(\frac{t}{3.5}) \\ 0.6\sin(\frac{t}{7}) \end{bmatrix},\ t \in [0, k_f]\,,\ k_f = 44$$

with a robot's initial pose $q(0) = [x(0),\, y(0),\, \theta(0)]^T = [0.6,\, 0,\, \pi]^T$. It is straightforward to verify that for the given trajectory and robot constraints, the associated disturbance set (4.2.7) is a ball of radius $r_d = 0.0287$ and that $\hat{\mathcal{U}}$ is a ball of radius $r_u = 0.1982$ (see (4.2.9)). Consequently, since $B\hat{\mathcal{U}}$ is a ball of radius $T_s r_u = 0.0297$, the condition $\mathcal{D} \subset B\hat{\mathcal{U}}$ is satisfied, see Assumption 4.1. To cover the initial tracking error $\tilde{z}(0) = z_r(0) - z(0) = [0.4106,\, -0.0447]^T$, a family of $N = 396$ ROSC sets has been computed using (4.2.11). Moreover, the ST-RHC algorithm has been configured to use a multi-objective cost function $J(\tilde{z}, u) = \|A\tilde{z}(k) + Bu - Bu_r(k)\|_2^2 + 0.5\|u\|_2^2$, where the first term takes into account the tracking error and the second one the control effort.

The obtained experimental results are shown in Figs. 4.2-4.4 and Table 4.1. Fig. 4.2 shows the reference robot's trajectories where it can be noted that the proposed ST-MPC strategy, similarly to [25, 84], allows the robot's trajectory to quickly converges to the reference. Indeed, from Fig. 4.4 it is evident that the tracking error enters the RCI region $\mathcal{T}^0$ at $k = 1.8$[sec] where it remains confined thereafter. Fig. 4.3 shows that the computed left and right wheel angular velocities fulfill the prescribed constraints and that the robot's orientation error remains bounded. In the same figure, it is possible to appreciate how the compensated control action (4.2.15) allows $\tilde{z}(k)$ to remain bounded in a neighbourhood of the origin that is much smaller than the worst-case region $\mathcal{T}^0$ (obtained for $\hat{u}_r(k) = 0, \forall\, k$).

Finally, Table 4.1 and Fig. 4.5 summarize and contrast the tracking performance of the proposed T-ST-RCH strategy with the selected alternative schemes. The obtained numerical results confirm that the tracking performance obtained by T-ST-RHC is superior to the ones obtained by the competitors. The latter can be justified as follows. The controller in [84] only used instantaneous information about the (x_r, y_r) coordinates of the reference trajectory, while the proposed solution exploits their first and second derivatives. The solutions [19, 22, 25] are developed without explicitly taking into account the robot's maximum velocity constraints. Consequently, saturation phenomena arising from the large initial tracking error degrade their performance. Finally, for average CPU times required by the proposed solution to solve (4.2.16) and (4.2.14) are 1.58 $[ms]$ and 13.02 $[ms]$. On the other hand, the CPU times of the competitors are: 2.1 $[ms]$ for [84], 0.78 $[ms]$ for [25], 0.025 $[ms]$ for [19], and 0.032 $[ms]$ for [22].

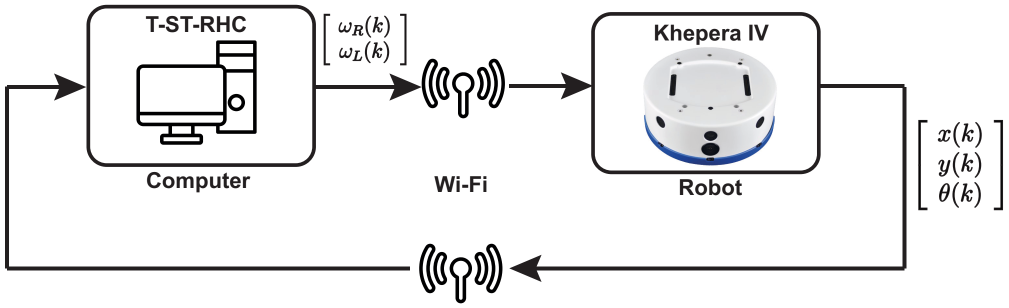

T-ST-RHC
$\begin{bmatrix} \omega_R(k) \\ \omega_L(k) \end{bmatrix}$
Khepera IV
Computer
Wi-Fi
Robot
$\begin{bmatrix} x(k) \\ y(k) \\ \theta(k) \end{bmatrix}$

Figure 4.1: Hardware-in-the-loop setup with Khepera IV.

Table 4.1: Average Tracking performance Indices.

	IAE	**ISE**	**ITAE**	**ITSE**
T-ST-RHC	0.690	0.225	1.942	0.148
[84]	1.433	0.258	19.320	0.560
[19]	1.421	0.325	6.503	0.554
[22]	1.767	0.400	10.653	0.735
[25]	2.853	0.378	53.130	3.341

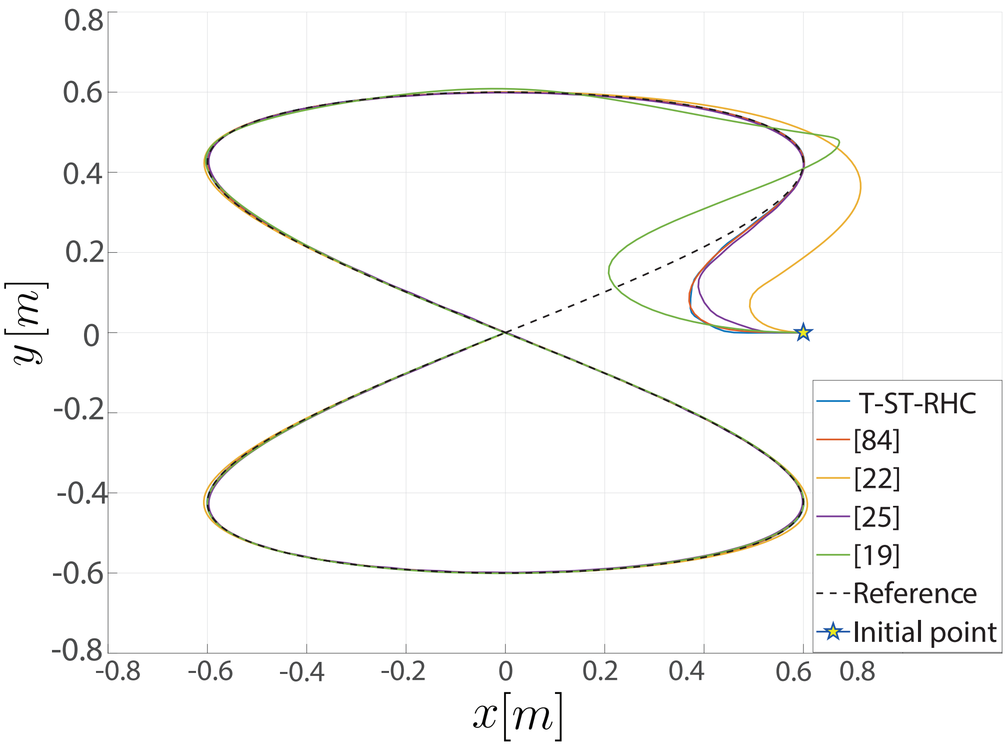


Figure 4.2: Trajectories performed by the robot.

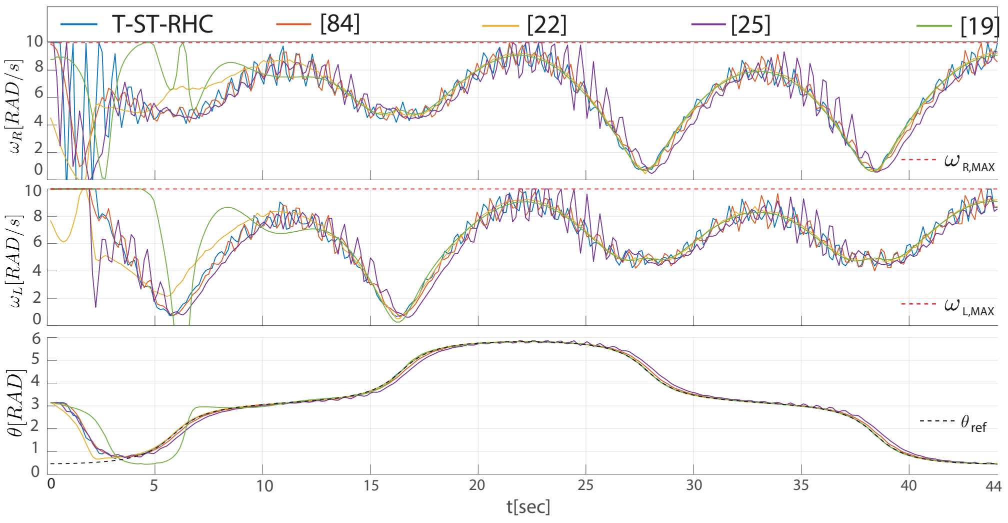


Figure 4.3: Wheels angular velocities and robot's orientation.

4.4 Conclusions

In this chapter, a novel set-theoretic receding horizon control strategy has been proposed to solve the trajectory tracking problem for input-constrained differential-drive robots. By considering an input-output linearized description of the vehicle kinematics, the strategy has been designed to take into account the associated time-varying and orientation-dependent

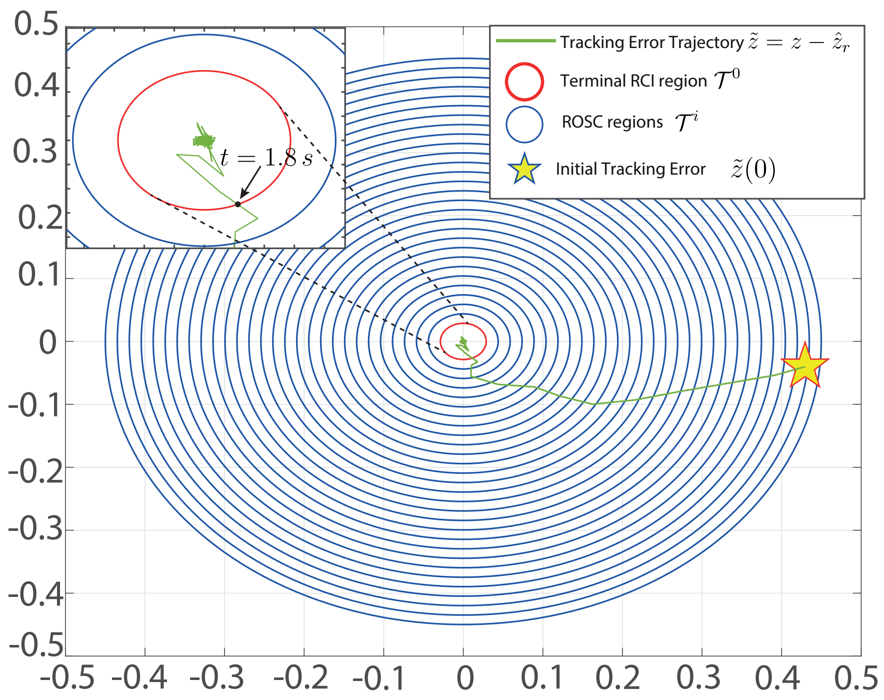


Figure 4.4: Computed $\{\mathcal{T}^i\}$ and trajectory tracking error for the feedback linearized robot. For the sake of clarity, only some of the computed ROSC sets are shown.

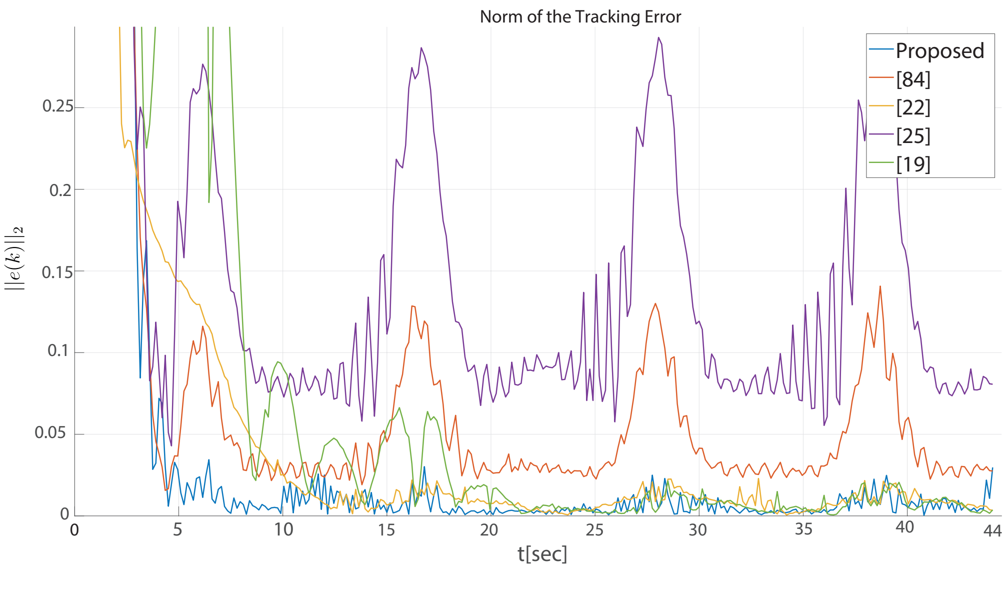


Figure 4.5: Comparison of trajectory tracking errors

constraints. To this end, a worst-case approximation of the constraint set has been exploited to offline design the smallest control invariant region for the tracking error and a family of robust one-step controllable sets whose union characterizes the worst-case domain of attraction of the proposed controller. Then, online, non-conservative and constraint-admissible control inputs have been computed resorting to a receding horizon strategy exploiting the knowledge of the robot's orientation and reference trajectory. Experimental results obtained using a Khepera IV differential-drive robot and comparison with four alternative schemes have shown the superior tracking performance of the proposed solution.

5 Collision-Free Platooning of Mobile Robots Through a Predictive Control Approach based on Feedback-Linearization

In this chapter, a solution to achieve collision-free platooning for input-constrained differential-drive robots is presented. The platooning policy is based on a leader-follower approach where the leader tracks a reference trajectory while followers track the leader's pose with an inter-agent delay. First, the leader and the follower kinematic models are feedback linearized and the platoon's error dynamics and input constraints are formally characterized. Then, a set-theoretic model predictive control strategy discussed in Chapter 4 is extended to address the platooning trajectory tracking control problem. An ad-hoc collision avoidance policy is also proposed to guarantee collision avoidance amongst the agents. The properties and the results of the proposed control architec- ture are validated through experiments performed on the formation of Khepera IV differential drive robots.

5.1 Formation setup and problem formulation

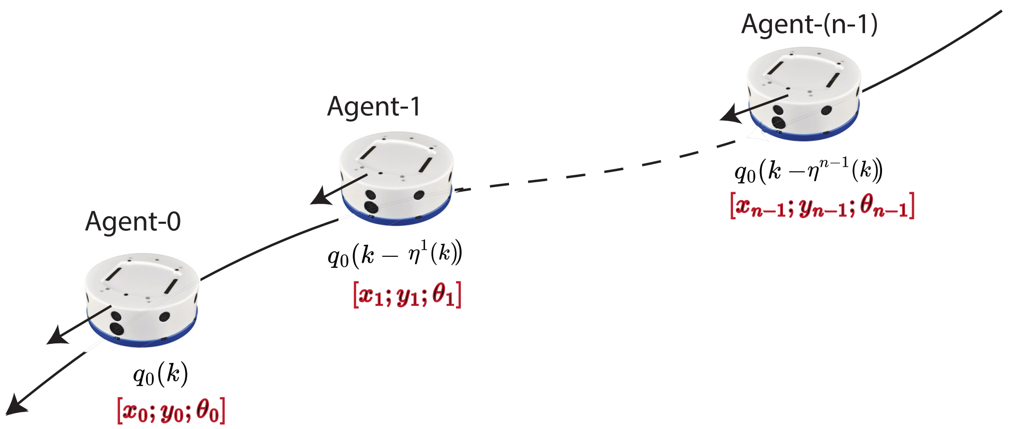


Figure 5.1: Platoon of N vehicles following the reference

Considered setup: Consider a formation of N mobile robots (i.e., the agents) described by the following constrained kinematic model (see Chapter 2)

$$\begin{array}{rcl} x^i(k+1) & = & x^i(k) + T_s \frac{R}{2}\left(\omega_R^i(k) + \omega_L^i(k)\right)\cos\theta^i(k) \\ y^i(k+1) & = & y^i(k) + T_s \frac{R}{2}\left(\omega_R^i(k) + \omega_L^i(k)\right)\sin\theta^i(k) \\ \theta^i(k+1) & = & \theta^i(k) + T_s \frac{R}{D}(\omega_R^i(k) - \omega_L^i(k)) \end{array} \qquad (5.1.1)$$

where $T_s > 0$ is the sampling time, $q^i = [x^i, y^i, \theta^i]^T$ is the pose of the center of mass of the robot. The left and the right wheel angular speeds $\omega_R^i, \omega_L^i \in \mathbb{R}$ are the control inputs of the system.

Furthermore, the control inputs are subject to box-like constraints, i.e., the set of admissible wheels' angular speed for the differential drive:

$$\begin{aligned} \mathcal{U}_d &= \{[\omega_R^i, \omega_L^i]^T \in \mathbb{R}^2 : \; H_d\,[\omega_R^i, \omega_L^i]^T \leq 1\}, \\ H_d &= \begin{bmatrix} \frac{-1}{\overline{\Omega}} & 0 & \frac{1}{\overline{\Omega}} & 0 \\ 0 & \frac{-1}{\overline{\Omega}} & 0 & \frac{1}{\overline{\Omega}} \end{bmatrix}^T \end{aligned} \tag{5.1.2}$$

where $\overline{\Omega}$ is the maximum angular speed the wheels' motors can perform.

The differential-drive kinematics (5.1.1) can be transformed into equivalent unicycle kinematics via the following change of input variables:

$$\begin{bmatrix} v^i(k) \\ \omega^i(k) \end{bmatrix} = T \begin{bmatrix} \omega_R^i(k) \\ \omega_L^i(k) \end{bmatrix}, \quad T = \begin{bmatrix} \frac{r}{2} & \frac{r}{2} \\ \frac{r}{D} & -\frac{r}{D} \end{bmatrix} \tag{5.1.3}$$

obtaining:

$$\begin{aligned} x^i(k+1) &= x^i(k) + T_s v^i(k) \cos\theta^i(k) \\ y^i(k+1) &= y^i(k) + T_s v^i(k) \sin\theta^i(k) \\ \theta^i(k+1) &= \theta^i(k) + T_s \omega^i(k) \end{aligned} \tag{5.1.4}$$

where $v^i, \omega^i \in \mathbb{R}$ are the linear and angular speeds of the robot respectively. Consequently $q_i = [x^i, y^i, \theta^i]^T$ denotes the pose of the vehicle, and $[\omega_R^i, \omega_L^i]^T$ the control input vector.

The input constraint set (5.1.2), mapped into the unicycle input space, transforms into a rhombus-like set, $\mathcal{U}_u \subset \mathbb{R}^2$, $0_2 = [0, 0]^T \in \mathcal{U}_u$, which defines the admissible linear and angular velocities for the unicycle, i.e.,

$$\mathcal{U}_u = \{[v, \omega]^T \in \mathbb{R}^2 : H_u \left[v^i, \omega^i\right]^T \leq 1\}, \; H_u = H_d T^{-1} \tag{5.1.5}$$

The agents are organized in a leader-followers configuration, where $i = 0$ denotes the index of the leader robot and $i = 1 \ldots N-1$ the indexes of the followers.

We assume that the leader agent is equipped with an online path planner providing a bounded and smooth 2D-trajectory in terms of reference position $(x_r(t), y_r(t))$, velocities $(\dot{x}_r(t), \dot{y}_r(t)$, and accelerations $(\ddot{x}_r(t), \ddot{y}_r(t))$ for the leader robot's center of mass, where $t \in \mathbb{R}^+$. Then the leader's pose and control inputs are broadcasted to all the follower's agents. To this end, different communication channels are established, i.e., between the leader agent and the followers, and between every two consecutive agents i and $i+1$, $\forall i = 0, 1, \dots N-2$ (see the network topology in Fig.5.2). The latter requirement is essential to guarantee collision avoidance capabilities between subsequent agents. We also assume that the leader's path planner module is capable of generating a safe trajectory that does not intersect the followers' positions, with a certain safe distance $\overline{d}$, at any given time, i.e.,

$$dist(z_r(k), \mathcal{B}(\overline{d}, z^i(k)) > 0, \forall k \geq 0, \forall i = 1, \dots, N-1$$

where

$$\mathcal{B}(\overline{d}, z^i(k) = \{z^i \in \mathbb{R}^2 \mid (z^i - z^i(k))^T Q_{\overline{d}}^{-1}(z^i - z^i(k)) \leq 1\}$$

with $Q_{\overline{d}} = \overline{d}^2 I$. Moreover, the reference longitudinal velocity is assumed to be lower bounded by $\underline{v_r}$, i.e., $v_r(k) > \underline{v_r}$, $\forall k \geq 0$ All the vehicles are required to follow the same reference trajectory with a desired inter-vehicles delay $\overline{\eta}^i > 0$ where $\overline{\eta}^i > \overline{\eta}^j$ if $\forall i > j$

Remark 5.1. In order to guarantee collision-avoidance requirements, the inter-agent delay is assumed to be dynamically adjustable at runtime. To this end, in the following, the inter-agent delay is treated as a function of time k, namely $\eta^i(k) > 0$.

Problem 3. *Given the reference pose $q_r(k) = [x_r(k), y_r(k), \theta_r(k)]^T$ and the setup described above, design a platooning control strategy such that all the agents can track a delayed reference trajectory while ensuring absence of collisions. Consequently, the leader and follower subproblems of interest are:*

[P1-1]: Design a trajectory tracking control law $[\omega_R^0(k), \omega_L^0(k)]^T = \phi^0(k, q^0(k), q_r(k)) \in \mathcal{U}_d$ such that the tracking error of the leader with respect to the reference trajectory, namely $\tilde{q}^0(k) = q^0(k) - q_r(k)$ remains bounded $\forall k \geq 0$.

[P1-2]: Design a trajectory tracking control law $[\omega_R^i(k), \omega_L^i(k)]^T = \phi^i(k, q^i(k), q^0(k - \eta^i(k))) \in \mathcal{U}_d$ such that $\tilde{q}^i(k) = q^i(k) - q^0(k - \eta^i(k))$ remains bounded $\forall k \geq 0$, $\forall i = 1 \dots N-1$, i.e. the followers track the leader pose delayed of $\eta^i(k) \in \mathbb{N}^+$ time instants while avoiding collisions with the other robots.

Remark 5.2. $\eta^i(k)$ is the *inter-vehicle delay* for the agent i, i.e. a time delay between the leader's reference trajectory and the one tracked by the agent i. It is worth it underlying that such a delay is not a communication or processing latency, but rather a control parameter adjusted at runtime to avoid collisions between agents.

Remark 5.3. The robot's reference orientation $\theta_r(t)$, longitudinal velocity $v_r(t)$ and angular velocity $\omega_r(t)$, fulfilling the unicycle kinematics, can be computed as [19]:

$$\begin{aligned}
\begin{bmatrix} v_r(t) \\ \omega_r(t) \end{bmatrix} &= \begin{bmatrix} \sqrt{\dot{x}_r(t)^2 + \dot{y}_r(t)^2} \\ \frac{\ddot{y}_r(t)\dot{x}_r(t) - \ddot{x}_r(t)\dot{y}_r(t)}{\dot{x}_r(t)^2 + \dot{y}_r(t)^2} \end{bmatrix} \\
\theta_r(t) &= \text{ATAN}_2\left(\dot{y}_r(t), \dot{x}_r(t)\right)
\end{aligned} \tag{5.1.6}$$

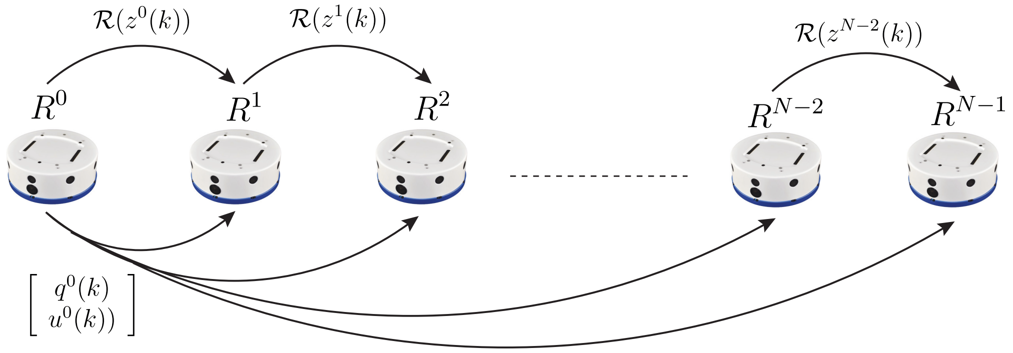


Figure 5.2: Vehicle to Vehicle (V2V) communication graph

5.2 Proposed Solution

In this chapter, the platooning formation control problem is solved by combining input-output feedback linearizations and set-theoretic MPC arguments. Specifically, first feedback linearization is used to derive an equivalent linear model describing the unicycle kinematics (5.1.4). Then, such a linearized model is exploited to derive a collision-free control strategy that drives the platoon along a desired reference trajectory.

5.3 Linearized vehicle kinematics via input-output linearization

By introducing the following change of output coordinates:

$$z^i(k) = \begin{bmatrix} x^i(k) + b\cos\theta^i(k), \; y^i(k) + b\sin\theta^i(k) \end{bmatrix}^T \tag{5.3.1}$$

with $b > 0$ i.e., representing the position of a point $\mathbf{B}^i$ displaced with respect to the center of mass of the robot, and by using the following input transformation

$$\begin{bmatrix} v^i(k) \\ \omega^i(k) \end{bmatrix} = T_{FL}(\theta^i) \begin{bmatrix} u_1^i(k) \\ u_2^i(k) \end{bmatrix}, \; T_{FL}(\theta^i) = \begin{bmatrix} \cos\theta^i & \sin\theta^i \\ \frac{-\sin\theta^i}{b} & \frac{\cos\theta^i}{b} \end{bmatrix}$$

the unicycle model (5.1.4) is recast into the following two-single integrator model,

$$z^i(k+1) = Az^i(k) + Bu^i(k), \; A = I_{2\times2}, \; B = T_s I_{2\times2} \tag{5.3.2a}$$

$$\theta^i(k+1) = \theta^i(k) + T_s \frac{-\sin\theta^i(k) u_1^i(k) + \cos\theta^i(k) u_2^i(k)}{b} \tag{5.3.2b}$$

where $u^i(k) = [u_1^i(k),\, u_2^i(k)]^T \in \mathbb{R}^2$ are the control inputs of linearized robot's model, while (5.3.2b) defines a decoupled nonlinear internal dynamics.

5.4 Agents' error dynamics

Here, the feedback-linearized tracking error dynamics for each agent are derived. First, let's define the reference pose for the i-th agent as

$$q_r^i(k) = \begin{cases} [x_r(k), y_r(k), \theta_r(k)]^T, & \text{If } i = 0 \\ [x^0(k-\eta^i(k)), y^0(k-\eta^i(k)), \theta^0(k-\eta^i(k))]^T \\ \qquad \text{If } i = 1 \ldots N-1 \end{cases}$$

where, $q_r^i(k) = [x_r^i(k), y_r^i(k), \theta_r^i(k)]^T$ and its reference control inputs as

$$\begin{bmatrix} v_r^i(k) \\ \omega_r^i(k) \end{bmatrix} = \begin{cases} [v_r(k), \omega_r(k)]^T, & \text{If } i = 0 \\ [v^0(k-\eta^i(k)), \omega^0(k-\eta^i(k))]^T, \\ \qquad \text{If } i = 1 \ldots N-1 \end{cases}$$

i.e, the reference is defined as the generated reference trajectory for the agent 0, and as the leader delayed reference for all the agents $i = 1 \ldots N-1$.

Remark 5.4. The reference pose and inputs are assumed to satisfy the unicycle kinematics

(5.1.4), $\forall i = 0, 1 \ldots N-1$

Then, the feedback linearized tracking error is defined as

$$\tilde{z}^i(k) = z^i(k) - z_r^i(k)$$

where

$$z_r^i = \left[\begin{array}{c} x_r^i(k) + b\cos\theta_r^i(k),\; y_r^i(k) + b\sin\theta_r^i(k) \end{array}\right]^T \tag{5.4.1}$$

As shown in [85], the linearized tracking error dynamics can be computed as follows:

$$\tilde{z}^i(k+1) = A\tilde{z}^i(k) + Bu^i(k) + d^i(k) \tag{5.4.2}$$

where

$$d^i(k) = -Bu_r^i(k), \quad u_r^i(k) = T_{FL}^{-1}(\theta_r^i(k)) \left[v_r^i(k), \omega_r^i(k)\right]^T \tag{5.4.3}$$

Remark 5.5. *Under the assumption of bounded reference, the disturbance $d^0(k)$ is also bounded. Furthermore, since $d^i(k)$, $\forall i = 0, 1 \ldots N-1$, depends on the leader's control inputs and orientation which are assumed bounded, $d^i(k) \in \mathcal{D}^i \subset \mathbb{R}^2$, $\forall i = 0, 1 \ldots N-1$. Moreover, knowing the bound of $d^i(k)$, $\mathcal{D}^i$ can be over-approximated with a ball of radius r_{d^i}, i.e,*

$$\mathcal{D}^i = \{d^i \in \mathbb{R}^2 : d^{iT} {Q_d^i}^{-1} d^i \leq 1\},\; Q_d^i = {r_d^i}^2 I_{2\times 2} \tag{5.4.4}$$

Lemma 5.1. *[93] If a control law $u(\cdot)$ is such that* (5.4.2) *is stable, the point* $\mathbf{B^i}$ *tracks any bounded reference trajectory with a bounded internal dynamic. Consequently, also the tracking error $\tilde{q}(k)$ is bounded.* □

5.4.1 Input constraints characterization

In [103] it has been proved that the set of admissible inputs for the model (5.3.2a), and consequently for the error dynamics (5.4.2), is the following orientation-dependent polyhedral set

$$\begin{gathered}\mathcal{U}(\theta^i) = \{[u_1^i, u_2^i]^T \in \mathbb{R}^2 : \, H(\theta^i)\, [u_1^i, u_2^i]^T \leq 1\}, \\ H(\theta^i) = H_d T^{-1} T_{FL}(\theta^i) = \\ = \begin{bmatrix} \frac{D\sin\theta^i - 2\cos\theta^i b}{2\bar{\Omega}Rb} & \frac{-D\cos\theta^i - 2\sin\theta^i b}{2\bar{\Omega}Rb} \\ \frac{-D\sin\theta^i - 2\cos\theta^i b}{2\bar{\Omega}Rb} & \frac{D\cos\theta^i - 2\sin\theta^i b}{2\bar{\Omega}Rb} \\ \frac{-D\sin\theta^i + 2\cos\theta^i b}{2\bar{\Omega}Rb} & \frac{D\cos\theta^i + 2\sin\theta^i b}{2\bar{\Omega}Rb} \\ \frac{D\sin\theta^i + 2\cos\theta^i b}{2\bar{\Omega}Rb} & \frac{-D\cos\theta^i + 2\sin\theta^i b}{2\bar{\Omega}Rb} \end{bmatrix}\end{gathered} \tag{5.4.5}$$

It has also been proved that there exists a worst-case circular inner approximation $\mathcal{U}(\theta^i)$, $\forall\theta^i$, defined as follows:

$$\hat{\mathcal{U}}^i = \bigcap_{\forall\theta^i} \mathcal{U}^i(\theta^i) = \{u^i \in \mathbb{R}^2 \, | {u^i}^T {Q_u^i}^{-1} u^i \leq 1\}, \; Q_u^i = r_u^{i2} I_{2\times 2} \tag{5.4.6}$$

where $r_u^i = \frac{2\bar{\Omega}Rb}{\sqrt{4b^2+D^2}}$.

Lemma 5.2. *$\mathcal{U}(\theta^i)$ admits the following circular outer approximation, $\forall\theta^i$ (see Fig. 5.3):*

$$\tilde{\mathcal{U}} = \bigcup_{\forall\theta^i} \mathcal{U}(\theta^i) = \{u^i \in \mathbb{R}^2 \, | {u^i}^T \tilde{Q}_u^{-1} u^i \leq 1\}, \; \tilde{Q}_u = \tilde{r}_u^2 I_{2\times 2} \tag{5.4.7}$$

where $\tilde{r}_u = \max\{\bar{\Omega}R, \, 2\bar{\Omega}Rb \backslash D\}$

Proof. Following the same arguments presented in [103], one finds that the lengths of the semi-diagonals of the time-varying polyhedral set $\mathcal{U}(\theta^i)$, $\forall\theta^i$ are:

$$sd_1 = \bar{\Omega}R, \quad sd_2 = \tfrac{2\bar{\Omega}Rb}{D}$$

Therefore, the radius of the smallest circle inscribing the polyhedron is given by

$$\tilde{r}_u = \max\{sd_1, \, sd_2\} = \max\{\bar{\Omega}R, \, 2\bar{\Omega}Rb \backslash D\}$$

Given the radius of the circle, the outer approximation can be written as (5.4.7), concluding the proof. □

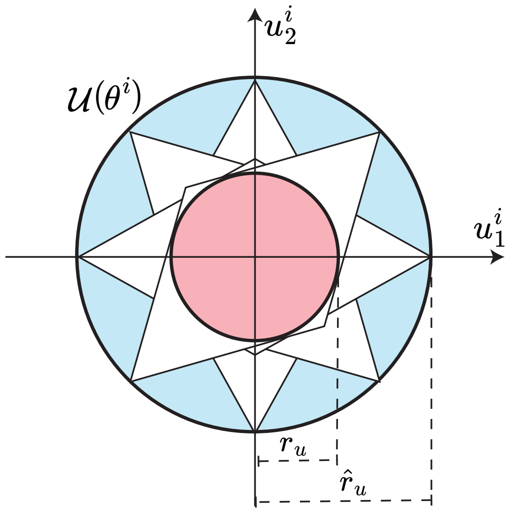


Figure 5.3: Outer approximation of the time-varying input constraint set $\mathcal{U}(\theta(k))$

5.4.2 Set-Theoretic receding horizon control for trajectory tracking

To address the trajectory tracking requirements imposed by the considered platooning tracking control problem 3, we extend the set-theoretic RHC proposed in Chapter 4 [85] to solve a trajectory tracking control problem for input-output linearized mobile robot described by (5.4.2). Such a strategy can be summarized as follows.

Notation: in the following, $\mathcal{T}_j^i$ denotes the j-th set for the i-th robot.

The algorithm consists of two distinct phases:

- *Offline:* For each agent $i = 0, 1, \dots N-1$, first, define the optimal state-feedback control law

$$u^i(k) = -B^{-1}\tilde{z}^i(k) \tag{5.4.8}$$

which ensures that the disturbance-free model is asymptotically stable. Then, under the assumption $\mathcal{D}^i \subset B\hat{\mathcal{U}}^i$, the smallest RCI set $\mathcal{T}_0^i$ (see definition 2.9) associated with the feedback control law and with the worst-case input constraint $\hat{\mathcal{U}}^i$ is given by $\mathcal{T}_0^i = \mathcal{D}^i$. Finally, starting from $\mathcal{T}_0^i$, build a family of ROSC sets $\{\mathcal{T}_j^i\}_{j=1}^{N_s^i}$, $N_s^i > 0$, until a desired region of the state-space is covered.

- *Online ($\forall\, k$):* First, compute the set-membership index

$$j^i(k) := \min_{j^i} s.t. \{j^i \geq 0 : \tilde{z}^i(k) \in \mathcal{T}_j^i\}$$

Then:

- if $j^i(k) > 0$ solve:
$$\begin{aligned} u^i(k) &= \arg\min_{u^i} J(\tilde{z}^i(k), u^i) \ s.t. \\ A\tilde{z}^i(k) + Bu^i - d^i(k) &\in (\mathcal{T}^i_{j(k)-1}) \\ u^i &\in \mathcal{U}(\theta^i(k)) \end{aligned} \tag{5.4.9}$$
where $J^i(\tilde{z}^i(k), u^i)$ is a convex cost function.

- else $u^i(k) = -B^{-1}\tilde{z}^i(k) + \hat{u}^i_r(k)$ where $\hat{u}^i_r(k)$ is computed such that $u^i(k)$ complies with the current input constraints, i.e.,

$$\hat{u}^i_r(k) = \arg\min_{\hat{u}_r} \|\hat{u}^i_r - u^i_r(k)\|_2^2 \quad s.t. \tag{5.4.10}$$
$$-B^{-1}\tilde{z}^i(k) + \hat{u}^i_r \in \mathcal{U}(\theta^i(k)) \tag{5.4.11}$$

Property 5.1. In [103], it has been proved that

- $\forall \tilde{z}^i(0) \in \bigcup_{j=0}^{N} \mathcal{T}^i_j \, \forall d^i(k) \in \mathcal{D}^i$, the tracking-error state trajectory is Uniformly Ultimately Bounded (UUB) in $\mathcal{T}^i_0$, i.e., there exists a sequence of at most N^i_s control inputs that brings $\tilde{z}^i(k)$ into the terminal set $\mathcal{T}^i_0$.

- Optimization (5.4.9) is recursively feasible by construction.

- The offline computed terminal feedback control law (5.4.8), is optimal with respect to a Linear-Quadratic (LQ) cost.

- Optimization (5.4.10)-(5.4.11) provides the best feed-forward term $u_r(k)$, compatible with the time-varying input constraints, that cancels out, completely or partially, the effect of the disturbance $d^i(k)$.

- Given the circular structure of the sets $\hat{\mathcal{U}}^i$ and $\mathcal{D}^i$, starting from $\mathcal{T}^i_0$ a family of circular ROSC sets can be built as follows:

$$\mathcal{T}^i_j = \{\tilde{z}^i \in \mathbb{R}^2 : \tilde{z}^{i^T} Q^{i^{-1}}_j \tilde{z}^i \le 1\}, \quad Q^i_j = r^{i^2}_j I_{2\times 2} \tag{5.4.12}$$
$$r^i_j = r^i_{j-1} - r^i_d + T_s r^i_u \tag{5.4.13}$$

and the set-membership signal $j(k)$ can be computed as:

$$j(k) := \min\{j : \tilde{z}^i(k)^T Q_j^{i^{-1}} \tilde{z}^i(k) \leq 1\} \tag{5.4.14}$$

Remark 5.6. Since the bound of the set $\mathcal{D}^i$ depends on the reference trajectory (see Eq. (5.4.3)), the containment condition $B\mathcal{U} \subset \mathcal{D}^i$, $\forall$ imposes a constraint the linear and angular velocity of the reference trajectory. Moreover, to guarantee that the containment condition is satisfied for each follower robot, the input constraint set of the leader, namely $\hat{\mathcal{U}}^0$, must be such that $\hat{\mathcal{U}}^0 \subset \hat{\mathcal{U}}$

5.4.3 Proposed Predictive Platooning Tracking Control

In the following, the above-discussed predictive control strategy is extended to solve the platooning control problem 3 (see steps 5-7 of algorithm 3 and steps 9-11 of algorithm 4). To this end, in the following, the leader's and follower's control algorithms are addressed separately. In order to provide formal guarantees of the absence of collisions between the agents a further assumption on the initial formation's configuration is needed.

Assumption 5.1. Initial spatial configurations are sequentially assigned to agents depending on their indexes, i.e.,

$$\begin{gathered} \|z^0(0) - z^1(0)\|_2^2 < \|z^0(0) - z^2(0)\|_2^2 < \dots \\ \dots < \|z^0(0) - z^{N-1}(0)\|_2^2 \end{gathered} \tag{5.4.15}$$

1) Leader's control strategy:

Lemma 5.3. *Algorithm 3 solves Problem P1-1.*

Proof. See [85]. ☐

Algorithm 3 Leader's Tracking Algorithm

```
Offline:
1: Set $\mathcal{U} = \hat{\mathcal{U}}^0$ with $\hat{\mathcal{U}} \subset \hat{\mathcal{U}}^0 \subset B^{-1}\mathcal{D}^0$, $K = B^{-1}$, and $\mathcal{T}_0^0 = \mathcal{D}^0$; Build $\{\mathcal{T}_j^0\}_{j=1}^{N_s}$ using (5.4.12); Store $\{\mathcal{T}_j^0\}_{j=0}^{N_s}$.
Online:
1: Measure $x^0(k)$, $y^0(k)$, $\theta^0(k)\, v^0(k), \omega^0(k)$ and compute $\tilde{z}^0(k) = z^0(k) - z_r(k)$, with $z^0(k)$ as in (5.3.1), $z_r(k)$ as in (5.4.1);
2: Send $q^0(k), v^0(k), \omega^0(k)$ to each agent $i = 1, \ldots, N-1$
3: Compute $\mathcal{U}(\theta^0)$ as in (5.4.5) and $u_r(k)$ as in (5.4.3);
4: Find $j(k)$ as in (5.4.14);
5: if $j(k) > 0$, then
       $u^0(k) = \arg\min_u J(x,u)\ s.t.$   (5.4.16a)
       $A\tilde{z}^0(k) + Bu^0 - Bu_r(k) \in \mathcal{T}^0_{i(k)-1},\ u \in \hat{\mathcal{U}}^0$   (5.4.16b)
6: else
       $u^0(k) = -B^{-1}\tilde{z}^0(k) + \hat{u}_r(k),$ where   (5.4.17)
       $\hat{u}_r(k) = \arg\min_{\hat{u}_r} \|\hat{u}_r - u_r(k)\|_2^2 \quad s.t.$   (5.4.18a)
       $-B^{-1}\tilde{z}^0(k) + \hat{u}_r \in \hat{\mathcal{U}}^0$   (5.4.18b)
7: end if
8: Compute
       $\left[\omega_R^0(k), \omega_L^0(k)\right]^T = T^{-1}T_{FL}(\theta^0(k))u^0(k)$   (5.4.19)
   and apply it to the robot; $k \leftarrow k+1$, go to 1;
```

2) Follower's control strategy: First, it is worth noting that Problem P1-1 is equivalent to Problem P1-2 where $q_r = q^0(k - Ts\eta^i(k))$, i.e., the reference trajectory is replaced with the leader's pose delayed of $\eta^i(k)$ time instants. To this end, Algorithm 3, can be extended to solve Problem P1-2. However, although the online planner module (see Chapter 5.1 ensures the absence of collisions between the leader robot and each follower, the possibility of collision between followers may arise.

To provide collision-free guarantees, in the following, ROSC sets are exploited to ensure there are no intersections between the trajectories performed by the agents. In particular, by denoting with $z^i(k)$ the i-th follower robot's position, and with $\mathcal{R}^i(z^i(k))$ its one step-reachable set starting from the point $z^i(k)$ (see Definition 2.11), a collision-avoidance policy can be stated as follows:

$$\begin{aligned} &\text{if } \mathcal{R}^i(z^i(k)) \bigcap \mathcal{R}^{i-1}(z^{i-1}(k)) \neq \emptyset \\ \implies\ & u^i(k) = 0,\ \eta^i(k) \leftarrow \eta^i(k) + 1,\ \forall i = 1, \dots N-1 \end{aligned} \tag{5.4.20}$$

i.e., if at any time $k \geq 0$, the ROSR of agent i intersects the one of its immediate predecessor then the agent i is stopped and its inter-agent delay $\eta^i(k)$ incremented (see Fig.5.4). Such a policy ensures that the robots' trajectories never overlap.

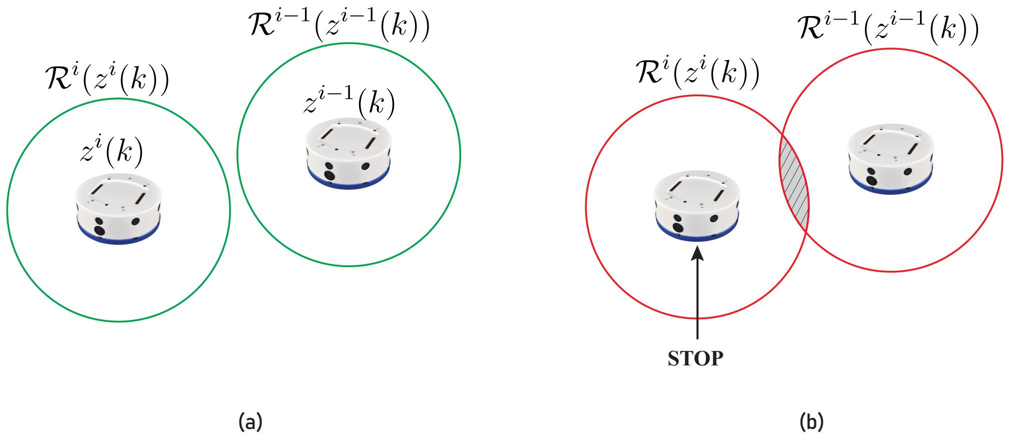


Figure 5.4: Collision-free (a) and potential collision (b) scenarios

Remark 5.7. By applying Definition 2.11, the ROSR set from the point $z^i(k)$ is given by:

$$\mathcal{R}(z^i(k)) = \{z \in \mathbb{R}^n : z \in Az^i(k) \oplus (B \cdot \mathcal{U}(\theta^i(k))) \oplus \mathcal{D}^i\}$$

where $\mathcal{U}(\theta^i(k))$ is a time-varying polyhedral set. Therefore, the computation of $\mathcal{R}(z^i(k))$ requires a numerical procedure. In order to provide an analytical way to compute such ROSR sets, we over-approximate $\mathcal{U}(\theta^i(k))$ with a circular set $\tilde{\mathcal{U}}$ (see Lemma 5.2. Then, the ROSR set $\mathcal{R}^i(z^i(k))$ is the following circular set:

$$\begin{gathered}\mathcal{R}^i(z^i(k))=\{z^i\in\mathbb{R}^2:(z^i-z^i(k))^T {Q_R^i}^{-1}(z^i-z^i(k))\leq 1\}\\ Q_R^i={r_R^i}^2 I_{2\times 2},\quad r_R^i=r_d^i+T_s\tilde{r}_u^i\end{gathered}\tag{5.4.21}$$

Proposition 5.4. *Algorithms 3-4 solve Problem 3*

Proof. First, as proved in [85] under the assumption $\hat{\mathcal{BU}}^i \supset B\mathcal{U}^0$, algorithm 4 guarantees bounded tracking error with respect to the delayed leader's trajectory, i.e. $\tilde{q}^i(k) = q^i(k) - q^0(k - \eta^i(k))$ is bounded under the effect of the control law (5.4.22)-(5.4.24), which is admissible with respect to $\mathcal{U}(\theta^i(k)), \forall\theta^i(k)$. Furthermore, the absence of collision holds. The following arguments prove such a point. Under the collision-avoidance policy (5.4.20), whenever the current DoA of agents i and $i-1$ are overlapped, the agent i is stopped. Therefore, starting from a feasible initial configuration, under the assumption of the feasible trajectory (see Chapter 5.1, and given the result of Lemma 5.3, the trajectory performed by the leader never intersects the one performed by the followers. Furthermore, since the tracking error $\tilde{q}^i(k)$ with respect to the leader pose is bounded, each agent's trajectory does not intersect the one performed by its predecessors, which is sufficient to guarantee the absence of collisions. Finally, since $v_r(k)$ is assumed to be lower bounded by a value $\overline{v}_r$, then the added incremental delay $\eta^i(k)(k)$ is consequently upper-bounded by a certain constant value $\hat{\eta}^i$ which ensures the convergence of the platoon to a stable configuration, concluding the proof. $\square$

Remark 5.8. It is worth mentioning that the proposed collision avoidance policy adjusts the inter-agent delay $\eta^i(k)$ by increasing it every time the policy is activated. However, the adaptive mechanism, as illustrated in Algorithms 3-4, never decreases such a delay. Depending on the specific application and trajectory, the adaptive mechanism may be extended as follows:

$$\begin{gathered}\text{If } \mathcal{R}^i(z^i(k))\bigcap\mathcal{R}^{i-1}(z^{i-1}(k))=\emptyset \;\text{ and }\; \eta(k)^i>\overline{\eta}^i \implies\\ \implies \eta(k)^i=\eta(k)^i-1\end{gathered}$$

i.e., every time the policy is not activated the inter-agent delay is decreased until it reaches the desired value $\overline{\eta}^i$. However, such an adaptive mechanism increases the number of times a follower vehicle is stopped, and it has not been implemented for practical reasons.

Algorithm 4 $i-$th follower's Tracking Algorithm

```
Offline:
1: Set $\mathcal{U} = \hat{\mathcal{U}}$, $K = B^{-1}$, and $\mathcal{T}_0^i = \mathcal{D}^i$; Build $\{\mathcal{T}_j^i\}_{j=1}^{N_s}$ using (5.4.12); Store $\{\mathcal{T}_j^i\}_{j=0}^{N_s}$.
Online:
1: Measure $x^i(k)$, $y^i(k)$, $\theta^i(k)$ and compute $\tilde{z}^i(k) = z^i(k) - z_r(k)$, with $z^i(k)$ as in (5.3.1),
   $z_r(k)$ as in (8.2.9);
2: Receive $q^0(k)$, $v^0(k)$, $\omega^0(k)$ from the leader and store them into a sequence;
3: Compute $\mathcal{R}^i(z^i(k)$ as in (5.4.21) and send it to agent $i+1$
4: Receive $\mathcal{R}^{i-1}(z^{i-1}(k))$ from agent $i-1$;
5: if $\mathcal{R}^i(z^i(k)) \bigcap \mathcal{R}^{i-1}(z^{i-1}(k)) \neq \emptyset$ then $u^i(k) = 0, \quad \eta^i(k) \leftarrow \eta^i(k) + 1$
6: else
7:     Compute $\mathcal{U}(\theta^i)$ as in (5.4.5) and $u_0(k)$ as in (5.4.3);
8:     Find $j(k)$ as in (5.4.14);
9:     if $j(k) > 0$, then   $u^i(k) = \arg\min_u J(\tilde{z}^i(k), u^i)$ s.t.   (5.4.22a)
           $A\tilde{z}^i(k) + Bu^i - Bu^0(k - \eta^i(k)) \in \mathcal{T}^i_{i(k)-1},\ u \in \hat{\mathcal{U}}^i$   (5.4.22b)
10:    else
           $u^i(k) = -B^{-1}\tilde{z}^i(k) + \hat{u}_r(k), \quad \text{where}$   (5.4.23)
           $\hat{u}_r(k) = \arg\min_{\hat{u}_r} \|\hat{u}_r - u^0(k - \eta^i(k))\|_2^2 \quad s.t. \quad -B^{-1}\tilde{z}^i(k) + \hat{u}_r \in \hat{\mathcal{U}}^i$
11:    end if
12: end if
13: Compute
           $\left[\omega_R^i(k), \omega_L^i(k)\right]^T = T^{-1} T_{FL}(\theta^i(k)) u^i(k)$   (5.4.25)
    and apply it to the robot; $k \leftarrow k+1$, go to 1;
```

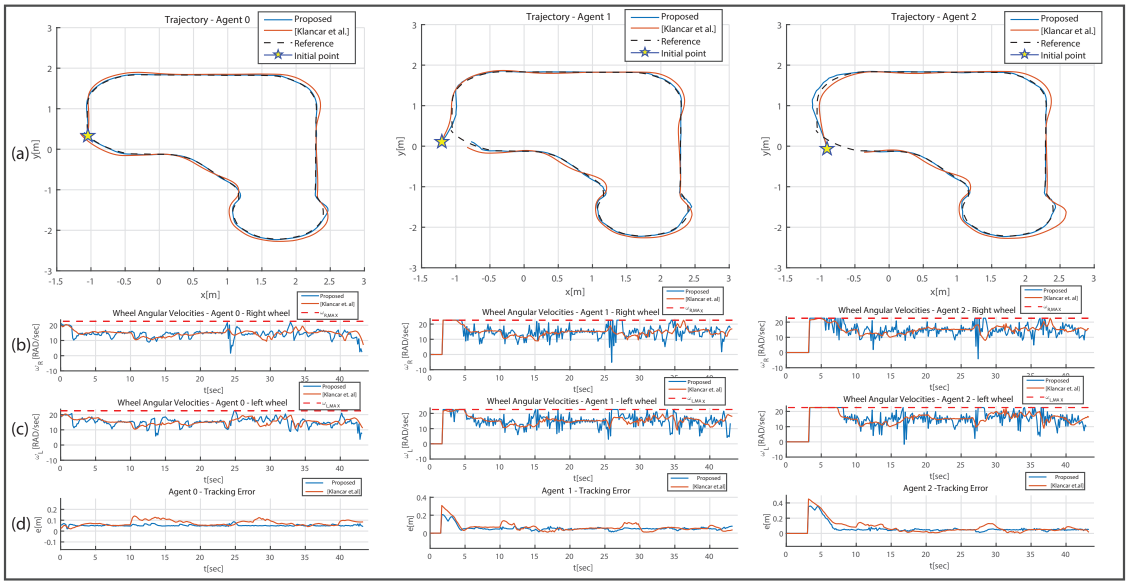


Figure 5.5: Experimental results: performed trajectory, wheel's angular velocity, and tracking error for agents $i = 0, 1, 2$.

5.5 Experimental Results and Conclusion

The proposed platooning control strategy has been validated through hardware-in-the-loop experiments, conducted with a platoon of $N = 3$ Khepera IV differential drive robots. A demo of the proposed experiment can be found at the following web link: `https://youtu.be/n5oOM-hr_rU?si=Wj_iZT6ymytNnb3O`.

Each robot consists of two independently-driven wheels of radius $R = 0.021[m]$, and axis length $D = 0.1047[m]$, capable of performing a maximum angular velocity $\overline{\Omega} = 1200[steps/sec] = 38.71[rad/sec]$. However, to avoid unmodeled dynamic effects the maximum allowed angular velocity has been reduced to $\overline{\Omega} = 700[steps/s] = 22.5833[rad/sec]$. Furthermore, the robots' kinematics have been feedback-linearized using $b = 0.1$ and discretized using a sampling time $T_s = 0.15$. An ad-hoc Indoor Positioning System (IPS) has been realized using a Vicon motion capture system and an unscented Kalman Filter algorithm [104], which is capable of providing accurate measurements of each agent's pose.

The performances of the proposed platooning tracking control algorithm have been contrasted with the solution presented in [50]. In both cases, the control strategy has been implemented on a workstation equipped with an *Intel I7-12700F* processor, running Matlab 2022b. Each robot communicates with its own controller through a TCP communication channel.

Control algorithm configuration: The proposed strategy has been configured with the following parameters: $J(\tilde{z}, u) = \|A\tilde{z}(k) + Bu\|_2^2$, $r_d^0 = 0.0507[m]$, $r_u^0 = 0.40[m]$, $r_d^i = 0.40[m]$,

$r_u^i = 0.4202[m]$, $\forall i = 1 \ldots N-1$, $\tilde{r}_u = 0.9059[m]$, $N_s^i = 1000$. On the other hand, the configuration parameters for the competitor control scheme can be found in [50].

An admissible reference trajectory, complying with the assumptions made in Sec.5.1 is generated by a planner module running on the same workstation. The generated reference trajectory spans across an area of $3.5[m] \times 4.25[m]$, and it is characterized by a constant velocity along the path (approximately $0.32[m/sec]$).

In order to compare the tracking performance, the following indexes are used, which are based on the tracking error $e(k) = \sqrt{(x_r^i(k) - x^i(k))^2 + (y_r^i(k) - y^i(k))^2}$: (a) integral absolute error (IAE) ($\int_0^{k_f} |e(k)|dk$), (b) integral square error (ISE) ($\int_0^{k_f} e(k)^2 dk$), (c) integral time-weighted absolute error (ITAE) ($\int_0^{k_f} k|e(k)|dk$), (d) integral time squared error (ITSE) ($\int_0^{k_f} ke(k)^2 dk$).

The results of the performed experiments are shown in Fig. 5.5 and Table 5.1. Specifically, Fig. 5.5 shows the comparison in terms of trajectory performed by the agents (a), angular velocities generated by the control algorithms (b)-(c), and considered tracking error $e(k)$ (d). Specifically, it can be appreciated how the proposed control algorithm is capable of ensuring a small tracking error for all the agents. On the contrary, the algorithm presented in [50] suffers from saturation phenomena that cause a loss of tracking performance for all the agents, as justified by Fig. 5.5(d). Furthermore, in Figs. 5.5(b)-(c), it can be appreciated how the control inputs fulfill the prescribed input constraint. On the other hand, the control signals generated using the algorithm in [50], have been manually saturated to the value $\overline{\Omega}$, in order to provide a fair comparison between the two algorithms. Finally, Tab. 5.1 shows the comparison in terms of error indexes. It can be observed, how the tracking performance is superior to the one obtained with a competitor control scheme for all the agents $i = 0, 1, 2$. The obtained results confirm how the formation converges to a stable platoon configuration, i.e., the inter-agents delay $\eta(k)^i \rightarrow \overline{\eta}^i$, as $k \rightarrow \infty, \forall i = 0, 1, 2$.

Table 5.1: Tracking Performance index of Platoon

Index	Agent-0		Agent-1		Agent-2	
	Proposed	[50]	Proposed	[50]	Proposed	[50]
IAE	2.2551	3.225	2.2785	2.7738	2.7161	3.3353
ISE	0.1206	0.2664	0.1634	0.2802	0.3708	0.5713
ITSE	2.6008	5.9389	2.5286	3.7351	3.5748	5.4430
ITAE	48.9147	72.0839	46.3983	52.8883	48.6321	55.7160

5.6 Conclusions

In this chapter, a novel control strategy has been proposed to address a platooning formation control problem for mobile robots. The proposed solution has been derived by combining feedback linearization and set-theoretic MPC arguments to achieve bounded trajectory tracking error for the considered platoon and deal with the input constraints of the considered mobile robots. Based on the concept of one-step forward reachable sets, a collision avoidance policy has been designed to guarantee the absence of collisions among agents formally. Finally, the proposed solution has been experimentally validated using a formation of Khepera IV robots, and its performance has been contrasted with a competitor scheme. The obtained results show that the proposed solution achieves better performance in terms of formation tracking error.

6 An Obstacle-Avoidance Receding Horizon Control Scheme for Constrained Differential-Drive Robot via Dynamic Feedback Linearization

In this chapter, a collision avoidance control strategy is proposed to control constrained differential-drive robots moving in static but unknown obstacle scenarios. The robot is assumed to be equipped with an on-board path planner providing a sequence of obstacle-free waypoints, and an *ad-hoc* constrained control strategy is designed to ensure absence of collisions and velocity constraints fulfillment. To this end, the nonlinear robot kinematics is redefined via a dynamic feedback linearization procedure, while a receding horizon control strategy is tailored to deal with time-varying state and input constraints. First, by considering the worst-case constraints realization, a conservative solution is offline determine to guarantee stability, recursive feasibility, and absence of collisions. Then, online, the tracking performance is significantly improved leveraging a non-conservative representation of the input constraints and set-theoretical containment conditions. Simulation results involving a differential-drive robot operating in a maze-like obstacle scenario are presented to show the effectiveness of the proposed solution.

6.1 Preliminaries and Problem Formulation

The following definitions will be used in the following of this chapter to support the proposed arguments.

Definition 6.1. A Linear Time-Invariant (LTI) system subject to structured norm-bounded perturbation (uncertainty) can be model as [99]:

$$\begin{aligned} z(k+1) &= Az(k) + Bu(k) + B_\rho \rho(k) \\ \rho(k) &= \Delta q(k) \\ q(k) &= C_q z(k) + D_{qu} u(k) \end{aligned} \tag{6.1.1}$$

where $k \in \{0, 1, \ldots\}$ are discrete-time instants, $z(k) \in \mathbb{R}^n$, $u(k) \in \mathbb{R}^m$ are the state and input vectors, and $\rho(k), q(k) \in \mathbb{R}^s$, $B_p \in \mathbb{R}^{n\times s}$, $Cq \in \mathbb{R}^{s\times n}$, $D_{qu} \in \mathbb{R}^{s\times m}$ and

$$\Delta = \begin{bmatrix} \Delta_1 & & \\ & \ddots & \\ & & \Delta_s \end{bmatrix}, \quad \Delta_i \leq 1 \tag{6.1.2}$$

describe the uncertainty, with Δ a convolution operator such that

$$\sum_{i=0}^{k} \rho(i)^T \rho(i) \leq \sum_{i=0}^{k} q(i)^T q(i), \quad \forall k \geq 0 \tag{6.1.3}$$

Definition 6.2. The distance between a point $p \in \mathbb{R}^n$ and a set $\mathcal{S} \subset \mathbb{R}^n$ is defined as $dist(p, \mathcal{S}) = \inf_{s\in\mathcal{S}} \|p - s\|_2$.

Problem Formulation:

Let's consider a differential-drive robot whose kinematics is described by the nonlinear model (2.1.3), subject to the wheels' angular velocity constraints (2.1.6). The problem addressed in this chapter is a navigation problem thereafter denoted as *Waypoint-Tracking with Obstacle AVoidance*, which can be formally stated as follows.

Problem 4. ***(Waypoints Tracking with Obstacle Avoidance (WPT-OA))****. Let $p(k) = [x(k), y(k)]^T \in \mathbb{R}^2$ be the planar position of the differential-drive robot* (2.1.3) *at time k, $p_f = [x_f, y_f]^T \in \mathbb{R}^2$ the desired target location, and $\mathcal{O}_f(k)$ the obstacle-free region detected by an on-board perception module at k. Under the assumption that an obstacle-free path exists from $p(0)$ to p_f, design a feedback control strategy*

$$[\omega_r(k), \omega_l(k)]^T = f(p(k), p_f, \mathcal{O}_f(k)), \quad \forall k$$

such that the robot is asymptotically driven to p_f, avoiding collisions and fulfilling the velocity constraints (2.1.6)*, i.e.,*

$$\lim_{k\to\infty} p(k) = p_f, \; p(k) \in \mathcal{O}_f(k), \; [\omega_r(k), \omega_l(k)]^T \in \mathcal{U}_d \tag{6.1.4}$$

□

The proposed solution is hereafter organized as follows. First, the nonlinear robot model is recast into a norm-bounded linear uncertainty model via dynamic feedback-linearization and the worst-case input constraint realization is formally characterized (Chapter 6.5). Then, a receding-horizon controller is developed and its convergence and stability proved (Chapter 6.6).

6.2 Linearized Vehicle Model

Following the procedure outlined in Chapter 2.2.5, the unicyle kinematics feedback linearized using the input transformation (2.2.27) and the dynamic compensator (2.2.26) obtaining the following linear system

$$\dot{z}(t) = A_c z(t) + B_c u(t), \quad A_c = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad B_c = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{6.2.1}$$

where $z = [x, y, \dot{x}, \dot{y}]^T$ is the vector state, and $u = [u_1, u_2]^T$ is the vector of linearizing inputs.

Remark 6.1. The input transformation (2.2.27) has a potential singularity in $v = 0$. As suggested in [19], such occurrence can be avoided by considering a saturation whenever the variable v falls below a sufficiently small threshold $\underline{v} > 0$.

By defining the sampling time $T_s > 0$, using the forward Euler discretization method, (6.2.1) can be discretized obtaining

$$z(k+1) = Az(k) + Bu(k) \tag{6.2.2}$$

with

$$A = I + T_s A_c = \begin{bmatrix} 1 & 0 & T_s & 0 \\ 0 & 1 & 0 & T_s \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad B = T_s B_c = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ T_s & 0 \\ 0 & T_s \end{bmatrix}$$

Remark 6.2. It can be proved that the above discretization and dynamic feedback linearization commute, i.e., the discrete-time linear model obtained via dynamic FL and consequent discretization is equal to the discrete-time linear model obtained by first discretizing the unicycle kinematics (2.1.8) and then applying the dynamic feedback linearization. The proof follows the same arguments outlined in Property 3.1 for the input-output linearization case and it has been omitted.

6.3 Input Constraint Mapping

Here, the set of admissible inputs for the unicycle is mapped into a set of admissible commands for the previously derived feedback linearized model.

First, it is possible to write the unicycle velocity $v(k)$ as

$$v(k) = T_s a(k) + v(k-1) \tag{6.3.1}$$

Therefore, the unicycle constraints $H_u \left[v(k), \omega(k)\right]^T \leq 1$ in (2.1.9) can be re-written as

$$\begin{aligned} H_u \begin{bmatrix} T_s & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a(k) + \frac{v(k-1)}{T_s} \\ \omega(k) \end{bmatrix} &\leq 1, \text{ or} \\ H_u H_{T_s} \begin{bmatrix} a(k) \\ \omega(k) \end{bmatrix} + H_u H_{T_s} \eta(k) &\leq 1 \end{aligned} \tag{6.3.2}$$

where

$$H_{T_s} = \begin{bmatrix} T_s & 0 \\ 0 & 1 \end{bmatrix}, \quad \eta(k) = \begin{bmatrix} \frac{v(k-1)}{T_s} \\ 0 \end{bmatrix} \tag{6.3.3}$$

Then, by applying the state-dependent input transformation (2.2.27) to (6.3.2) we obtain

$$H_u H_{T_s} T_{FL}^{-1}(\theta(k), v(k)) u(k) + H_u H_{T_s} \eta(k) \leq 1 \tag{6.3.4}$$

and by defining the shaping matrix

$$H(\theta, v) = H_u H_{T_s} T_{FL}^{-1}(\theta, v)$$

the input constraint set acting on (6.2.2) is

$$\mathcal{U}(\theta, v, \eta) = \{u \in \mathbb{R}^2 : H(\theta, v) u + H_u H_{T_s} \eta \leq 1\} \tag{6.3.5}$$

6.4 Model Representation via Norm-Bounded Uncertainty:

Since the input constraint set (6.2.2) is affine and it depends on three variables (θ, v, η), it is not straightforward to find its worst-case inner approximation. To address such an issue, we here propose an alternative constrained norm-bounded uncertainty representation [99] of (6.2.2),(6.3.5). By introducing the input translation

$$u(k) = \hat{u}(k) - T_{FL}(\theta, v)\eta(k) \tag{6.4.1}$$

the linearized model (6.2.2) and the constraint set (6.3.5) become

$$z(k+1) = Az(k) + B\hat{u}(k) + \zeta(k) \tag{6.4.2}$$

and

$$\hat{\mathcal{U}}(\theta, v) = \{\hat{u} \in \mathbb{R}^2 : H(\theta, v)\hat{u} \leq 1\} \tag{6.4.3}$$

where $\zeta(k)$ is the following affine term

$$\begin{aligned} \zeta(k) &= -BT_{FL}(\theta, v)\eta(k) \\ &= -\left[0, 0, v(k-1)\cos\theta(k), v(k-1)\sin\theta(k)\right]^T \end{aligned} \tag{6.4.4}$$

Remark 6.3. Note that the transformation (6.4.1) has the purpose to obtain an affine linearized model of the robot (6.4.2) subject to a zero-centered input constraint set (6.4.3). Such a modeling is instrumental to describe the dependency of the constrained feedback-linearized model from $\theta(k)$ as a structured norm-bounded uncertainty [99]. □

Proposition 6.1. *The differential-drive feedback linearized model* (6.4.2) *is equivalent to a norm-bounded uncertainty LTI model* (6.1.1), *where*

$$B_\rho = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ -1 & 0 \\ 0 & -1 \end{bmatrix}, C_q = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, D_{qu} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{6.4.5}$$

Proof. By recalling that $[z_3(k), z_4(k)]^T = [v(k)\cos\theta(k), v(k)\sin\theta(k)]^T$, if we define

$$q(k) = \begin{bmatrix} z_3(k) \\ z_4(k) \end{bmatrix} = C_q z(k) + D_{qu} u(k)$$

it is straightforward to show that C_q, D_{qu} are equal to the matrices in (6.4.5). Moreover, by defining

$$\rho(k) = \begin{bmatrix} v(k-1)\cos\theta(k) \\ v(k-1)\sin\theta(k) \end{bmatrix}$$

we have that $\zeta(k) = B_\rho \rho(k)$ if B_ρ is as shown in (6.4.5). Finally, the norm-bound condition (6.1.3) on the uncertainty Δ is also satisfied, i.e.,

$$\sum_{i=0}^{k} \rho(i)^T \rho(i) = \sum_{i=0}^{k} v^2(i-1) \leq \sum_{i=0}^{k} q(i)^T q(i) = \sum_{i=0}^{k} v^2(i)$$

concluding the proof. $\square$

6.5 Worst-case Input Constraint Set Characterization

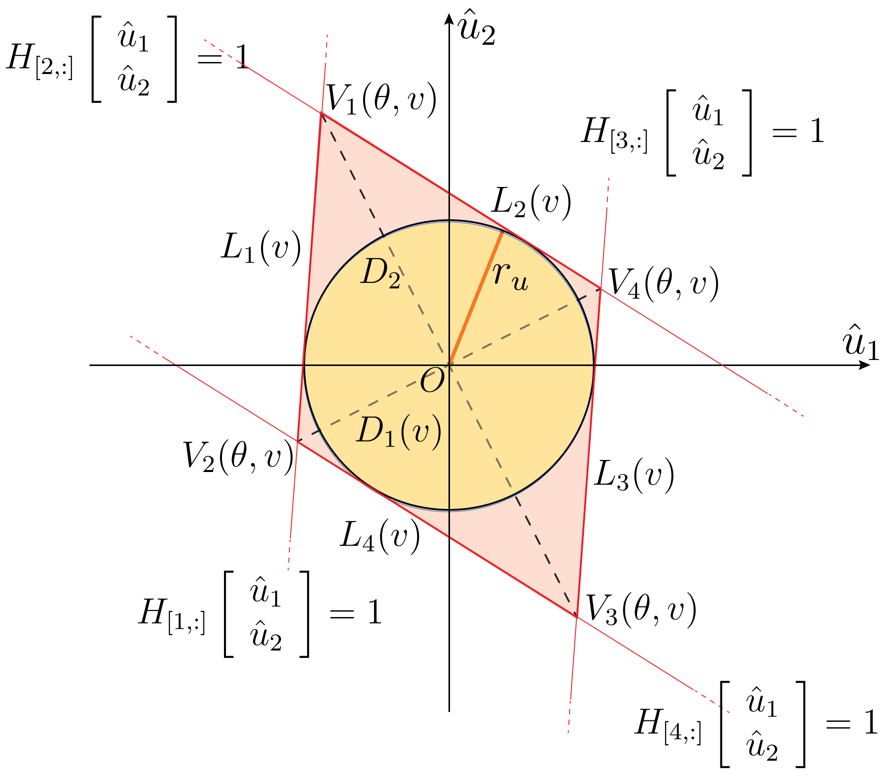
The input constraint set (6.4.3) is time-varying and its worst-case time-invariant inner approximation $\underline{\mathcal{U}} \subset \hat{\mathcal{U}}(\theta, v)$, $\forall(\theta, v)$ is, by definition,

$$\underline{\hat{\mathcal{U}}} := \bigcap_{\forall(\theta,v)} \hat{\mathcal{U}}(\theta, v) \qquad (6.5.1)$$

Lemma 6.2. *The worst-case input constraint set* (6.5.1) *is equal to the following ball set*

$$\underline{\hat{\mathcal{U}}}(0_2, r_u) = \{u \in \mathbb{R}^2 \,|\, u^T u \leq r_u^2\},\; r_u = \frac{2\bar{\Omega} r \underline{v}}{\sqrt{4\underline{v}^2 T_s^2 + d^2}} \qquad (6.5.2)$$

where $\underline{v}$ is the imposed minimum linear velocity (see Remark 6.1).



$H_{[2,:]}\begin{bmatrix}\hat{u}_1\\ \hat{u}_2\end{bmatrix}=1$
$\hat{u}_2$
$V_1(\theta,v)$
$H_{[3,:]}\begin{bmatrix}\hat{u}_1\\ \hat{u}_2\end{bmatrix}=1$
$L_2(v)$
$L_1(v)$
D_2
r_u
$V_4(\theta,v)$
$\hat{u}_1$
O
$D_1(v)$
$V_2(\theta,v)$
$L_3(v)$
$L_4(v)$
$H_{[1,:]}\begin{bmatrix}\hat{u}_1\\ \hat{u}_2\end{bmatrix}=1$
$V_3(\theta,v)$
$H_{[4,:]}\begin{bmatrix}\hat{u}_1\\ \hat{u}_2\end{bmatrix}=1$

Figure 6.1: Input constraint sets $\hat{\mathcal{U}}(\theta, v)$ and $\underline{\hat{\mathcal{U}}}(0, r_u)$.

Proof. First, the shaping matrix $H(\theta, v)$ in (6.4.3) is equals to

$$H(\theta,v)=\begin{bmatrix} \frac{d\sin\theta-2T_s\cos\theta v}{2\bar{\Omega}rv} & \frac{-d\cos\theta-2T_s\sin\theta v}{2\bar{\Omega}rv} \\ \frac{-d\sin\theta-2T_s\cos\theta v}{2\bar{\Omega}rv} & \frac{d\cos\theta-2T_s\sin\theta v}{2\bar{\Omega}rv} \\ \frac{-d\sin\theta+2T_s\cos\theta v}{2\bar{\Omega}rv} & \frac{d\cos\theta+2T_s\sin\theta v}{2\bar{\Omega}rv} \\ \frac{d\sin\theta+2T_s\cos\theta v}{2\bar{\Omega}rv} & \frac{-d\cos\theta+2T_s\sin\theta v}{2\bar{\Omega}rv} \end{bmatrix} \tag{6.5.3}$$

Therefore, the vertices $V_i(\theta)$, $i = 1, \ldots, 4$ of the polyhedron (6.4.3) can be analytically computed intersecting the four hyperplanes shaped by $H_{[i,:]}(\theta, v)[\hat{u}_1,\ \hat{u}_2]^T = 1$, $i = 1, \ldots 4$, (with $H_{[i,:]}$ denoting the $i - th$ row of H), obtaining

$$\begin{aligned} V_1(\theta,v) &= \left[-\frac{\bar{\Omega}r\cos\theta}{T_s}, -\frac{\bar{\Omega}r\sin\theta}{T_s}\right]^T,\ V_3(\theta,v) = -V_1(\theta,v) \\ V_2(\theta,v) &= \left[\frac{2\bar{\Omega}rv\sin\theta}{d}, -\frac{2\bar{\Omega}rv\cos\theta}{d}\right]^T,\ V_4(\theta,v) = -V_2(\theta) \end{aligned} \tag{6.5.4}$$

Then, by computing the lengths L_i, $i = 1, \ldots, 4$ of each side of $\mathcal{U}(\theta)$, it is straightforward to verify that they are all equal and independent of θ, i.e.,

$$L_i(v) = \frac{\bar{\Omega} r}{dT_s}\sqrt{4T_s^2 v^2 + d^2}, \quad i = 1, \ldots, 4$$

As a consequence, (6.4.3) is a rhombus-like set centered in the origin, rotating into the plane, according to the orientation θ (see Fig. 6.1). Furthermore, its two diagonals $D_1(v), D_2$ are

$$D_1(v) = \tfrac{4v\bar{\Omega}r}{d}, \quad D_2 = \tfrac{2\bar{\Omega}r}{T_s}$$

showing that only the first diagonal is directly proportional to the linear velocity v. As a consequence, the set $\hat{\underline{\mathcal{U}}}$ is equivalent to the biggest circle inscribed into $\hat{\mathcal{U}}(\theta, v)\ \forall\, \theta$, and for the worst-case minimum linear velocity $\underline{v}$ (see Remark 6.1). The radius of such a circle can be analytically determined resorting to simple geometric formulas as

$$r_u = \frac{D_1 D_2}{2\sqrt{D_1^2 + D_2^2}} = \frac{2\bar{\Omega}r\underline{v}}{\sqrt{4T_s^2\underline{v}^2 + d^2}} \tag{6.5.5}$$

□

6.6 Proposed RHC Tracking Controller

Here, first, an admissible, although conservative, solution to WPT-OA is found starting from the norm-bounded uncertain model (6.4.2), worst-case input constraint set (6.5.2) and a sequence of viable waypoints. Then, by leveraging the available online state measurements and proper set-membership containment conditions, a non-conservative control strategy, with guaranteed recursive feasibility, is derived.

By denoting with $z_f = [x_f, y_f, 0, 0]^T$ the equilibrium state of the affine model (6.4.2) associated to target position p_f, the objectives (6.1.4) can be re-formulated as

$$\lim_{k\to\infty} z(k) = z_f,\ \ z(k) \in \mathcal{O}_f^z(k),\ \ \hat{u}(k) \in \hat{\mathcal{U}}(\theta, v) \tag{6.6.1}$$

where $\mathcal{O}_f^z(k) = \mathcal{O}_f \times \mathbb{R}^2 \subseteq \mathbb{R}^4$, is the obstacle-free region mapped in the augmented space of the linearized model.

In principle, by denoting with $w(j|k)$ the $j-$step ahead prediction of a generic signal $w(k)$, the above problem can be, in principle, solved by means of following infinite-horizon optimization problem

$$\min_{\{\hat{u}(j|k),\, j\geq k\}} J_\infty(k) \ s.t. \tag{6.6.2a}$$

$$z(j+1|k) = Az(j|k) + B\hat{u}(j|k) + \zeta(j|k),\ j \geq k \tag{6.6.2b}$$

$$z(j|k) \in \mathcal{O}_f^z(j|k),\ j \geq k \tag{6.6.2c}$$

$$\hat{u}(j|k) \in \hat{\mathcal{U}}(\theta(j|k), v(j|k)),\ j \geq k \tag{6.6.2d}$$

where

$$J_\infty(k) = \sum_{j=k}^{\infty} (z(j|k) - z_f)^T Q_\xi (z(j|k) - z_f) + \hat{u}(j|k)^T Q_u \hat{u}(j|k)$$

is the Linear-Quadratic (LQ) cost, and $Q_\xi = Q_\xi^T > 0$, $Q_u = Q_u^T > 0$ matrices of proper dimensions.

Remark 6.4. Although appealing, the optimization (6.6.2) is not of practical interest due to two main issues:

- The vision radius $R < \infty$ is limited, an $\mathcal{O}_f^z(j|k)$ is typically non-convex and it cannot be predicted.
- Given $z(k)$, $\zeta(k|k)$,$\hat{\mathcal{U}}(\theta(k|k), v(k|k))$ are known. However, their prediction for $j > k$ is nonlinear and the associated constraints (6.6.2b),(6.6.2d) non-convex.

Hence, motivated by the above-mentioned difficulties, an ad-hoc customization of (6.6.2) is instead developed.

Assumption 6.1. The robot is equipped with a vision module capable of detecting the obstacle scenario, namely $\mathcal{O}(k)$, within a ball of radius R. Moreover, a path planning algorithm is used to find a sequence of viable 2D waypoints $p_i \in \mathbb{R}^2$ from $p(0)$ to p_f such that

$$\|p_{i+1} - p_i\|_2 \leq \bar{d} - \varepsilon,\ \forall i \tag{6.6.3}$$

$$dist(p_i, \mathcal{O}_j) \geq \bar{d},\ \forall i \text{ and } \forall \mathcal{O}_j \in \mathcal{O}(k) \tag{6.6.4}$$

where $\bar{d} > 0$ and $\varepsilon > 0$, with $\bar{d} - \varepsilon > 0$, are the desired safe distance from the obstacles and a tolerance, respectively. □

Remark 6.5. Note that Assumption 6.1 is not restrictive since it is trivially satisfied by different existing planners, such as Probabilistic Roadmap (PRM) [105], A^* [106], and Rapidly-exploring Random Tree (RRT) [107]. Moreover, Assumption 6.1 ensures the existence of an obstacle-free ball $\mathcal{B}(p_i, \bar{d}) := \{p \in \mathbb{R}^2 \,|(p-p_i)^T Q_{\mathcal{B}_{\bar{d}}}^{-1}(p-p_i) \leq 1\}$, $Q_{\mathcal{B}_{\bar{d}}} = \bar{d}^2 I_2$ that includes p_{i+1} and p_i. $\square$

6.7 Guaranteed Offline Feedback Controller

Let $\bar{z}_i = \left[p_i^T, 0, 0\right]^T$ be the equilibrium state of (6.2.2) associated to p_i (for a zero input), $\xi_i(k) = \bar{z}_i - z(k)$. Moreover, we denote with $\mathcal{B}^{\xi}(0_4, \bar{d}) = \mathcal{B}(0_2, \bar{d}) \times \mathbb{R}^2 \subset \mathbb{R}^4$ the augmented state constraint mapping $\mathcal{B}(0_2, \bar{d})$ into the augmented linerized model. For a waypoint p_i, a stabilizing sequence of control actions can be found using the norm-bounded uncertainty model derived in Proposition 6.1, and the worst-case realization of the input constraint set (6.5.2):

$$\min_{\{\hat{u}(k+j|k),\, j\geq k\}} \max_{\Delta} J_\infty(k), s.t. \tag{6.7.1a}$$

$$\xi_i(j+1|k) = A\xi_i(j|k) + B\hat{u}(j|k) + B_\rho \rho(j|k),\ j \geq k \tag{6.7.1b}$$

$$\rho(j|k) = \Delta q(j|k),\ j \geq k \tag{6.7.1c}$$

$$q(j|k) = C_q \xi_i(j|k) + D_{qu}\hat{u}(j|k),\ j \geq k \tag{6.7.1d}$$

$$\hat{u}(j|k)^T \hat{u}(j|k) \leq r_u^2,\ \ \xi_i(j+1|k) \in \mathcal{B}^{\xi}(0_4, \bar{d})\ j \geq k \tag{6.7.1e}$$

As shown in [99], if an upper bound of the LQ cost in (6.7.1) is considered, a solution can be obtained in a receding horizon fashion by means of a (LMI) optimization problem, see [99, Sec. 3].

Proposition 6.3. *Consider the norm-bounded uncertainty description* (6.1.1) *of the differential-drive robot (see Proposition6.1), the worst-case input constraint set* (6.5.2)*, the Assump-*

tion 6.1, and the LMI optimization problem:[4]

$$[Q_0^*,\, Y_0^*] = \arg \min_{\gamma, Q, Y, \Lambda} \quad \gamma \quad s.t. \tag{6.7.2a}$$

$$\begin{bmatrix} 1 & \xi_i(0)^T \\ \xi_i(0) & Q \end{bmatrix} \geq 0, \quad Q = Q^T > 0 \tag{6.7.2b}$$

$$\begin{bmatrix} Q & * & * & * & * \\ Q_u^{\frac{1}{2}} Y & \gamma I & * & * & * \\ Q_\xi^{\frac{1}{2}} Q & 0 & \gamma I & * & * \\ C_q Q + D_{qu} Y & 0 & 0 & \Lambda & * \\ AQ + BY & 0 & 0 & 0 & Q - B_\rho \Lambda B_\rho^T \end{bmatrix} \geq 0 \tag{6.7.2c}$$

$$\begin{bmatrix} r_u^2 I & Y \\ Y^T & Q \end{bmatrix} \geq 0 \tag{6.7.2d}$$

$$\Lambda = \begin{bmatrix} \lambda_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \lambda_s \end{bmatrix} > 0 \tag{6.7.2e}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} Q \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}^T = Q_{\mathcal{B}_{\bar{d}}} \tag{6.7.2f}$$

If (6.7.2) *admits a solution and the waypoints are switched according to the condition*

$$||p(k) - p_i||_2 \leq \varepsilon \tag{6.7.3}$$

then the control law

$$\begin{bmatrix} \omega_R(k) \\ \omega_L(k) \end{bmatrix} = T^{-1} H_{T_s} T_{FL}^{-1}(\theta(k), v(k)) \hat{u}_0(k) + T^{-1} H_{T_s} \eta(k) \tag{6.7.4}$$

$$\hat{u}_0(k) = Y_0^* Q_0^{*^{-1}} \xi_i(k) \tag{6.7.5}$$

solves the WPT-OA problem.

[4]The value of "$*$" is inferred by symmetry.

Proof. If (6.7.2a)-(6.7.2e) admits a solution, then the ellipsoid region $\mathcal{E}_0 = \{\xi^T \in \mathbb{R}^4 \,|\xi {Q_0^*}^{-1} \xi\}$ is robust positively invariant under the admissible and stabilizing control law (6.7.5) [99, Sec. 3]. Moreover, the LMI (6.7.2f) imposes that the projection of the invariant region along the (x, y) axis is equal to the safe ball $\mathcal{B}(0, \bar{d})$. This is instrumental to ensure that control law is admissible under the waypoint switching rule (6.7.3). Indeed, if at $\bar{k} > k$, $||p(\bar{k}) - p_i||_2 \leq \varepsilon$, then the distance $||p(\bar{k}) - p_{i+1}||_2 \leq \bar{d}$ (see Assumption 6.1) and $\xi_{i+1}(\bar{k}) \in \mathcal{E}_0$. Moreover, for viability arguments (see e.g., [108, Proposition 1]) the switching condition (6.7.3) is activated, in a finite time interval, i.e. $\bar{k} - k \leq \infty$, $\forall\, p_i$. As a consequence, for a given finite sequence of viable waypoints from $p(0)$ to p_f, the control law (6.7.4), obtained applying to (6.7.5) the inverse transformation of (2.2.27) and (6.3.1), solves the WPT-OA problem, concluding the proof. $\square$

6.8 Online receding-horizon controller

The admissible control law (6.7.5) has been offline determined considering the worst-case realization of the affine linearized model (6.4.2) (i.e., the worst-case realization of $\zeta(k)$) under the worst-case input-constraint set (6.5.2). However, such a source of conservativeness can be online eliminated using a receding-horizon approach. In particular, given the measurements $v(k-1)$ and $\theta(k)$, the polyhedral constraint set (6.3.5) is exactly known and the linearized nominal constrained model (6.2.2)-(6.3.5) can be directly used.

In what follows, by exploiting the above arguments and properly combining the controllers proposed in [99, 100, 109], a non-conservative receding-horizon controller is designed.

Proposition 6.4. *If* (6.7.2) *admits a solution at* $k = 0$*, and* p_i *complies with Assumption 6.1*

$\forall i$, then the following LMI optimization problem

$$[Q_k^*,\, Y_k^*] = \arg \min_{\bar{Q}_k \geq 0, \gamma_k, Q_k, Y_k} \quad \gamma \quad s.t. \tag{6.8.1a}$$

$$\begin{bmatrix} 1 & \xi_i(k)^T \\ \xi_i(k) & Q_k \end{bmatrix} \geq 0, \quad Q = Q^T > 0 \tag{6.8.1b}$$

$$\begin{bmatrix} Q_k + Q_k^T - \bar{Q}_k & * & * & * \\ AQ_k + BY_k & Q_k & * & * \\ Q_\xi^{\frac{1}{2}} Q_k & 0 & \gamma_k I & * \\ Q_u^{\frac{1}{2}} Y_k & 0 & 0 & \gamma_k I \end{bmatrix} \geq 0 \tag{6.8.1c}$$

$$\begin{bmatrix} Q_k & Y_k^T H^T(\theta(k))_{[j,:]} \\ H(\theta(k))_{[j,:]} Y_k & 1 \end{bmatrix} \geq 0 \tag{6.8.1d}$$

$$j = 1, \ldots, 4$$

$$Q_k \leq \mathcal{Q}_0^* \tag{6.8.1e}$$

has a solution $\forall k \geq 0$, and the state-feedback control law

$$\begin{bmatrix} \omega_R(k) \\ \omega_L(k) \end{bmatrix} = T^{-1} H_{T_s} T_{FL}^{-1}(\theta(k), v(k)) u(k) + T^{-1} H_{T_s} \eta(k) \tag{6.8.2}$$

$$u(k) = Y_k^* Q_k^{*^{-1}} \xi_i(k) - T_{FL}(\theta, v)\eta(k) \tag{6.8.3}$$

is non-conservative with respect to the robot's input constraints. Moreover, the control law (6.8.2)*, together with the switching rule* (6.7.3)*, provides a solution to the WPT-OA problem.*

Proof. First, if $\zeta(k) \equiv 0$, then $\hat{u}(k) = Y^* Q^{*^{-1}} \xi_i(k)$, obtained solving the LMI (6.8.1a)-(6.8.1d), defines a control law that fulfills the polyhedral input constraint (6.3.5) acting on (6.2.2), see [100, 109]. Therefore, the control law (6.8.3) complies with (6.3.5) $\forall\, \eta(k)$ by introducing a compensation for $H_u H_{T_s} \eta(k)$. Note that the norm-bounded uncertainty model used to solve (6.7.2) includes, among all its admissible realizations, the nominal model (6.2.2)-(6.3.5) used in (6.8.1). Moreover, since $\hat{\mathcal{U}}(\theta(k), v) \supset \underline{\hat{\mathcal{U}}}, \forall\, k$, and $\mathcal{E}_k = \{\xi^T \in \mathbb{R}^4 \,|\xi Q_k^{*^{-1}} \xi\} \subseteq \mathcal{E}_0$ (see LMI (6.8.1d)), the offline solution of (6.7.2) is always admissible for (6.8.1). The latter is sufficient to ensure recursive feasibility of (6.8.1) and that under (6.7.3), the control law (6.8.2) solves the WPT-OA problem. Finally, since at each time instant k, the non-conservative constraint set $\hat{\mathcal{U}}(\theta(k), v) \supset \hat{\underline{\mathcal{U}}}$ is directly exploited, then the proposed solution does not present any conservativeness with respect to the robot input constraint set. $\square$

Finally, Algorithm 5 summarizes all the above developments.

Algorithm 5 WPT-OA Control Strategy

```
Configuration:

  • Differential-drive robot parameters: $\bar{\Omega}, r, d$; Vision radius: $R$; LQ cost matrices:
    $Q_\xi, Q_u$; Obstacle-free ball shaping matrix $Q_{\mathcal{B}_{\bar{d}}} = \bar{d}^2 I_2$; Tolerance $\varepsilon > 0$;

  • Set $i = 0,\ k = 0$;

Offline ($k = 0$):

  1. Compute $\underline{\mathcal{U}}(r_u)$ as in (6.5.2)

  2. Select $p_0 \in \mathcal{B}(p(0), \bar{d})$, define $\bar{z}_0 = \left[p_0^T, 0, 0\right]^T$, $\xi(0) = \bar{z}_0 - z(0)$, solve (6.7.2).
     Store $Q_0^*$

Online ($\forall\, k > 0$):

  1. If $||p(k) - p_i||_2 \leq \varepsilon$ then obtain the next waypoint $p_{i+1} \in \mathcal{B}(p(k), \bar{d})$ from the
     planner. Set $i \leftarrow i + 1$;
     EndIf

  2. Compute $\bar{z}_i = \left[p_i^T, 0, 0\right]^T$

  3. Compute $T_{FL}(\theta, v)$ using (2.2.25), the input-constraint shaping matrix $H(\theta)$
     using (6.5.3), and the vector $\eta(k)$ using (6.3.3)

  4. Determine $Q_k^*, Y_k^*$ by solving (6.8.1);

  5. Compute $[\omega_R(k), \omega_L(k)]^T$ as in (6.8.2) and apply it to the robot
```

6.9 Simulation Results

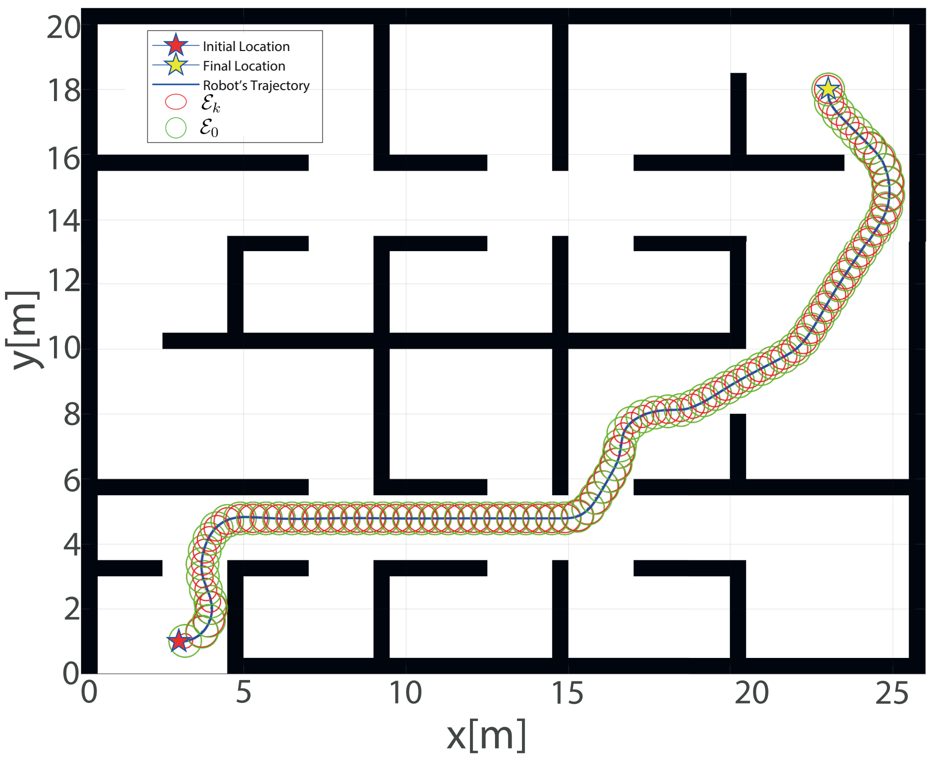


Figure 6.2: Maze-like environment and robot's trajectory.

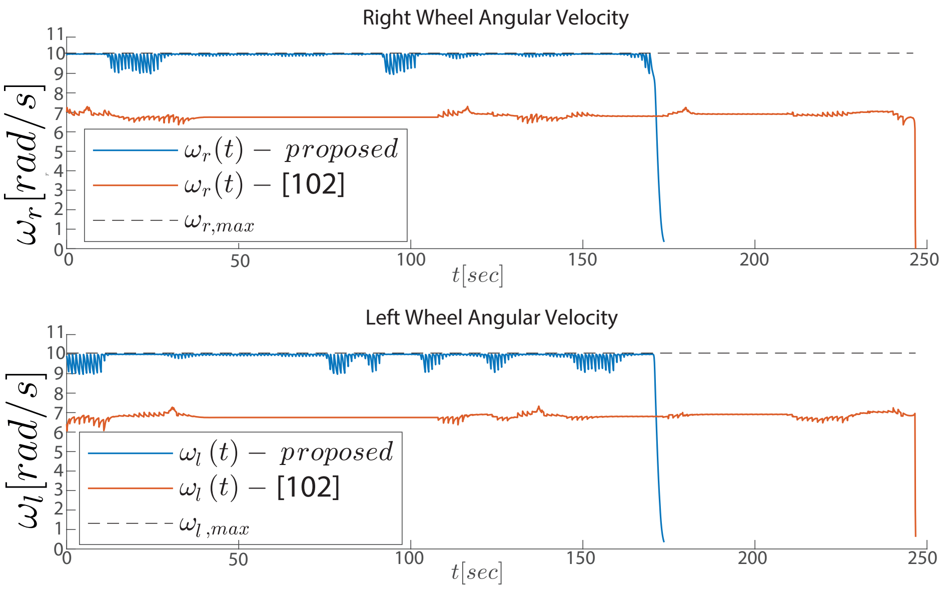


Figure 6.3: Wheels' angular velocities.

The effectiveness of proposed collision avoidance strategy is here verified by means of a simulation campaign where a differential-drive robot moves in a maze-like working environment (see the scenario in Fig. 6.2). The simulations have been carried in Matlab where we have used the built-in semi-definite programming solver "*mincx*" to solve the LMI optimizations (6.7.2) and (6.8.1). Moreover, the built-in function "*plannerHybridAStar*" has been used to implement the hybrid A^* path planner developed in [110] and obtain a sequence of viable waypoints $\{p_i\}$. The vehicle is a circular-shaped differential-drive robot having two independently driven wheels of radius $r = 2cm$, connected to each other by an axis of length $d = 5cm$. The two wheels are assumed to have a maximum angular velocity $\bar{\omega}_r = \bar{\omega}_l = \Omega = 10\, rad/sec$. The vision module describes the obstacle scenario by means of a binary occupancy map and its knobs are $\bar{d} = 50cm$ and $\varepsilon = 0.30cm$. Moreover, to ensure that the A^* planner output complies with (6.6.4), the detected obstacles are enlarged by $\bar{d} = 50cm$.

The control module has been implemented with $T_s = 0.1\,\text{sec}$, $Q_\xi = diag(100, 100, 0, 0)$, $Q_u = 0.01 I_{2\times 2}$ and $\underline{v} = 0.02m/s$, where "$diag(\cdot)$" is a function building a diagonal matrix containing its arguments on the main diagonal.

By considering a robot initial position $p_0 = [3, 1]^T$ and a target destination $p_f = [23, 18]^T$, the obtained simulation results are collected in Fig. 6.2-6.3. Fig. 6.2 shows that the robot's is able to reach p_f avoiding collisions with the obstacles. Moreover, it is also possible to appreciate how the the state trajectory is embedded in a worst-case tube of obstacle-free admissible trajectories shaped by the offline computed control invariant ellipsoidal region $\mathcal{E}_0$ (see the green circular sets in Fig. 6.2). Finally, Fig. 6.3 shows the angular velocities of the wheels and contrasts the proposed solution with the strategy in [102]. In particular, it is shown that the robot's velocity constraints are always fulfilled and that the proposed solution does not present any conservativeness w.r.t. the angular velocities, viz. the robot is able to move to its maximum velocity. On the other hand, the solution in [102] presents evident signs of conservativeness. This finds also confirmation in the fact that using the proposed controller the target is reached in 173.5 sec while resorting to [102] the same target is reached in 246.7 sec.

6.10 Conclusions

In this chapter, a solution to the obstacle avoidance problem for input-constrained differential-drive robots has been presented. The proposed controller has been developed by taking advantage of a novel norm-bounded uncertainty description of the robot dynamics. This was instrumental in offline characterizing the worst-case input realization of the time-varying input constraints acting on the feedback-linearized vehicle model. By resorting to a receding horizon approach, it has been shown that the proposed solution does not suffer from any conservativeness with respect to the input constraints. Moreover, under mild assumptions on the capabilities of the on-board vision module, the resulting receding horizon controller ensures the recursive feasibility property and the absence of collisions. Simulations, conducted on a differential-drive robot moving in a maze-like environment, have testified the main peculiarities of the proposed framework. Future research will be devoted executing the controller in real-time scenarios and providing comparisons with existing competitors.

7 On the Design of Control Invariant Regions for Feedback Linearized Car-Like Vehicles

In this chapter, a novel procedure is presented to design control invariant regions for feedback-linearized car-like vehicles subject to linear and steering velocity constraints. The input constrained kinematic model of a car-like vehicle and its input output linearized description are considered (see Sections 2.1.3 and 2.2.4). It is formally proved that the linear and steering velocity constraints recast into polyhedral time-varying constraints for the feedback lienearized system. Is is also shown that such time-varying input constraints admit a worst-case circular inner approximation. The proposed input constraint characterization is very important to formally guarantee stability and constraint fullfillment when a trajectory tracking control problem is considered. Specifically, the analytical design of a tracking controller with an associated invariant region is proposed, which is capable of ensuring constraints fulfillment. Finally, the proposed simulation results show the effectiveness of this solution and its potential to enable the design, via control invariance, of a new class of constrained and model predictive solutions for input-constrained feedback-linearized car-like vehicles.

7.1 Problem Formulation

Let's consider a car-like vehicle (see fig. 7.1), whose kinematics is described by the bicycle model (2.1.10), subject to linear and angular steering velocity constraints (2.1.12).

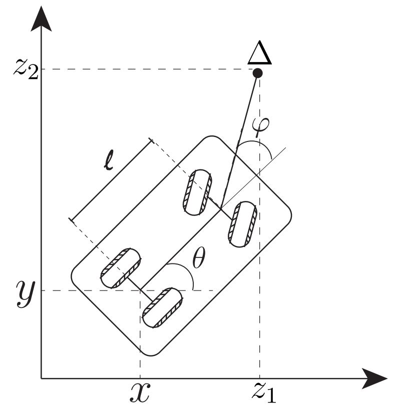


Figure 7.1: Car-like Vehicle Representation

Let's also consider an arbitrary reference trajectory described in terms of cartesian positions $x_r(t)$, $y_r(t)$, velocities $\dot{x}_r(t)$, $\dot{y}_r(t)$ and accelerations $\ddot{x}_r(t)$, $\ddot{y}_r(t)$ of the vehicle. Then, by considering the kinematic model (2.1.10), the reference values in terms of linear velocity $v_r(t)$, steering angular velocity $\omega_r(t)$, orientation $\theta_r(t)$, and steering angle $\varphi_r(t)$ can be computed as follows [91]:

$$\begin{array}{rc} v_r(t) = & \sqrt{\dot{x}_r(t)^2 + \dot{y}_r(t)^2} \\ \omega_r(t) = & l v_r(t) \frac{(\dddot{y}_r(t)\dot{x}_r(t) - \dddot{x}_r(t)\dot{y}_r(t))v_r^2(t) - 3(\ddot{y}_r(t)\dot{x}_r(t) - \ddot{x}_r(t)\dot{y}_r(t))(\dot{x}_r(t)\ddot{x}_r(t) + \dot{y}_r(t)\ddot{y}_r(t))}{v_r^6(t) + l^2(\ddot{y}_r(t)\dot{x}_r(t) - \ddot{x}_r(t)\dot{y}_r(t))^2} \\ \theta_r(t) = & \mathrm{ATAN}_2\left(\frac{\dot{y}_r(t)}{v_r(t)}, \frac{\dot{x}_r(t)}{v_r(t)}\right) \\ \varphi_r(t) = & \arctan\left(\frac{l(\ddot{y}_r(t)\dot{x}_r(t) - \ddot{x}_r(t)\dot{y}_r(t))}{v_r(t)^3}\right) \end{array} \tag{7.1.1}$$

Assumption 7.1. The reference trajectory $q_r(t) = [x_r(t), y_r(t), \theta_r(t), \varphi_r(t)]^T$ for (2.1.10) is uniformly bounded and smooth, i.e., $\exists \mathit{\Gamma} > 0 : \|q_r(t)\| < \mathit{\Gamma}, \forall t \geq 0, \ q_r(t) \in \mathcal{C}^2$. □

Definition 7.1 (Stable full-state tracking [93])**.** Consider the car model (2.1.10) and a reference trajectory $q_r(t)$ complying with Assumption 7.1. By considering the tracking error with respect to the reference trajectory, namely $\tilde{q}(t) = q_c(t) - q_r(t)$, stable full-state tracking is achieved if

$$\exists \delta > 0, \ t_0 \geq 0 \ \text{ s.t. } \|\tilde{q}(t_0)\| < \delta \implies \|\tilde{q}(t)\| < \varepsilon, \forall t \geq t_0$$

□

In order to achieve stable tracking of the reference trajectory, the input-output linearization procedure discussed in Chapter 2.2.4 can be deployed. Specifically, by using the state-feedback control law:

$$\begin{bmatrix} v \\ \omega_s \end{bmatrix} = T_{FL} w \tag{7.1.2}$$

where T_{FL} is computed as in (2.2.19), the kinematic of the car can be equivalently described via the following LTI system:

$$\dot{z} = \begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \tag{7.1.3a}$$

$$\begin{bmatrix} \dot{\theta} \\ \dot{\varphi} \end{bmatrix} = O(\theta, \varphi) \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \tag{7.1.3b}$$

where (7.1.3a) is the input-output linearized model and (7.1.3b) is a decoupled internal dynamics (see Chapter 2.2.4). Moreover, by substituting the input transformation (7.1.2), into the set of feasible control inputs (2.1.12), the following state-dependent and time-varying input constraint set is obtained for the feedback linearized system (7.1.3):

$$[w_1, w_2]^T \in \mathcal{U}(\theta, \varphi) = \{w \in \mathbb{R}^2 \colon L(\theta, \varphi)w \leq g\},\ L(\theta, \varphi) = HT_{FL}(\theta, \varphi) \tag{7.1.4}$$

Also the reference can be translated into the linearized system space by applying the output transformation (2.2.16), obtaining

$$z_r = \begin{bmatrix} x_r + lcos(\theta_r) + \Delta\cos(\theta_r + \varphi_r) \\ y_r + lsin(\theta_r) + \Delta\sin(\theta_r + \varphi_r) \end{bmatrix} \tag{7.1.5}$$

Consequently, by defining the tracking error with respect to the reference trajectory for (7.1.3a) as $\tilde{z}(t) = z(t) - z_r(t)$, the feedback-linearized error dynamics can be written as:

$$\dot{\tilde{z}}(t) = \tilde{w}(t), \tag{7.1.6a}$$

$$\tilde{w}(t) = w(t) - w_r(t),\ w_r(t) = M(\theta_r(t), \varphi_r(t))u_r(t) \tag{7.1.6b}$$

where (7.1.6b) represents the time-varying input constraint whose the linearized system is subject to.
The derived linearized error dynamics can be used to design a feedback control law and associated control invariant region that jointly solve the constrained tracking control problem for input-output linearized car-like vehicles.

Problem 5. *Design a tracking controller with associated control invariant region for the tracking error of the feedback linearized car-like model* (7.1.3) *guaranteeing stable full-state tracking (see Definition 7.1) while fulfilling the time-varying input constraints* (7.1.4).

7.2 Input constraints characterization

First step to design a control invariant region for the input-output linearized car-like vehicle is formally characterize the state-dependent and time-varying input constraint set $\mathcal{U}(\theta, \varphi)$. The following proposition analytically states the geometrical properties of $\mathcal{U}(\theta, \varphi)$.

Proposition 7.1. *The time-varying input constraint set* (7.1.4) *is a parallelogram-shaped set centered in* 0_2 *that rotates according to the state variables* $\theta(t)$ *and* $\varphi(t)$.

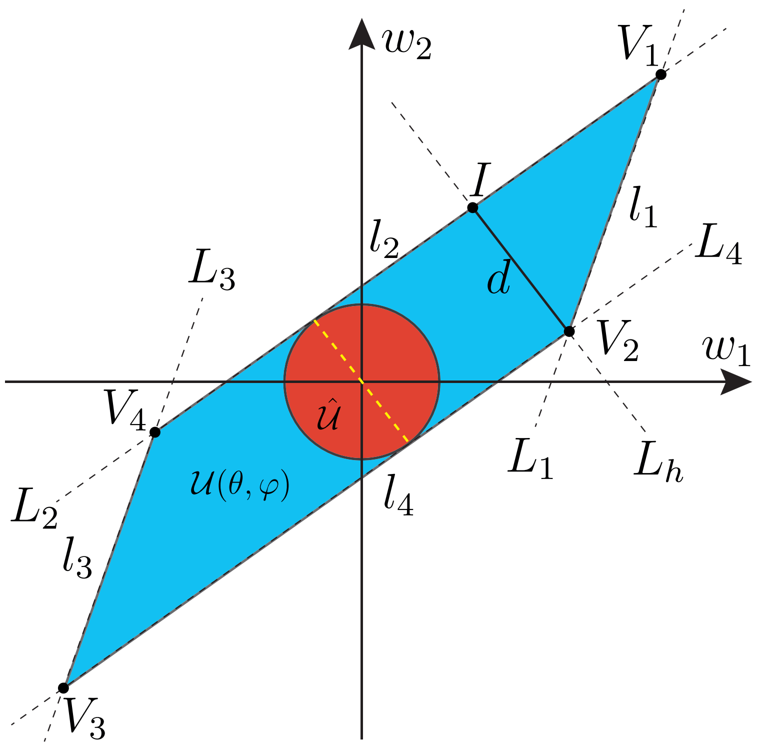


Figure 7.2: Time-varying input constraint

Proof. The shaping matrix $L(\theta,\varphi)$ of the polyhedral set (7.1.4) can be explicitly re-written as:

$$L(\theta,\varphi) = HT_{FL}(\theta,\varphi) =$$

$$\begin{bmatrix} -\frac{\cos(\theta)+\cos(\theta+2\varphi)}{2} & -\frac{\sin(\theta)+\sin(\theta+2\varphi)}{2} \\ \frac{\Delta\sin(\theta+2\varphi)-\Delta\sin(\theta)+2l\sin(\theta+\varphi)}{2\Delta l} & \frac{-\Delta\cos(\theta+2\varphi)+\Delta\cos(\theta)-2l\cos(\theta+\varphi)}{2\Delta l} \\ \frac{\cos(\theta)+\cos(\theta+2\varphi)}{2} & \frac{\sin(\theta)+\sin(\theta+2\varphi)}{2} \\ -\frac{\Delta\sin(\theta+2\varphi)-\Delta\sin(\theta)+2l\sin(\theta+\varphi)}{2\Delta l} & -\frac{-\Delta\cos(\theta+2\varphi)+\Delta\cos(\theta)-2l\cos(\theta+\varphi)}{2\Delta l} \end{bmatrix} \tag{7.2.1}$$

By intersecting the four hyperplanes characterizing the polyhedron, its vertices are (see Fig. 7.2):

$$V_1 = \begin{bmatrix} \frac{\Delta\sin(\theta+2\varphi)l\overline{\omega}+\Delta\sin(\theta)l\overline{\omega}+\Delta\overline{v}\cos(\theta)-\Delta\overline{v}\cos(\theta+2\varphi)-2\cos(\theta+\varphi)l\overline{v}}{(2l\cos(\varphi))} \\ \frac{-\overline{\omega}\Delta l\cos(\theta)-\overline{\omega}\Delta l\cos(\theta+2\varphi)-\Delta\sin(\theta+2\varphi)\overline{v}+\Delta\sin(\theta)\overline{v}-2\sin(\theta+\varphi)l\overline{v}}{2l\cos(\varphi)} \end{bmatrix} \tag{7.2.2}$$

$$V_2 = \begin{bmatrix} \frac{-\Delta\sin(\theta+2\varphi)l\overline{\omega}-\Delta\sin(\theta)l\overline{\omega}+\Delta\overline{v}\cos(\theta)-\Delta\overline{v}\cos(\theta+2\varphi)-2\cos(\theta+\varphi)l\overline{v}}{2l\cos(\varphi)} \\ \frac{\overline{\omega}\Delta l\cos(\theta)+\overline{\omega}\Delta l\cos(\theta+2\varphi)-\Delta\sin(\theta+2\varphi)\overline{v}+\Delta\sin(\theta)\overline{v}-2\sin(\theta+\varphi)l\overline{v}}{2l\cos(\varphi)} \end{bmatrix} \tag{7.2.3}$$

$$V_1 = -V_3, \quad V_2 = -V_4$$

Consequently, for symmetry, $\mathcal{U}(\theta,\varphi)$ is centred in 0, $\forall\theta\in\mathbb{R}$, $\forall\varphi\in(-\pi/2,\pi/2)$. By resorting to simple geometric arguments, the values of l_i, $i=1,2,3,4$ are

$$\begin{aligned} l_1 = l_3 &= 2\sqrt{\Delta^2\overline{\omega}^2} \\ l_2 = l_4 &= 2\sqrt{\tfrac{v_1^2(-\Delta^2\cos(2\varphi)+\Delta^2+2l^2)}{l^2(\cos(2\varphi)+1)}} \end{aligned} \tag{7.2.4}$$

and $m(L_1)=m(L_3)$, $m(L_2)=m(L_4)$, i.e., L_1 and L_3, and L_2 and L_4 are mutually parallel (see Fig. 7.2). Moreover, l_1 and l_3 are independent of $\theta(t)$ and $\varphi(t)$ and their values depend on the parameters Δ and $\overline{\omega}$. On the other hand, l_2 and l_4 are time-varying and their value is a function of $\varphi(t)$. Furthermore, l_2 and l_4 are minimal for $\varphi=0$ and monotonically increase with the module of φ until $\pi/2$ (from the left) or $-\pi/2$ (from the right) is approached. Consequently, by collecting all the above, it results that $\mathcal{U}(\theta,\varphi)$ is a time-varying parallelogram-shaped region centred in 0_2 that rotates around the origin according to θ and φ. □

7.3 Worst-Case Inner Approximation

The following proposition describes the inner and time-invariant worst-case approximation $\hat{\mathcal{U}}\subset\mathcal{U}(\theta,\varphi)$ complying with the cars' input constraint (2.1.12) $\forall\theta\in\mathbb{R},\varphi\in(-\pi/2,\pi/2)$.

Proposition 7.2. *If $l_1<l_2$, the worst-case constraint set inscribed in $\mathcal{U}(\theta(t),\varphi(t))$, $\forall\,\theta(t)\in\mathbb{R},\forall\,\varphi(t)\in(-\pi/2,\pi/2)$ is a circular region $\hat{\mathcal{U}}$ of radius $\hat{r}>0$ centered in 0_2, i.e.,*

$$\hat{\mathcal{U}}=\{w\in\mathbb{R}^2\,|w^Tw\leq\hat{r}^2\},\ \hat{r}=\Delta l\overline{\omega}\sqrt{\frac{1}{\Delta^2+l^2}} \tag{7.3.1}$$

Proof. First, for any fixed value of φ, the polyhedral set $\mathcal{U}(\theta(t),\varphi(t))$ rigidly rotates in the plane around 0_2 describing a circle centered at the origin (see Fig. 7.3). Moreover, since only l_2 and l_4 are a function of φ, the radius of the circle changes with $\varphi\in(-\pi/2,\pi/2)$. By observing the geometry of $\mathcal{U}(\theta,\varphi)$ in Fig. 7.2-(a), if $l_1<l_2$,, every segment orthogonal to the edges l_2 and l_4 corresponds to the height of $\mathcal{U}(\theta(t),\varphi(t))$, namely $d>0$, and to the diameter of the inner circular region. On the other hand, d is equal to the length of the segment $\overline{IV_2}$, where I is the intersection point between L_2 and the line, namely L_h, passing by V_2 and orthogonal to L_2. By noting that the equation describing L_2 can be obtained considering

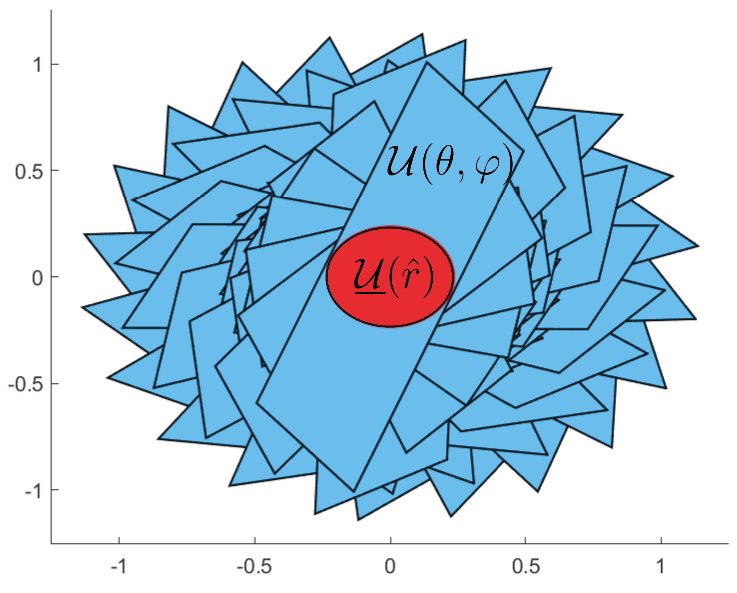


Figure 7.3: Time-varying input constraint set and its worst-case approximation

the second row of $L(\theta,\varphi)$ and g, we have

$$\begin{gathered} L_2: \quad \alpha(\theta,\varphi)w_1 + \beta(\theta,\varphi)w_2 = \gamma \\ \alpha(\theta,\varphi) = L[2,1]\,,\ \beta(\theta,\varphi) = L[2,2]\,,\ \gamma = g[2] \end{gathered} \tag{7.3.2}$$

which can be re-written as:

$$\begin{gathered} L_2: \quad w_2 = m(\theta,\varphi)w_1 + h(\theta,\varphi) \\ m(\theta,\varphi) = -\tfrac{\alpha(\theta,\varphi)}{\beta(\theta,\varphi)}, \quad h(\theta,\varphi) = \tfrac{\gamma}{\beta(\theta,\varphi)} \end{gathered} \tag{7.3.3}$$

Consequently, the equation describing L_h is

$$L_h: \quad w_2 - V_2[2] = -\frac{1}{m(\theta,\varphi)}(w_1 - V_2[1]) \tag{7.3.4}$$

Given L_2 and L_h, the intersection point I is given by the following linear system:

$$\begin{bmatrix} -m(\theta,\varphi) & 1 \\ \frac{1}{m(\theta,\varphi)} & 1 \end{bmatrix} I = \begin{bmatrix} h(\theta,\varphi) \\ \frac{V_2[1]}{m} + V_2[2] \end{bmatrix} \tag{7.3.5}$$

which admits a single solution

$$I = \begin{bmatrix} -m(\theta,\varphi) & 1 \\ \frac{1}{m(\theta,\varphi)} & 1 \end{bmatrix}^{-1} \begin{bmatrix} h(\theta,\varphi) \\ \frac{V_2[1]}{m(\theta,\varphi)} + V_2\,[2] \end{bmatrix} \tag{7.3.6}$$

Then, the radius $r(\varphi)$ of the circle inscribed in $\mathcal{U}(\theta,\varphi)$ can be computed as:

$$\begin{aligned} r(\varphi) &= \tfrac{1}{2}d = \tfrac{1}{2}\sqrt{(I-V_2)^T(I-V_2)} = \\ &= \Delta l\overline{\omega}\sqrt{\tfrac{1}{\Delta^2-\Delta^2\cos(\varphi)^2+l^2}} \end{aligned} \tag{7.3.7}$$

Hence, the value of $r(\varphi)$ decreases as φ approaches $\pi/2$ from the left or $-\pi/2$ form the right. In conclusion, the radius $\hat{r}$ of the worst case (smallest) circular region can be obtained by $r(\varphi)$ for φ approaching $\pm\pi/2$, i.e.,

$$\hat{r} = \lim_{|\varphi|\to\frac{\pi}{2}} r(\varphi) = \Delta l\overline{\omega}\sqrt{\frac{1}{\Delta^2+l^2}}$$

and $\hat{\mathcal{U}}$ can be written as in (7.3.1), concluding the proof. □

Remark 7.1. Although φ could in principle vary in the range $\left(-\frac{\pi}{2},\frac{\pi}{2}\right)$ in a practical application the steering angle is restricted in the range $\left[-\frac{\pi}{3},\frac{\pi}{3}\right]$. Therefore, to reduce the conservativeness of the inner approximation one may compute its radius by considering a restricted range of the steering angle φ.

As an example, in Fig. 7.3-(b) the time-varying input constraint set is depicted considering a feedback linearized robot having $l = 0.5$, $\Delta = 0.35$, $\overline{v} = 0.5$, $\overline{\omega}_s = \frac{\pi}{4}$. It can be appreciated how such a polyhedron rotates in the plane $\forall\theta \in [-\pi,\pi]$, $\forall\varphi \in \left[-\frac{\pi}{3},\frac{\pi}{3}\right]$. However regardless the specific realization of $\theta(k)$, $\varphi(k)$, it is always possible to identify a circular intersection set whose radius r can be computed through the formula (7.3.7), where $\varphi = \frac{\pi}{3}$, obtaining $r = 0.2351$

7.4 Tracking controller and associated control invariant set

In this subsection, the above characterization of the worst-case input constraint set $\hat{\mathcal{U}}$ is exploited to design a constrained controller with associated control invariant region capable of solving problem 5.

The following lemma, which has been proved in [93], will be used in the following to prove the overall stability of the proposed control scheme.

Lemma 7.3. *Consider a reference trajectory $q_r(t)$ complying with Assumption 7.1 with $v_r(t)$ and $\omega_r(t)$ satisfying* (7.1.1) *and such that*

$$0 < v_r(t) \leq V,\, V > 0 \quad \textit{and} \quad \forall |\varphi_r(t)| \leq \frac{\pi}{2}, \quad \forall t \tag{7.4.1}$$

Then, the tracking-error zero dynamics with respect to q_r, i.e.,

$$\begin{bmatrix} \dot{\tilde{\theta}} \\ \dot{\tilde{\varphi}} \end{bmatrix} = O_{FL}(\tilde{\theta}, \tilde{\varphi}) T_{FL}(\tilde{\theta}, \tilde{\varphi}) \begin{bmatrix} v_r \\ \omega_r \end{bmatrix} \tag{7.4.2}$$

are asymptotically stable [93, Theorems 1-3]. Consequently, if a generic control law $w(t)$ stabilizes (7.1.6a)*, then stable full-state tracking is achieved [93].* □

By considering a sufficiently small sampling time interval $T_s > 0$ and by resorting to a ZOH discretization method, the discretized model of (7.1.6) is

$$\tilde{z}(k+1) = A\tilde{z}(k) + Bw(k) - Bw_r(k), \tag{7.4.3a}$$

$$w(k) \in \mathcal{U}(\theta(k), \varphi(k)) \tag{7.4.3b}$$

where $A = I_{2\times 2}$, $B = T_s I_{2\times 2}$.

Remark 7.2. For the considered car-like model (2.1.10) and ZOH discretization method, it is straightforward to prove that discretization and feedback linearization commute. □

Since the reference signal is bounded (see Assumption 7.1), then $w_r(k)$ in (7.4.3a) is bounded and the term $-Bw_r(k)$ can be considered as a disturbance. As a consequence, a stabilizing controller can be designed starting from the disturbance-free model

$$\tilde{z}(k+1) = A\tilde{z}(k) + Bw(k) \tag{7.4.4}$$

By considering the constraint-free model (7.4.4), a (LQ) state-feedback control law

$$w(t) = -K\tilde{z}(k) \tag{7.4.5}$$

can be designed such that

$$\tilde{z}(k+1) = (A - BK)\tilde{z}(k) \tag{7.4.6}$$

is asymptotically stable. The following proposition shows that given the structure of (7.4.4), the LQ design problem admits, for a proper choice of the cost, an analytical solution. Such a point is instrumental in showing (see Theorem 7.5) the existence of a control invariant region associated with the LQ controller.

Proposition 7.4. *Consider the following discrete-time infinite horizon LQ cost*

$$J(w(k)) = \sum_{k=0}^{\infty} \tilde{z}(k)^T Q \tilde{z}(k) + w(k)^T R w(k), \; Q \geq 0, \; R > 0 \tag{7.4.7}$$

for the linear system (7.4.4) *under the control law* (7.4.5). *If the LQ weighting matrices are* $Q = qI_{2\times 2}$ *and* $R = \rho I_{2\times 2}$ *with* $q, \rho > 0$, *the control gain* K *minimizing* $J(w(k))$ *is*

$$K = \kappa I, \quad \kappa = \frac{qT_s^2 + \sqrt{qT_s^2(T_s^2 q + 4\rho)}}{(qT_s^2 + \sqrt{qT_s^2(T_s^2 q + 4\rho)} + 2\rho)T_s} \tag{7.4.8}$$

Proof. Since for (7.4.4) the pair (A, B) is stabilizable, for any $Q = Q_z^T Q_z \geq 0$ and $R > 0$ such that (A, Q_z) is detectable, the gain K that ensure a finite and minimal value of $J(w(k))$ is

$$K = (B^T P B + R)^{-1} B^T P A \tag{7.4.9}$$

where $P = P^T > 0$ is the only symmetric and positive-definite solution to the following Discrete-time Algebraic Riccati Equation (DARE) [111]:

$$A^T P A - P - A^T P B (B^T P B + R)^{-1} B^T P A + Q = 0 \tag{7.4.10}$$

Since for (7.4.4), $A = I_{2\times 2}$ and $B = T_s I_{2\times 2}$, if $Q = qI_{2\times 2}$ and $R = \rho I_{2\times 2}$, the equation (7.4.10) can be analytically solved, obtaining

$$P = pI_{2\times 2}, \quad p = \frac{qT_s^2 + \sqrt{T_s^4 q^2 + 4T_s^2 q\rho}}{2T_s^2}$$

Finally, by substituting the obtained P into (7.4.9), the gain K in (7.4.8) is obtained, concluding the proof. □

The above-designed LQ controller does not ensure that the input constraints (7.4.3b) are fulfilled. Therefore, in what follows, a Positively Invariant (PI) region associated to the closed-loop dynamics (7.4.6) and complying with (7.4.3b) is derived exploiting the worst-case inner approximation $\hat{\mathcal{U}}$ of (7.4.3b).

Theorem 7.5. *Consider a constant reference signal $\overline{z}_r$ and the closed-loop error model* (7.4.6) *where the gain K is designed as in* (7.4.8). *The set*

$$\tilde{\mathcal{Z}} = \{\tilde{z} \in \mathbb{R}^2 \,|\tilde{z}^T S \tilde{z} \leq 1\}, \quad S = \frac{\kappa^2}{\hat{r}^2} I_{2\times 2} \tag{7.4.11}$$

is positively invariant and it complies with the input constraint set (7.4.3b) *irrespectively of any $\theta \in \mathbb{R}$ and $\varphi \in (-\pi/2, \pi/2)$.*

Proof. First, the set $\tilde{\mathcal{Z}} \subset \mathbb{R}^2$ of initial states $\tilde{z}(0) = z(0) - \overline{z}_r$ such that the LQ controller (7.4.5), with the gain K as in (7.4.8), does not violate the input constraints $\forall\, \theta \in \mathbb{R}, \varphi \in (-\pi/2, \pi/2)$ can be found substituting (7.4.5) in the worst-case input constraint set (7.3.1), obtaining (7.4.11). As shown in [112], a generic ellipsoidal set

$$\mathcal{E} = \{\tilde{z} \in \mathbb{R}^2 \,|\tilde{z}^T S \tilde{z} \leq 1\} \tag{7.4.12}$$

is positively invariant for (7.4.6) if the following matrix inequality is satisfied:

$$(A - BK)^T S (A - BK) - S < 0 \tag{7.4.13}$$

By noting that the set $\tilde{\mathcal{Z}}$ is equal to (7.4.12) if $S = K^T Q_u K = \frac{\kappa^2}{r^2} I_{2\times 2}$, the set (7.4.11) is positively invariant for (7.4.6) if

$$\frac{\kappa^2}{r^2}((A - BK)^T I (A - BK) - I) < 0 \tag{7.4.14}$$

Since A, B, K are scalar matrices, then $A - BK$ is also a scalar matrix containing on the diagonal the closed-loop eigenvalues, i.e., $A - BK = \lambda I_{2\times 2}$. Moreover, since $w(k) = -K\tilde{z}$ is stabilizing, then $|\lambda| < 1$, $(A - BK)^T I (A - BK) - I < 0$ and the condition (7.4.14) is fulfilled. □

Note that the controller (7.4.5) has been designed considering the offset-free model (7.4.4). Consequently, the actual tracking error (7.1.6a) is not guaranteed to remain in $\tilde{\mathcal{Z}}$ for any reference signal. However, since $\tilde{\mathcal{Z}}$ enjoys contractivity under (7.4.5), then $\tilde{\mathcal{Z}}$ is also robustly invariant for (7.1.6a) under a sufficient small disturbance $w_r(k)$ [113]. Moreover, under Assumption 7.1, $w_r(k)$ is bounded and can be over approximated (for a given reference trajectory) as a circular set $\mathcal{W}(r_d)$ of radius $r_d > 0$, where $\mathcal{W}(r_d) = \{w_r \in \mathbb{R}^2 \,|w_r^T W_{r_d} w_r \leq 1\}, W_{r_d} = \frac{1}{r_d^2} I_{2\times 2}$. Consequently, $\tilde{\mathcal{Z}}$ is robustly invariant $\forall\, w_r(k) \in \mathcal{W}(r_d)$ if $(A - BK)\tilde{\mathcal{Z}} \subseteq \tilde{\mathcal{Z}} \sim B\mathcal{W}(r_d)$ [113], where $\sim$ is the Minkowski set difference operator. Given that all the involved sets have a circular shape, the above set containment condition can be equivalently rewritten as the following matrix inequality (adapted from [113, Section 3]):

$$\eta^{-1}(A - BK)^T S^{-1}(A - BK) + (1 - \eta)^{-1} B^T W_{r_d}^{-1} B \leq S^{-1} \tag{7.4.15}$$

where $\eta = 1 - \sqrt{\xi}$ and ξ is the only repeated eigenvalue of the matrix $G^T B^T W_{r_d}^{-1} BG$, with G such that $G^T S^{-1} G = I_{2\times 2}$.

Theorem 7.6. *If the reference trajectory* (7.1.1) *complies with Assumption 7.1 and the condition* (7.4.15) *is fulfilled, then $\tilde{\mathcal{Z}}$ is robustly invariant for* (7.4.3) *under* (7.4.5)*. Moreover, if $\tilde{z}(0) \in \tilde{\mathcal{Z}}$, the nonlinear state-feedback tracking control law*

$$u(k) = -M^{-1}(\theta(k), \varphi(k))K(z(k) - z_r(k)) \tag{7.4.16}$$

ensures stable full-state tracking (see Definition 7.1) and input constraints (2.1.12) *fulfillment for the car-like vehicle* (2.1.10).

Proof. First, as proved in Theorem 7.5, for any initial error in $\tilde{\mathcal{Z}}$ and constant reference, the control law (7.4.5), with K as in (7.4.8), asymptotically stabilizes (7.4.6) while fulfilling (by worst-case) the input constraints (7.1.4). On the other hand, if $z_r(k)$ is a generic reference trajectory complying with Assumption 7.1 such that the corresponding disturbance $-w_r(k)$ in (7.4.3a) is bounded by a set $\mathcal{W}(r_d)$ that fulfills the condition (7.4.15), then, the control law (7.4.5) ensures that $\tilde{\mathcal{Z}}$ is robustly invariant irrespective of $w_r(k)$ and that the evolution of (7.4.3a) is stable. Consequently, given the results stated in Lemma 7.3, the nonlinear control law (7.4.16) (obtained applying the inverse transformation of (7.1.2) to (7.4.5)) ensures full-state tracking and constraint (2.1.12) fulfillment for the car-like model (2.1.10). □

7.5 Simulation Results

The effectiveness of the proposed tracking controller is thereafter discussed. The simulations have been performed in Matlab, where the car-like model (2.1.10) has been configured with the following set of parameters: $l = 0.5m$, $\overline{v} = 0.5\, m/s$, and $\overline{\omega} = \frac{\pi}{4}\, RAD/s$. Moreover, we have assumed that the coordinates $x(t)$ and $y(t)$, the orientation $\theta(t)$, and steering angle $\varphi(t)$ of the car are available for control purposes. The car-like model has been feedback linearized using the transformations (2.2.16)-(2.2.20) and setting $\Delta = 0.35m$. Consequently, by applying the formula in (7.3.1), the radius of the worst-case input constraint set (7.3.1) for the linearized model (7.1.6a) is $\hat{r} = 0.2252$. Then, the linearized car model has been discretized using a sampling time $T_s = 0.1$ sec, and the used LQ parameters are $q = 1$ and $\rho = 0.01$ (see Proposition 7.4). Consequently, the control gain K computed as in (7.4.8) is $K = 6.1803 I_{2\times 2}$, and the positively invariant region associated to this controller is obtained as in (7.4.11) and it is characterized by a shaping matrix $S = 753.1737 I_{2\times 2}$. The resulting positively invariant region is shown in Fig. 7.6 by means of a red solid line. In the performed simulation, the reference signal has an eight-shape and its timing law in the $x - y$ plane is (see the dashed red line in Fig. 7.4)

$$x_r(t) = \sin(t/10)\,,\; y_r(t) = \sin(t/20)$$

The reference velocities $\dot{x}_r$, $\dot{y}_r$ and accelerations $\ddot{x}_r$, $\ddot{y}_r$ have been obtained from the above via differentiation, and the corresponding orientation θ_r and steering angle ϕ_r computed via (7.1.1). For the chosen reference and control parameters, we have that $r_d = 0.1838$, $\eta = 0.4956$ and that the condition (7.4.15) is fulfilled. Moreover, the car's initial conditions are assumed to be $x_0 = 0$, $y_0 = -0.035$, $\theta_0 = 0$, $\varphi_0 = 0$.

The obtained simulation results are collected in Figs. 7.4-7.6. Specifically, In Fig. 7.4, it is possible to appreciate the tracking capability of the considered controller, even though the initial car's initial point (see the yellow star) is far from the desired trajectory. Fig. 7.5 shows that the car's linear and steering velocities ($v(t)$ and $\omega(t)$) fulfill the prescribed constraints (see the dashed red line in the first two subplots) regardless of the robot's angles $\theta(t)$ and $\varphi(t)$, confirming the correctness of the used worst-case approximation of the car's constraints. Moreover, the small discrepancies between the angles of the vehicle $\theta(k)$ and $\varphi(k)$ and their reference value $\theta_r(r)$ and $\varphi_r(k)$, testify that the evolution of the tracking error internal dynamics is bounded and actually in the proximity of the reference trajectories. Finally, Fig. 7.6 shows that the linear system's tracking error trajectory $\tilde{z}(t) = z(t) - \hat{z}_r(t)$ (blue solid line) is confined in $\tilde{\mathcal{Z}}$ (red solid circle), confirming that the joint use of the proposed feasibility governor and constrained tracking controller make $\tilde{\mathcal{Z}}$ positively invariant.

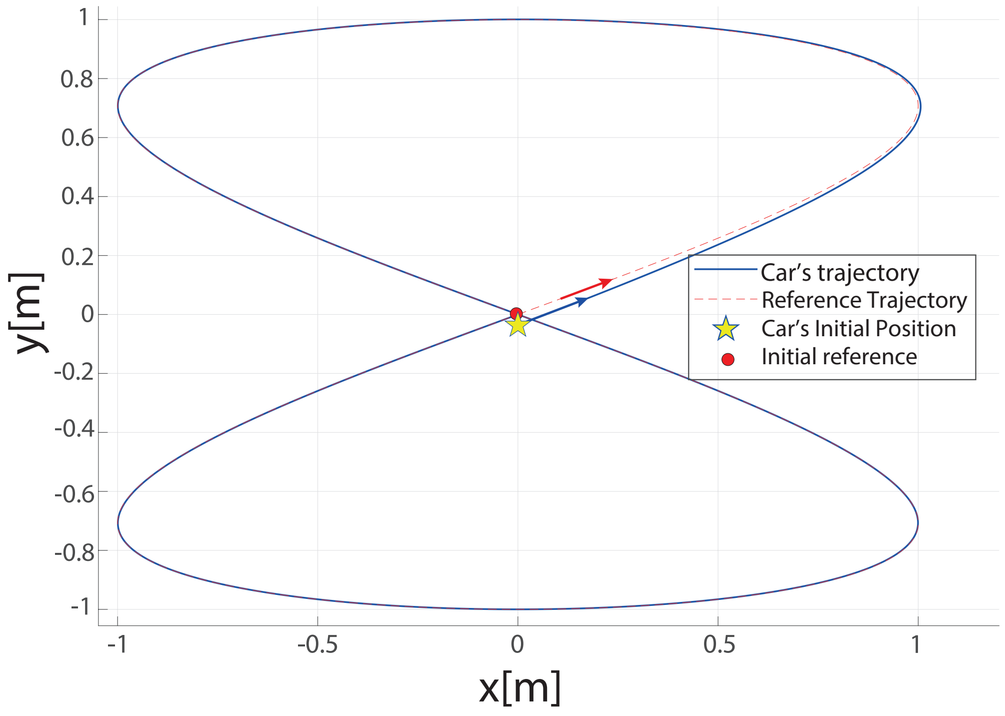


Figure 7.4: Trajectory tracking.

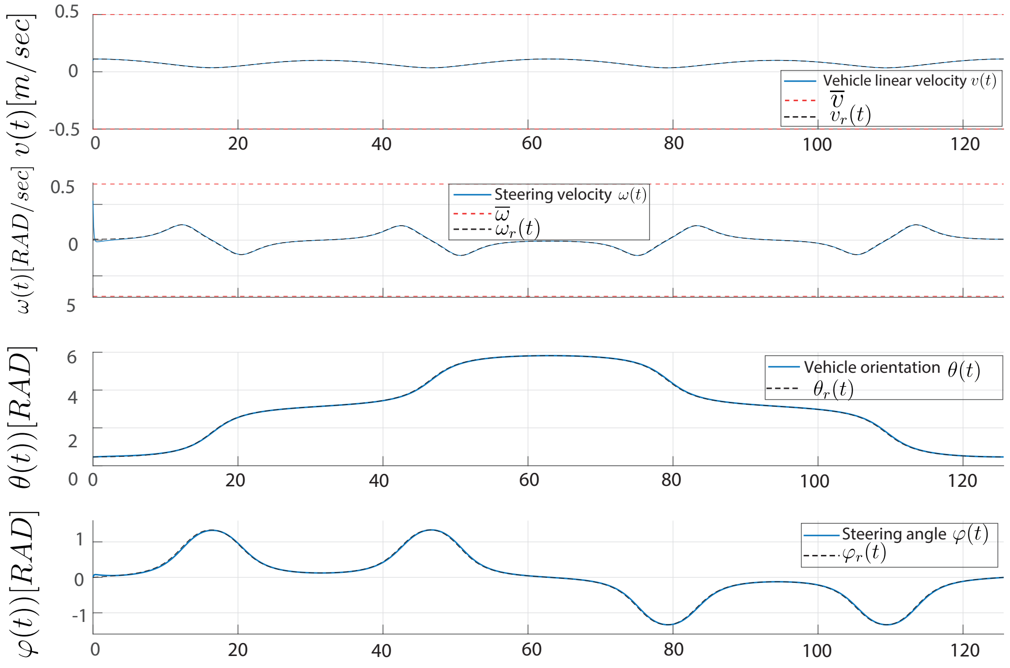


Figure 7.5: Car's velocities and angles.

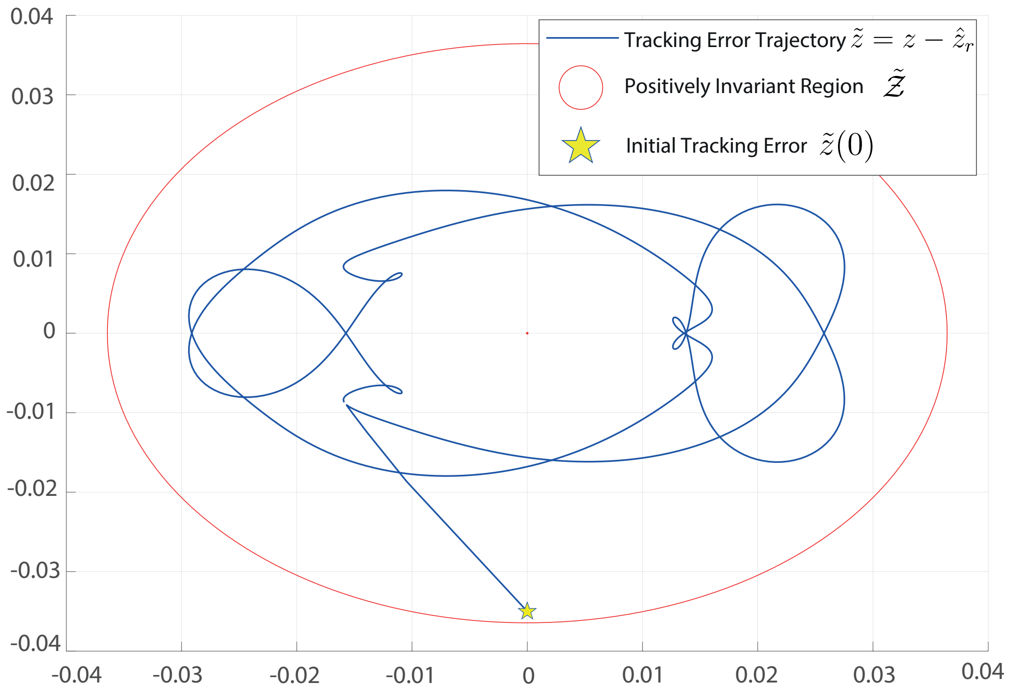


Figure 7.6: Trajectory tracking error for the feedback linearized car-like model and positively invariant region $\tilde{\mathcal{Z}}$.

7.6 Conclusions

In this chapter, we have shown that it is possible to design a constrained tracking controller with associated control invariant region for an input constrained feedback linearized car-like vehicle. This has been obtained by properly deriving a worst-case approximation of the time-varying input constraints arising when the nonlinear car model is feedback linearized. Such a characterization of the constraint set has been then exploited to design a state-feedback controller where the associated invariant region is analytically derived. Although the proposed solution can be used as a standalone constrained tracking controller, its features might allow using it as the terminal controller of dual-mode mode predictive controller strategies. Future studies will be devoted to investigating such a possibility.

8 A Feedback-Linearized Model Predictive Control Strategy for Input Constrained Self-Driving Cars

A novel real-time affordable solution is thereafter proposed to solve the trajectory tracking control problem for self-driving cars subject to longitudinal and steering angular velocities constraints. To this end, we develop a dual-mode Model Predictive Control solution starting from an input-output feedback linearizated description of the vehicle kinematics. First, we derive the state-dependent input constraints acting on the linearized model and characterize their worst-case time-invariant inner approximation. Then, a dual-mode MPC is derived to be real-time affordable and ensuring, by design, constraints fulfillment, recursive feasibility, and uniformly ultimate boundedness of the tracking error in an ad-hoc built robust control invariant region. The proposed invariant control design is based on the arguments outlined in Chapter 7. The approach's effectiveness and performance are experimentally validated via laboratory experiments on a Quanser Qcar. The obtained results show that the proposed solutions is computationally affordable and with tracking capabilities that outperform two alternative control schemes.

8.1 Problem's statement

Let's consider a smooth reference trajectory described in terms of Cartesian positions $x_r(t)$, $y_r(t)$, velocities $\dot{x}_r(t)$, $\dot{y}_r(t)$ and accelerations $\ddot{x}_r(t)$, $\ddot{y}_r(t)$ and jerks $\dddot{x}_r(t)$, $\dddot{y}_r(t)$ of the rear axis center of the car. The corresponding reference car's state is denoted as $q_r(t) = [x_r(t), y_r(t), \theta_r(t), \varphi_r(t)]^T$, where $\theta_r(t)$ and $\varphi_r(t)$ are the heading and steering angles associated to the given trajectory which can be computed as [114]:

$$\begin{aligned} \theta_r(t) = & \quad \mathrm{ATAN}_2\left(\tfrac{\dot{y}_r(t)}{v_r(t)}, \tfrac{\dot{x}_r(t)}{v_r(t)}\right) \\ \varphi_r(t) = & \quad \arctan\left(\tfrac{l(\ddot{y}_r(t)\dot{x}_r(t) - \ddot{x}_r(t)\dot{y}_r(t))}{v_r(t)^3}\right) \end{aligned} \tag{8.1.1}$$

On the other hand, the reference inputs associated with the trajectory are given by $u_r(t) = [v_r(t),\, \omega_r(t)]^T$, where

$$\begin{aligned} v_r(t) = & \quad \sqrt{\dot{x}_r(t)^2 + \dot{y}_r(t)^2} \\ \omega_r(t) = & \quad l v_r \tfrac{(\dddot{y}_r\dot{x}_r - \dddot{x}_r\dot{y}_r)v_r^2 - 3(\ddot{y}_r\dot{x}_r - \ddot{x}_r\dot{y}_r)(\dot{x}_r\ddot{x}_r + \dot{y}_r\ddot{y}_r)}{v_r^6 + l^2(\ddot{y}_r\dot{x}_r - \ddot{x}_r\dot{y}_r)^2} \end{aligned} \tag{8.1.2}$$

Notice that the time dependency has been omitted on the right-hand side for compactness.

Assumption 8.1. The reference trajectory $q_r(t) = [x_r(t), y_r(t), \theta_r(t), \varphi_r(t)]^T$ for (2.1.10) is uniformly bounded and smooth, i.e., $\exists \Gamma > 0 : \|q_r(t)\| < \Gamma, \forall t \geq 0, \ q_r(t) \in \mathcal{C}^2$. $\square$

Problem 6. *Design a constrained state feedback controller*

$$u(t) = \phi(t, q(t), q_r(t), u_r(t)) \tag{8.1.3}$$

such that $u(t) \in \mathcal{U}_{car}, \forall\, t \geq 0$ *and stable full-state tracking is achieved, i.e.,*

$$\exists \delta > 0,\, t_0 \geq 0 \ \ s.t. \ \|\tilde{q}(t_0)\| < \delta \implies \|\tilde{q}(t)\| < \varepsilon,\, \forall t \geq t_0$$

where $\tilde{q}(t) = q(t) - q_r(t)$ *is the tracking error.*

8.2 Proposed Solution

Here, the considered problem is addressed by combining feedback-linearization and MPC arguments. First, the control problem is described as a standard nonlinear MPC over a finite prediction horizon. Then, the nonconvex nature of the underlying MPC optimization is analyzed, and a novel predictive framework based on feedback linearization is proposed to recover a convex optimization problem that fulfills constraints while guaranteeing a bounded tracking error.

8.2.1 Nonlinear MPC

Let's define $\tilde{u} = u - u_r$ and a LQ cost

$$\begin{aligned} J_N(k, \tilde{q}(k), \tilde{u}(k)) = \sum_{i=0}^{N-1} & \tilde{q}(k+i+1|k)^T Q \tilde{q}(k+i+1|k) + \\ & + \tilde{u}(k+i|k)^T R \tilde{u}(k+i|k) \end{aligned}$$

where $N > 0$ is the prediction horizon, and $Q = Q^T \geq 0$, $Q \in \mathbb{R}^n$, $R = R^T > 0$, $R \in \mathbb{R}^m$ are weighting matrices for the state and control input tracking errors, respectively. Then, the optimal control law that minimizes the defined cost function over the prediction horizon

N can be computed as:

$$u(k) = \underset{u(k),\ldots,u(k+N-1)}{\arg\min} \; J_N(k, \tilde{q}(k), \tilde{u}(k)) \; s.t. \tag{8.2.1a}$$

$$q(k+i+1|k) = f_{car}(q(k+i|k), u(k+i)) \tag{8.2.1b}$$

$$u(k+i) \in \mathcal{U}_{car} \tag{8.2.1c}$$

$$\tilde{q}(k+N|k) \in \mathcal{Q}_N \tag{8.2.1d}$$

$$i = 0, 1 \ldots N-1$$

where $\mathcal{Q}_N$ is a predefined set, PI with respect to an offline-designed feedback terminal control law $u_N(k) = \phi_N(k, q(k), q_r(k), u_r(k)) \in \mathcal{U}_{car}$, $\forall k \geq N$. The above is known as dual-mode MPC, i.e., for the first N steps, the control law is obtained by solving the above optimization and applying the optimal solution in a receding horizon fashion, i.e., only the first sample $u(k)$ is applied to the system (2.1.10) and the optimization is solved at any sampling time. Then, once the error trajectory $\tilde{q}(k)$ reaches $\mathcal{Q}_N$, the control law $u_N(k)$ associated to $\mathcal{Q}_N$ is used.

Remark 8.1. The above dual-mode MPC strategy guarantees stability and input constraint fulfillment, for any initial condition $\tilde{q}(0)$ such that the optimization problem (8.2.1) is feasible [94]. Consequently, under the effect of the dual-mode MPC control law, the tracking error is bounded with respect to any trajectory complying with Assumption 8.1.

Remark 8.2. Although appealing, optimization (8.2.1) suffers from the following drawbacks:

- The optimization problem is highly nonconvex due to the presence of the constraints (8.2.1b) and (8.2.1d). Moreover, the obtained solutions may suffer from local minima problems [72];
- The computational burden associate to (8.2.1), especially for large prediction horizon N, may not allow the real-time implementation of the control scheme;
- The computation of $\mathcal{Q}_N$ and associated state-feedback controller u_N is not trivial for the nonlinear vehicle kinematic model (2.1.10).

Motivated by the above drawbacks, in what follows a novel MPC formulation based on feedback linearization arguments is proposed. In particular, first, the input-output linearization proposed in [114] is exploited to obtain a linear description of the car's kinematics.

Then, inspired by the idea introduced in [88], the time-varying input constraints acting on the linearized model and their worst-case realization are analytically characterized. Finally, the obtained constrained model and worst-case arguments are used to design a tracking control strategy that ensures stability, recursive feasibility, and input constraint fulfillment.

8.2.2 Input-Output Feedback Linearization

Here, the input-output feedback-linearization introduced in [114] is used to obtain a linearized description of the car's kinematic model.

Let's define two new outputs

$$z = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} x + l\cos(\theta) + \Delta\cos(\theta+\varphi) \\ y + l\sin(\theta) + \Delta\sin(\theta+\varphi) \end{bmatrix} \tag{8.2.2}$$

representing the Cartesian position of a point P at a distance $\Delta > 0$ from the center of front wheels' axis, and a new state vector $\eta = [\eta_1,\, \eta_2]^T = [\theta,\, \varphi]^T$. Then, by resorting to the following input transformation depending on η:

$$\begin{gathered} w = M(\eta)u, \\ M(\eta) = \begin{bmatrix} \cos(\eta_1) - \tan(\eta_2)(\sin(\eta_1) + \frac{\Delta}{l}s_1) & -\Delta s_1 \\ \sin(\eta_1) + \tan(\eta_2)(\cos(\eta_1) + \frac{\Delta}{l}c_1) & \Delta c_1 \end{bmatrix} \end{gathered} \tag{8.2.3}$$

where $s_1 = \sin(\eta_1+\eta_2) \quad c_1 = \cos(\eta_1+\eta_2)$, the kinematic model (2.1.10) is recast into

$$\dot{z} = w \tag{8.2.4a}$$

$$\dot{\eta} = O(\eta)w \tag{8.2.4b}$$

where

$$O(\eta) = \begin{bmatrix} \frac{\sin(\eta_2)c_1}{l} & \frac{\sin(\eta_2)s_1}{l} \\ \frac{-\sin(\eta_2)c_1}{l} - \frac{s_1}{\Delta} & \frac{-\sin(\eta_2)s_1}{l} + \frac{c_1}{\Delta} \end{bmatrix}$$

Notice that (8.2.4a) defines a two-single-integrator model subject to a decoupled nonlinear internal dynamics (8.2.4b). The above decoupled system can be discretized via forward Euler discretization method, obtaining:

$$z(k+1) = Az(k) + Bw(k), \quad A = I_{2\times 2}, \quad B = T_s I_{2\times 2} \tag{8.2.5a}$$

$$\eta(k+1) = \eta(k) + T_s O(\eta(k))w(k) \tag{8.2.5b}$$

Property 8.1. The input output feedback-linearization (8.2.2)-(8.2.4) and forward Euler discretization method commute for the car's kinematic model (2.1.10).

Proof. Let's consider the output transformation (8.2.2) and its first-order derivative

$$\begin{aligned}\dot{z}_1 &= \dot{x} - l\sin(\theta)\dot{\theta} - \Delta\sin(\theta+\varphi)(\dot{\theta}+\dot{\varphi})\\ \dot{z}_2 &= \dot{y} + l\cos(\theta)\dot{\theta} + \Delta\cos(\theta+\varphi)(\dot{\theta}+\dot{\varphi})\end{aligned}$$

Then, under forward Euler discretization arguments one obtains

$$\begin{aligned}\frac{z_1(k+1)-z_1(k)}{T_s} &= \frac{x(k+1)-x(k)}{T_s} - l\sin(\theta)\frac{\theta(k+1)-\theta(k)}{T_s} -\\ &\quad -\Delta\sin(\theta+\varphi)\left[\frac{\theta(k+1)-\theta(k)}{T_s}+\frac{\varphi(k+1)-\varphi(k)}{T_s}\right]\\ \frac{z_2(k+1)-z_2(k)}{T_s} &= \frac{y(k+1)-y(k)}{T_s} + l\cos(\theta)\frac{\theta(k+1)-\theta(k)}{T_s} +\\ &\quad +\Delta\cos(\theta+\varphi)\left[\frac{\theta(k+1)-\theta(k)}{T_s}+\frac{\varphi(k+1)-\varphi(k)}{T_s}\right]\end{aligned} \tag{8.2.6}$$

Let's now consider the discrete-time kinematic model of the car, obtained via forward-Euler discretization of (2.1.10), i.e.,

$$\begin{aligned}x(k+1) &= x(k) + T_s v(k)\cos(\theta(k))\\ y(k+1) &= y(k) + T_s v(k)\sin(\theta(k))\\ \theta(k+1) &= \theta(k) + T_s \tfrac{v(k)}{l}\tan(\varphi(k))\\ \varphi(k+1) &= \varphi(k) + T_s\omega(k)\end{aligned} \tag{8.2.7}$$

By substituting $x(k+1)$, $y(k+1)$, $\theta(k+1)$. $\varphi(k+1)$ with the right-hand sides of (8.2.7) and rewriting the equation in a compact form, the resulting discrete-time evolution of z_1 and z_2 is:

$$z(k+1) = z(k) + T_s M(\eta(k))u(k) \tag{8.2.8}$$

Finally, by using the input transformation (8.2.3), the discrete-time feedback linearized system (8.2.5) is obtained, which is equal to the discrete-time system obtained by discretization of (8.2.4a). Similarly, the nonlinear internal dynamics (8.2.4b) can be discretized obtaining (8.2.5b). Hence, input-output linearization and discretization commute. □

8.2.3 Tracking Error Model and Input Constraint Characterization

Here, the feedback-linearized tracking error model is formally derived. By applying the output transformation (8.2.2), the reference output for the input-output linearized system (8.2.4) is given by

$$z_r = \begin{bmatrix} x_r + lcos(\theta_r) + \Delta\cos(\theta_r + \varphi_r) \\ y_r + lsin(\theta_r) + \Delta\sin(\theta_r + \varphi_r) \end{bmatrix} \tag{8.2.9}$$

Similarly, reference inputs for (8.2.4) can be computed via (8.2.3), obtaining

$$w_r(t) = M(\theta_r(t), \varphi_r(t))u_r(t) \tag{8.2.10}$$

By defining the error vectors $\tilde{z} = z - z_r$ and $\tilde{w} = w - w_r$, $\tilde{\eta} = \eta - \eta_r$, $\eta_r = [\theta_r, \phi_r]^T$, the input-output linearized and internal tracking error dynamics are given by:

$$\dot{\tilde{z}}(t) = \tilde{w}(t) \tag{8.2.11a}$$

$$\dot{\tilde{\eta}}(t) = \kappa(\tilde{\eta}, \tilde{w}, \eta_r, w_r, t) = O(\eta(t))w(t) - O(\eta_r(t))w_r(t) \tag{8.2.11b}$$

which can be discretized by resorting to the Euler forward method and re-written as

$$\tilde{z}(k+1) = A\tilde{z}(k) + Bw(k) - Bw_r(k), \tag{8.2.12a}$$

$$A = I_{2\times 2}, \quad B = T_s I_{2\times 2} \tag{8.2.12b}$$

$$\tilde{\eta}(k+1) = \tilde{\eta}(k) + T_s\kappa(\tilde{\eta}(k), \tilde{w}(k), \eta_r(k), w_r(k), k) \tag{8.2.12c}$$

Remark 8.3. Since the reference trajectory is assumed to be bounded, then also $w_r(k)$ is bounded and the set of admissible $w_r(k)$ can be over-approximated by a ball $\mathcal{W}_r \subset \mathbb{R}^2$ of radius r_d, i.e.,

$$w_r \in \mathcal{W}_r = \{w_r \in \mathbb{R}^2 : w_r^T W_r^{-1} w_r \leq 1\},\ W_r = r_d^2 I_{2\times 2} \tag{8.2.13}$$

Given the feedback linearized tracking error dynamics, the following lemma establishes sufficient conditions for bounded internal dynamics.

Lemma 8.1. *If the reference trajectory $q_r(t)$ complies with Assumption 8.1, $v_r(t)$ and $\omega_r(t)$ satisfies* (8.1.2) *and* $0 < v_r(t) \leq V > 0, \forall t$ *and* $\forall|\varphi_r(t)| \leq \frac{\pi}{2}$, $\forall t$, *then the tracking-error zero dynamics* $\dot{\tilde{\eta}} = \kappa(\eta, 0, \eta_r, w_r, t)$ *are asymptotically stable [93, Theorems 1-3]. Consequently, if* (8.2.11a) *is stable, stable full-state tracking is achieved [93].* □

By applying the transformation (8.2.3) to the input constraints (2.1.12), the the tracking-error dynamics (8.2.12) are subject to the following time-varying polyhedral input constraints, depending on the internal dynamics state η i.e.,

$$w \in \mathcal{U}(\eta) = \{w \in \mathbb{R}^2 \colon L(\eta) w \leq g\},\ L(\eta) = TM^{-1}(\eta) \tag{8.2.14}$$

The following lemma analytically characterizes the polyhedral set $\mathcal{U}(\eta)$, which rotates and resizes in function of η.

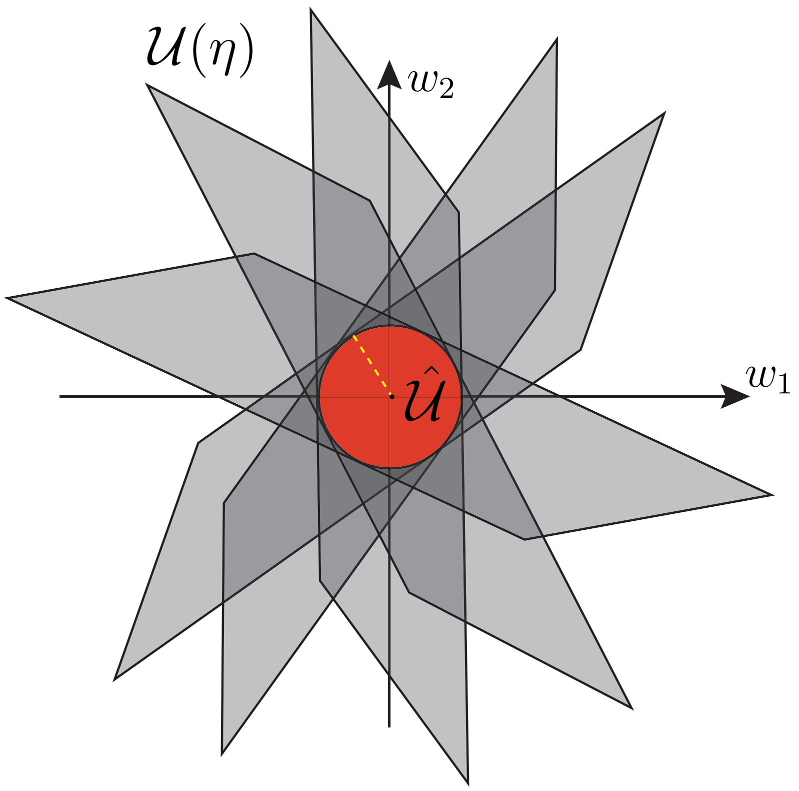


Figure 8.1: Time-varying input constraint set and its worst-case approximation

Lemma 8.2. *The polyhedral input constraint set* (8.2.14) *is a time-varying parallelogram that admits the following worst-case circular inner approximation (see Fig. 8.1):*

$$\begin{aligned} \hat{\mathcal{U}} &= \bigcap_{\forall \eta} \mathcal{U}(\eta) = \{w \in \mathbb{R}^2 \,|w^T w \leq \hat{r}^2\}, \\ \hat{r} &= \min\left(\tfrac{\Delta l \overline{\omega}}{\sqrt{\Delta^2 + l^2}}, \overline{v}\right) \end{aligned} \tag{8.2.15}$$

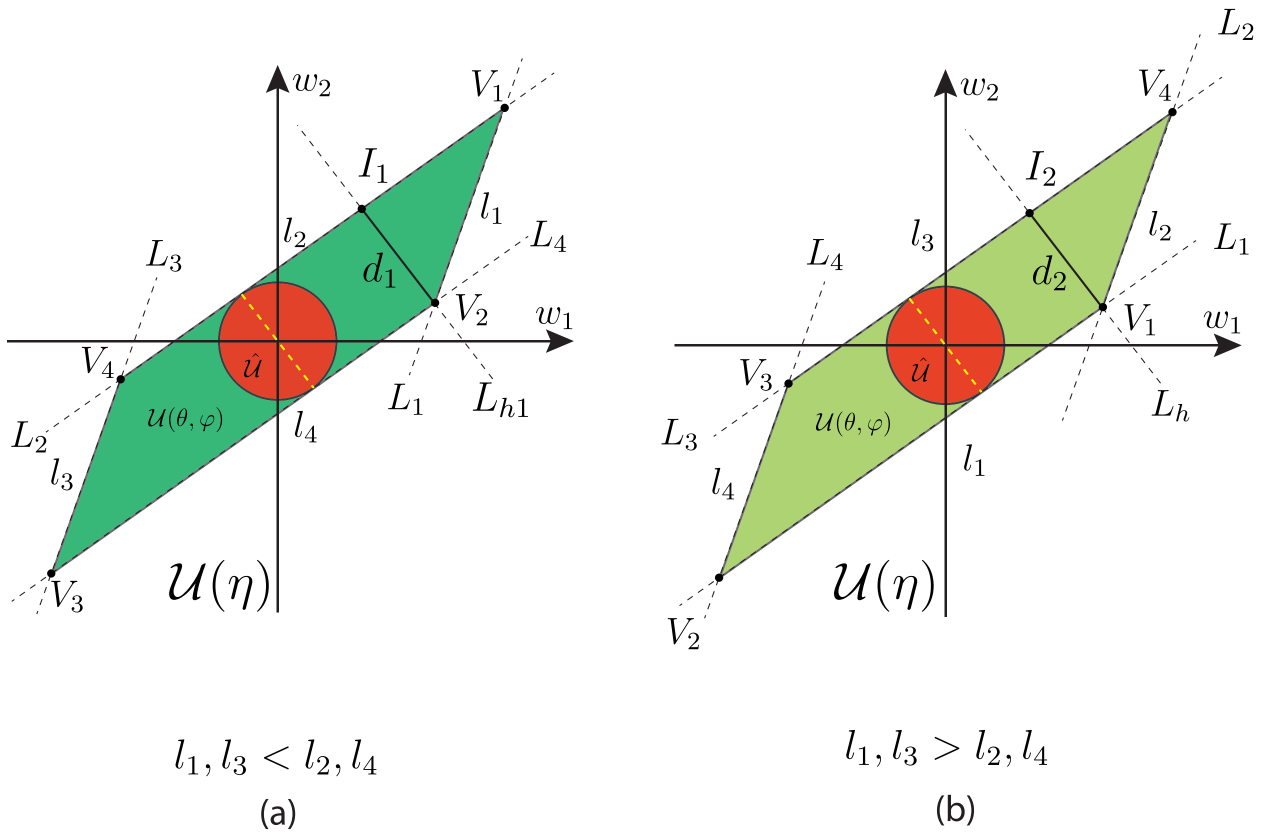


Figure 8.2: Possible side length configurations for $\mathcal{U}(\eta)$

Proof. By defining $s_2 = \sin(\eta_1 + 2\eta_2)$ and $c_2 = \cos(\eta_1 + 2\eta_2)$, the shaping matrix of the polyhedral set $\mathcal{U}(\eta)$ can be re-written as:

$$L(\eta) = \begin{bmatrix} -\frac{\cos(\eta_1)+c_2}{2} & -\frac{\sin(\eta_1)+s_1}{2} \\ \frac{\Delta s_1 - \Delta\sin(\eta_1)+2ls_2}{2\Delta l} & \frac{-\Delta c_2+\Delta\cos(\eta_1)-2lc_1}{2\Delta l} \\ \frac{\cos(\eta_1)+c_2}{2} & \frac{\sin(\eta_1)+s_1}{2} \\ -\frac{\Delta s_1-\Delta\sin(\eta_1)+2ls_2}{2\Delta l} & -\frac{-\Delta c_2+\Delta\cos(\eta_1)-2lc_1}{2\Delta l} \end{bmatrix}$$

By intersecting the four hyperplanes, the four vertices have the following analytical expression:

$$V_1(\eta) = \begin{bmatrix} \frac{\Delta s_1 l\overline{\omega} + \Delta \sin(\eta_1) l\overline{\omega} + \Delta\overline{v}\cos(\eta_1) - \Delta\overline{v}c_2 - 2c_1 l\overline{v}}{(2l\cos(\eta_2))} \\ \frac{-\overline{\omega}\Delta l\cos(\eta_1) - \overline{\omega}\Delta l c_2 - \Delta s_1\overline{v} + \Delta\sin(\eta_1)\overline{v} - 2s_2 l\overline{v}}{2l\cos(\eta_2)} \end{bmatrix} \tag{8.2.16a}$$

$$V_2(\eta) = \begin{bmatrix} \frac{-\Delta s_1 l\overline{\omega} - \Delta \sin(\eta_1) l\overline{\omega} + \Delta\overline{v}\cos(\eta_1) - \Delta\overline{v}c_2 - 2c_1 l\overline{v}}{2l\cos(\eta_2)} \\ \frac{\overline{\omega}\Delta l\cos(\eta_1) + \overline{\omega}\Delta l c_2 - \Delta s_1\overline{v} + \Delta\sin(\eta_1)\overline{v} - 2s_2 l\overline{v}}{2l\cos(\eta_2)} \end{bmatrix} \tag{8.2.16b}$$

$$V_1(\eta) = -V_3(\eta), \quad V_2(\eta) = -V_4(\eta) \tag{8.2.16c}$$

By computing the Cartesian distances between the vertices, each side of the parallelogram has the following length:

$$\begin{aligned} l_1 = l_3 &= 2\sqrt{\Delta^2\overline{\omega}^2} \\ l_2(\eta_2) = l_4(\eta_2) &= 2\sqrt{\frac{v_1^2(-\Delta^2\cos(2\eta_2)+\Delta^2+2l^2)}{l^2(\cos(2\eta_2)+1)}} \end{aligned} \tag{8.2.17}$$

It can also be noted that the angular coefficients of the four lines L_1, L_2, L_3, L_4 defining the polyhedron, namely $m_1(\eta_1,\eta_2)$, $m_2(\eta_1,\eta_2), m_3(\eta_1,\eta_2)$, $m_4(\eta_1,\eta_2)$, are such that $m_1(\eta_1,\eta_2) = m_3(\eta_1,\eta_2)$, and $m_2(\eta_1,\eta_2) = m_4(\eta_1,\eta_2)$, $\forall\eta_1 \in \mathbb{R}, \eta_2 \in [-\overline{\varphi},\overline{\varphi}]$. Consequently, $\mathcal{U}(\theta)$ is a time-varying parallelogram whose side lengths depend on the state variable η_2, and on the car's parameters Δ, l, $\overline{v}$, $\overline{\omega}$.

In order to find the radius of the smallest circle inscribed in the polyhedral set, we resort to geometric arguments. By referring to Fig. 8.2, two different cases must be considered *(a)* $l_1, l_3 < l_2, l_4$ and *(b)* $l_1, l_3 > l_2, l_4$. Depending on the specific case, the diameter of the inscribed circular set can be found either as the distance of vertex V_2 from the point I_1, or the distance of vertex V_1 from the point I_2. Note that I_1 is the intersection of the line L_2 and the orthogonal to L_2 crossing V_2 (case *(a)*), whereas I_2 is the intersection of the line L_3 and the orthogonal to L_3 crossing V_1 (case *(b)*).

Formally, the lines L_2 and L_3 crossing (V_1, V_4) and (V_4, V_3), respectively are

$$\begin{aligned} L_2: \quad w_2 &= m_2(\eta_1,\eta_2)w_1 + h_2(\eta_1,\eta_2) \\ m_2(\eta_1,\eta_2) &= -\tfrac{L[2,1]}{L[2,2]}, \quad h_2(\eta_1,\eta_2) = \tfrac{g[2]}{L[2,2]} \end{aligned} \tag{8.2.18}$$

$$\begin{aligned} L_3: \quad & w_2 = m_3(\eta_1,\eta_2)w_1 + h_3(\eta_1,\eta_2) \\ m_3(\eta_1,\eta_2) = -\tfrac{L[3,1]}{L[3,2]}, & \quad h_3(\eta_1,\eta_2) = \tfrac{g[3]}{L[3,2]} \end{aligned} \tag{8.2.19}$$

□

Moreover, by resorting to simple geometric arguments, the equations of the lines L_{h1} and L_{h2} are:

$$L_{h1}: \quad w_2 - V_2\,[2] = -\frac{1}{m_2(\eta_1,\eta_2)}(w_1 - V_2\,[1]) \tag{8.2.20}$$

$$L_{h2}: \quad w_2 - V_1\,[2] = -\frac{1}{m_3(\eta_1,\eta_2)}(w_1 - V_1\,[1]) \tag{8.2.21}$$

Then, the points I_1 and I_2 can be computed intersecting L_2 with L_{h1} and L_3 with L_{h2}, obtaining

$$I_1 = \begin{bmatrix} -m_2(\eta_1,\eta_2) & 1 \\ \frac{1}{m_2(\eta_1,\eta_2)} & 1 \end{bmatrix}^{-1} \begin{bmatrix} h_2(\eta_1,\eta_2) \\ \frac{V_2[1]}{m_2(\eta_1,\eta_2)} + V_2\,[2] \end{bmatrix} \tag{8.2.22}$$

$$I_2 = \begin{bmatrix} -m_3(\eta_1,\eta_2) & 1 \\ \frac{1}{m_3(\eta_1,\eta_2)} & 1 \end{bmatrix}^{-1} \begin{bmatrix} h_3(\eta_1,\eta_2) \\ \frac{V_1[1]}{m_3(\eta_1,\eta_2)} + V_1\,[2] \end{bmatrix} \tag{8.2.23}$$

By noticing that the inscribed circles have diameters equal to $d_1 = \overline{V_2 I_1}$ (case *(a)*) and $d_2 = \overline{V_1 I_2}$ (case *(b)*), the radii, namely $r_1(\eta_2)$ and $r_2(\eta_2)$, are

$$\begin{aligned} r_1(\eta_2) &= \tfrac{1}{2}d = \tfrac{1}{2}\sqrt{(I_1 - V_2)^T(I_1 - V_2)} = \\ &= \tfrac{\Delta l \overline{\omega}}{\sqrt{\Delta^2 - \Delta^2\cos(\eta_2)^2 + l^2}} \end{aligned} \tag{8.2.24}$$

$$\begin{aligned} r_2(\eta_2) &= \tfrac{1}{2}d = \tfrac{1}{2}\sqrt{(I_2 - V_1)^T(I_2 - V_1)} = \\ &= \sqrt{\tfrac{\overline{v}^2}{\cos(\eta_2)^2}} \end{aligned} \tag{8.2.25}$$

Furthermore, since $\eta_2 \in [-\overline{\eta_2}, \overline{\eta_2}]\,, \overline{\eta_2} < \frac{\pi}{2}$, the minimum value of $r_1(\eta_2)$ and $r_2(\eta_2)$ is obtained for $\eta_2 = 0$, that it is equals to

$$r_1 = r_1(0) = \frac{\Delta l \overline{\omega}}{\sqrt{\Delta^2 + l^2}}, \quad r_2 = r_2(0) = \overline{v}$$

Consequently, (8.2.15) defines the worst-case circle inscribed in $\mathcal{U}(\eta), \forall \eta$, concluding the proof.

8.2.4 Robust Invariant Control Design

By using similar arguments to the ones exploited in [88], the linearized tracking error dynamics can be exploited to design a state feedback controller that fulfills the prescribed time-varying and state-dependent input constraints in a properly defined robust invariant region.

Proposition 8.3. *The circular set*

$$\Sigma_N = \{\tilde{z} \in \mathbb{R}^2 \,|\tilde{z}^T S \tilde{z} \leq 1\}, \quad S = \frac{1}{\hat{r}^2} K^T K \tag{8.2.26}$$

is RPI for (8.2.12a) *under the state-feedback controller*

$$w(k) = K\tilde{z}(k) + \hat{w}_r(k) \tag{8.2.27}$$

where $\hat{w}_r(k)$ is the optimal solution of the following Quadratic Programming (QP) problem:

$$\hat{w}_r(k) = \arg\min_{\hat{w}_r} \|\hat{w}_r - w_r(k)\|_2^2 \quad s.t. \tag{8.2.28a}$$

$$K\tilde{z}(k) + \hat{w}_r \in \mathcal{U}(\eta) \tag{8.2.28b}$$

and K is such that

$$\lambda^{-1} A_{cl}^T S^{-1} A_{cl} + (1-\lambda)^{-1} B^T W_{r_d}^{-1} B \leq S^{-1}. \tag{8.2.29}$$

where $A_{cl} = A - BK$, $\lambda = 1 - \sqrt{\xi}$ and ξ is the only repeated eigenvalue of the matrix $G^T B^T W_{r_d}^{-1} BG$, with G such that $G^T S^{-1} G = I_{2\times 2}$.

Proof. For the disturbance-free model (i.e. obtained from (8.2.12a) when $w_r(k) = 0, \forall\, k$), any stabilizing controller $w(k) = K\tilde{z}(k)$ fulfills the input constraint for any $\tilde{z} \in \Sigma_N$ where Σ_N is as in (8.2.26). Specifically, the set Σ_N is obtained by plugging the state-feedback controller $w(k) = K\tilde{z}(k)$ into the circular region (8.2.15), representing the worst-case set of admissible input for (8.2.12). Moreover, since in the disturbance-free case, (8.2.29) reduces to a standard Lyapunov inequality $(A - BK)^T S^{-1}(A - BK) - S^{-1} \leq 0$, then if K fulfills (8.2.29) then Σ_N is also a positively invariant region.

On the other hand, in the presence of $w_r(k) \neq 0$ and under the control law $w(k) =$

$K\tilde{z}(k) + \hat{w}_r(k)$, the closed-loop system is

$$\begin{aligned} \tilde{z}(k+1) &= (A - BK)\tilde{z}(k) + B(\hat{w}_r(k) - w_r(k)) \\ &= A_{cl}\tilde{z}(k) + Bw_d(k) \end{aligned} \tag{8.2.30}$$

with $w_d = \hat{w}_r(k) - w_r(k)$. If $\hat{w}_r(k)$ is given by the solution of (8.2.28), then the control law $w(k) = K\tilde{z}(k) + \hat{w}_r(k)$ fulfils the input constraints for any $\tilde{z} \in \Sigma_N$. Moreover, w_d is bounded inside the set $\mathcal{W}_r$ (with the worst-case happening when $\hat{w}_r(k) = 0$). Finally, as proven in [113, Section 3], if K fulfils (8.2.29), then Σ_N is RPI for (8.2.30), concluding the proof.

□

Remark 8.4. In [88], the authors have proposed an analytical design of the state feedback controller such that it is optimal for a given linear quadratic cost. Also, it is worth mentioning that (8.2.29) represents a sufficient condition to ensure RPI. For a more exhaustive discussion on necessary and sufficient conditions, the interested reader may refer to [113].

8.2.5 Feedback Linearized Model Predictive Control

Under input-output linearization arguments, optimization (8.2.1) can be equivalently rewritten as follows:

$$\min_{w(k),\ldots,w(k+N-1)} J_N(k, \tilde{z}(k), \tilde{w}(k)) \tag{8.2.31a}$$

$$\tilde{z}(k+i+1|k) = A\tilde{z}(k+i|k) + Bw(i) - Bw_r(i) \tag{8.2.31b}$$

$$\eta(k+i+1|k) = \eta(k+i|k) + T_s O(\eta(k+i|k))w(k+i) \tag{8.2.31c}$$

$$L(\eta(k+i|k))w(k+i) \leq g \tag{8.2.31d}$$

$$\forall i = 0, 1, \ldots N-1$$

$$\tilde{z}^T(k+N|k)S\tilde{z}(K+N|k) \leq 1 \tag{8.2.31e}$$

Optimization (8.2.31) is still nonconvex due to constraints (8.2.31c)-(8.2.31d). Indeed, $\forall i \geq 1$, the input constraints depend on the predicted state of the internal dynamics which is a nonlinear function of the control inputs. One possible way to convexify the optimization problem is to substitute the polyhedral constraint (8.2.31d) with its quadratic worst-case approximation (8.2.15) $\forall i \geq 1$, which is independent of the nonlinear dynamics state η. On the other hand, to mitigate the conservativeness of the MPC controller, for $i = 0$, since $\eta(k)$ can be measured, the actual polyhedral constraint can be used. Consequently, optimization (8.2.31) can be rewritten as:

$$\min_{w(k),\ldots,w(k+N-1)} J_N(k, \tilde{z}(k), \tilde{w}(k)) \tag{8.2.32a}$$

$$\tilde{z}(k+i+1|k) = A\tilde{z}(k+i|k) + Bw(i) - Bw_r(i) \tag{8.2.32b}$$

$$\forall i = 0, 1, \ldots N-1$$

$$L(\eta(k))w(k) \leq g \tag{8.2.32c}$$

$$w(k+i)^T w(k+i) \leq \hat{r}^2,\ i = 1, \ldots N-1 \tag{8.2.32d}$$

$$\tilde{z}^T(k+N|k)S\tilde{z}(K+N|k) \leq 1 \tag{8.2.32e}$$

which is a Quadratically Constrained Quadratic Programming (QCQP) problem.

Proposition 8.4. *The QCQP problem* (8.2.32) *can be rewritten in the following standard form:*

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \frac{1}{2}\mathbf{w}^T H \mathbf{w} + p^T \mathbf{w}\ s.t. \tag{8.2.33a}$$

$$\hat{L}(\eta(k))\mathbf{w} \leq g \tag{8.2.33b}$$

$$\mathbf{w}^T \hat{Q}_u \mathbf{w} \leq 1 \tag{8.2.33c}$$

$$\mathbf{w}^T \Theta_N^T S \Theta_N \mathbf{w} + 2\tilde{z}^T(k)\Psi_N^T S \Theta_N \mathbf{w} \leq 1 - \tilde{z}^T(k)\Psi_N^T S \Psi_N \tilde{z}(k) \tag{8.2.33d}$$

where

$$H = \Theta^T \hat{Q} \Theta + \hat{R}$$

and

$$p = \Theta^T \hat{Q} \Psi \tilde{z}(k) - \Theta^T \hat{Q} \Theta \mathbf{w_r} - \hat{R}\mathbf{w_r}$$

with

$$\Psi = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix}, \quad \Theta = \begin{bmatrix} B & 0 & \ldots & 0 \\ AB & B & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A^{N-1}B & A^{N-2}B & \ldots & B \end{bmatrix}$$

$$\Psi_N = A^N, \quad \Theta_N = \left[A^{N-1}B,\, A^{N-2}B, \ldots, AB,\, B\right]$$

$$\hat{Q} = \begin{bmatrix} Q & 0 & \ldots & 0 \\ 0 & Q & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & Q \end{bmatrix}, \quad \hat{R} = \begin{bmatrix} R & 0 & \ldots & 0 \\ 0 & R & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & R \end{bmatrix}$$

$$\hat{Q}_u = \begin{bmatrix} 0 & 0 & \ldots & 0 \\ 0 & \frac{1}{\hat{r}^2} I_{2\times 2} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \frac{1}{\hat{r}^2} I_{2\times 2} \end{bmatrix}$$

$$\hat{L}(\eta(k)) = [L(\eta(k)),\, 0, \ldots 0]$$

Proof. Let's define the decision variables vector $\mathbf{w} = [w(k),\, w(k+1), \ldots w(k+N-1)]^T$, and the predicted reference input vector $\mathbf{w_r} = [w_r(k),\, w_r(k+1), \ldots w_r(k+N-1]^T$. Then, using (8.2.32b), the model predictions $\tilde{\mathbf{z}} = [\tilde{z}(k+1|k), \ldots, \tilde{z}(k+N|k)]^T$ can be rewritten in a compact form as $\tilde{\mathbf{z}} = \Psi\tilde{z}(k) + \Theta\mathbf{w}$. As a consequence the cost function $J_N(k, \tilde{z}(k), \tilde{w}(k)) = \frac{1}{2}\sum_{i=0}^{N-1} \tilde{z}(k+i+1|k)^T Q \tilde{z}(k+i+1|k) + \tilde{w}(k+i|k)^T R \tilde{w}(k+i|k)$ can be rewritten as $J(\mathbf{w}) = \frac{1}{2}[(\Phi\tilde{z}(k) + \Theta(\mathbf{w}-\mathbf{w_r}))^T \, \hat{Q} \, (\Phi\tilde{z}(k) + \Theta(\mathbf{w}-\mathbf{w_r})) + (\mathbf{w}-\mathbf{w_r})^T \hat{R}(\mathbf{w}-\mathbf{w_r})] = \frac{1}{2}\mathbf{w}^T H \mathbf{w} + p^T\mathbf{w} + c$. Notice that in optimization (8.2.33) the term c has been dropped since it does not affect the optimal solution of the optimization. By applying the same arguments, it is easy to show that $L(\eta(k+i|k))w(k+i) \le g,\ i = 0, 1, \ldots N-1 \iff \hat{L}(\eta(k))\mathbf{w} \le g$, $w(k+i)^T w(k+i) \le \hat{r}^2,\ i = 1, \ldots N-1 \iff \mathbf{w}^T \hat{Q}_u \mathbf{w} \le 1$, and $\tilde{z}^T(k+N|k) S \tilde{z}(k+N|k) \le 1 \iff \mathbf{w}^T \Theta_N^T S \Theta_N \mathbf{w} + 2\tilde{z}(k)\Psi_N^T S \Theta_N \mathbf{w} \le 1 - \tilde{z}^T(k)\Psi_N^T S \Psi_N \tilde{z}(k)$ □

Remark 8.5. The QCQP problem can be recast into a computationally more affordable QP problem. Specifically, the quadratic constraints (8.2.33c) and (8.2.33d) can be replaced with a polyhedral inner approximation. In particular, by defining two polyhedral sets $\mathcal{P}_w = \{w \in \mathbb{R}^2 : P_w w \le p_w\} \subset \hat{\mathcal{U}}$, $P_w \in \mathbb{R}^{n_w \times 2}$, $p_w \in \mathbb{R}^{n_w}$ and $\mathcal{P}_N = \{\tilde{z} \in \mathbb{R}^2 : P_{\tilde{z}_N}\tilde{z} \le p_{\tilde{z}_N}\} \subset \Sigma_N$, $P_{\tilde{z}_N} \in \mathbb{R}^{n_N \times 2}$, $p_{\tilde{z}_N} \in \mathbb{R}^{n_{\tilde{z}_N}}$, where n_w and $n_{\tilde{z}_N}$ are the number of sides of the polyhedral approximations $\mathcal{P}_w$ and $\mathcal{P}_{\tilde{z}_N}$, respectively. Then, constraint (8.2.33c) can be replaced with

$$\hat{P}_w \mathbf{w} \leq \hat{p}_w \tag{8.2.34}$$

where

$$\hat{P}_w = \begin{bmatrix} 0 & 0 & \dots & 0 \\ 0 & P_w & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & P_w \end{bmatrix}, \quad \hat{p}_u = \begin{bmatrix} 0 \\ p_w \\ \vdots \\ p_w \end{bmatrix}$$

Similarly, the quadratic constraint (8.2.32e) can be replaced with its polyhedral approximation $P_{\tilde{z}_N} \tilde{z}(k+N) \leq p_{\tilde{z}_N}$ which can be rewritten as a function of the decision variable $\mathbf{w}$, i.e.,

$$P_{\tilde{z}_N} \Theta_N \mathbf{w} \leq p_{\tilde{z}_N} - P_{\tilde{z}_N} \Psi_N \tilde{z}(k) \tag{8.2.35}$$

Therefore, replacing (8.2.33c)-(8.2.33d) with (8.2.34)-(8.2.35), a QP optimization is obtained. Notice that the conservativeness of the derived QP problem depends on the number of sides of the polyhedra approximations, i.e. n_w and n_N, which are additional design parameters.

All the above developments can be collected into the computable Algorithm 6, which, as proved in the following theorem, provides a solution to Problem 6.

Algorithm 6 Dual-Mode Feedback-Linearized Tracking MPC (Dual-Mode FL-MPC) algorithm

```
Offline:
1: Find $K$ solving (8.2.29) and set $S$ as in (8.2.26)
2: Set the prediction horizon $N$ such that (8.2.33) is feasible with the initial condition
   $\tilde{z}(0) = z(0) - z_r(0)$
Online:
1: Estimate $x(k)$, $y(k)$, $\theta(k)$, $\varphi(k)$ and compute $\tilde{z}(k) = z(k) - z_r(k)$, and $\eta(k) =$
   $[\theta(k), \varphi(k)]^T$.
2: Compute $L(\eta(k))$ as in (8.2.14) and $w_r(k)$ as in (8.2.10);
3: if $\tilde{z}(k) \notin \Sigma_N$ then
4:     Find $\mathbf{w}^*$ solving (8.2.33) and set $w(k) = w^*(k)$
5: else
       $w(k) = K\tilde{z}(k) + \hat{w}_r(k)$        (8.2.36)
   where $\hat{w}_r(k)$ is the optimal solution of (8.2.28)
6: end if
7: Compute
       $[v(k), \omega(k)]^T = M^{-1}(\eta(k))w(k)$        (8.2.37)
   and apply it to the car; $k \leftarrow k+1$, go to 1;
```

Theorem 8.5. *For any $\tilde{z}(0)$ such that (8.2.33) is feasible, the tracking FL-MPC strategy described in Algorithm 6 provides a solution to Problem 6.*

Proof. The proof can be divided into two parts:

(I) Stability and input constraint fulfillment of the linearized tracking error dynamics: First, let's consider the input-output linearized model (8.2.5). If at the generic time k, (8.2.33) admits a solution for a given initial condition $\tilde{z}(k)$ and for some $N > 0$, then the optimal control sequence $\{w^*(k), w^*(k+1), \dots, w^*(k+N-1)\}$ with $w^*(k+i) \in \mathcal{U}(\eta(k)), \forall \eta(k), \forall i = 0, 1, \dots N-1$, is such that $\tilde{z}(k+N) \in \Sigma_N$. At time $k+1$, a feasible solution to optimization (8.2.33) can be constructed from the optimal solution at time k, i.e., $\{w^*(k+1), w^*(k+2) \dots, w^*(k+N-1), K\tilde{z}(k+N)\}$.

Indeed, the last control move $K\tilde{z}(k+N)$ is, by construction, always constraint-admissible inside the RPI region Σ_N. As a consequence, the optimization (8.2.33) is recursively feasible ensuring that, in at most N steps, $\tilde{z}(k)$ is steered into Σ_N. Then, given the RPI nature of Σ_N, we can also conclude that $\tilde{z}(k)$ is UUB in Σ_N. Furthermore, since the used input-output linearization and discretization commutes (see property (8.1)), the linearized tracking error dynamics (8.2.11) is stable under the effect of the proposed dual-mode MPC.

(II) Bounded Tracking Error for (2.1.10): As proven in part (I), $w(k)$ computed by Algorithm 6 stabilizes the feedback linearized error dynamics (8.2.12a). Therefore, given the result of Lemma 8.1 and by applying the input transformation (8.2.3), the control law (8.2.37) solves the considered reference tracking problem with a bounded tracking error $\tilde{q}(k)$. □

8.3 Experimental Results

Here, the experimental results, obtained using a Quanser Qcar[5], are presented to show the effectiveness of the proposed FL-MPC tracking controller and compare it with alternative solutions. In particular, the chosen competitors are the nonlinear MPC tracking strategy described in Chapter 8.2.1 (namely "Nonlinear MPC"), and the the constrained adaptive backstepping controller developed in [61] (namely "Backstepping"). Two different versions of the proposed FL-MPC scheme have been tested. The first one exactly follows Algorithm 6 (hereafter referred to as "Dual FL-MPC"). On the other hand, the second one (namely "FL-MPC") executes Algorithm 6 but it never activates the terminal control law, i.e., it solves (8.2.33) for any $k \geq 0$.

8.3.1 Experimental Setup

The considered experimental setup is depicted in Fig. 8.3, and it consists of:

a) a Quanser Qcar;

b) a camera-based (IPS);

c) a ground workstation;

d) a Wifi communication channel between the ground workstation and the car.

[5]`https://www.quanser.com/products/qcar/`

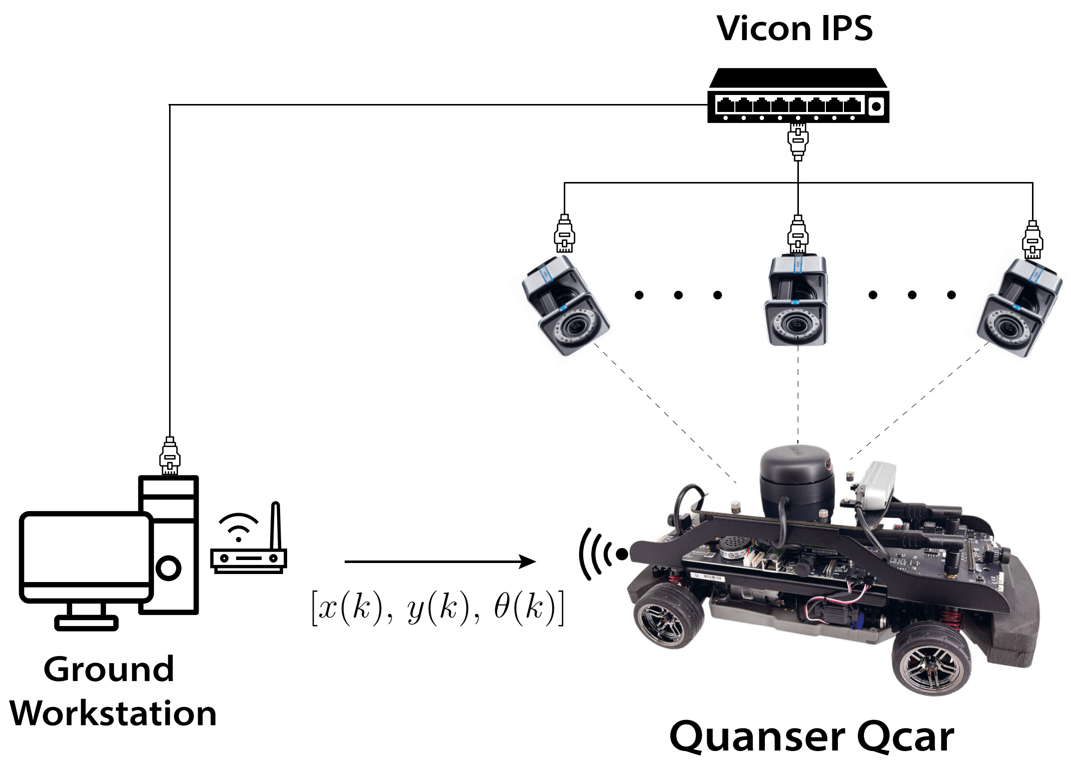


Figure 8.3: Proposed experimental setup

The autonomous car-like vehicle is the Quanser Qcar open-architecture prototype, which is designed for academic research experiments. The car has a size of $0.39 \times 0.19 \times 0.20\,m$, weights $2.7\,kg$, and its wheelbase measures $l = 0.256m$. Onboard, the car is equipped with different sensors (encoders gyroscope, accelerometer, magnetometer, lidar, and depth and RGB cameras) an *NVIDIA® Jetson™ TX2 with 2 GHz quad-core ARM Cortex-A57 64-bit + 2 GHz Dual-Core NVIDIA Denver2 64-bit CPU and 8GB memory.* The considered maximum longitudinal speed is $\overline{v} = 1\,m/s$, while the maximum steering angular velocity is $\overline{\omega} = 10\,rad/s$. In addition, due to the vehicle's mechanical construction, the front wheel's steering angle cannot exceed $\overline{\varphi} = 0.6\,rad$. The developed tracking algorithms have been implemented and cross-compiled in C language and run onboard on the Nvidia Jetson CPU. A sampling time $T_s = 0.01\,s$ has been used for all the performed tests.

The IPS consists of a set of 12 Vicon Vero cameras connected via a wired connection to the ground workstation. The camera system is used to localize the car in the workspace, similar to a standard GPS system. In particular, the cameras detect and track a set of reflective markers placed on the Qcar. The positions of the markers are in real-time collected and processed on the ground workstation by the Vicon Tracker software, which accurately reconstructs, via a triangulation algorithm, the position and orientation of car.

The ground workstation is a desktop computer consisting of a *13th Gen Intel(R) Core(TM)* *i9-13900KF* CPU, a *NVIDIA GeForce RTX 4070* GPU, and 64GB of RAM. The workstation is used to estimate the pose of the car and broadcast it via a TCP/IP communication channel.

8.3.2 Configuration of the proposed controller

To implement the proposed tracking controller strategy the following parameters have been considered: $\Delta = 0.35$, $Q = I_{2\times2}$, $R = 0.01 \cdot I_{2\times2}$, $K = 4I_{2\times2}$. The radius of the worst-case circular input constraint set has been computed as in (8.2.15), obtaining $\hat{r} = 1$. The reference trajectory is built to comply with (8.2.13), with $r_d = 11.54$. As a consequence, the feedback control gain $K = 4I_{2\times2}$ has been chosen such that RPI condition (8.2.29) is satisfied, with $A_{cl} = 0.96I_{2\times2}$, $S = 16I_{2\times2}$, $G = 0.25I_{2\times2}$, $\xi = 4.69 \cdot 10^{-8}$, $\lambda = 0.9998$, computed as outlined in in Proposition 8.3. The idea described in Remark 8.5 has been used to obtain a QP formulation of the derived QCQP optimization (8.2.33). In particular, the quadratic constraints (8.2.33c)-(8.2.33d) have been inner approximated using two Decahedra, i.e., two polyhedral sets defined as in (8.2.34)-(8.2.35), with $n_w = n_N = 10$. The control optimization problem has been solved on the car's processing unit considering a prediction horizons $N = 10$ using an *Active Set* solver algorithm. The *Active Set* algorithm makes use of a *Cholesky* decomposition of the hessian matrix H, which, being constant for the proposed optimization, has been precomputed to reduce the online computational load. The computational times obtained for the used solver are reported in Chapter 8.3.6.

8.3.3 Car's state estimation

The state of the car $q(k)$ is onboard estimated by means of an Unscented Kalman Filter (UKF). In particular, the implemented UKF is outlined in [115] and it exploits the nonlinear kinematic model (8.2.7) and different collected sensor information: (i) the estimated position and orientation of the Car provided by the Vicon Camera System, (ii) encoder and (iii) IMU (gyroscope and accelerometer) measurements. The UKF has been configured with the following parameters: process and measurement covariance matrices

$$Q_{UKF} = diag([10^{-3}, 10^{-3}, 10^{-1}, 10]), \quad R_{UKF} = diag([2 \cdot 10^{-5}, 2 \cdot 10^{-5}, 10^{-4}, 10^{-5}]),$$

initial state estimation covariance matrix

$$P^0_{UKF} = diag([10^{-5}, 10^{-5}, 10^{-6}, 10^{-6}]),$$

and sigma-points parameters $\alpha = 0.9$, $\beta = 2$, $\kappa = 0$ (the interested reader shall refer to [115] for further details about the used parameters).

8.3.4 Reference trajectory generation

A reference trajectory complying with assumption 8.1 has been generated using a cubic spline interpolation method. The interpolation algorithm received in input a sequence of waypoints describing the desired path and, in output, it assigns a crossing time based on path curvature and desired average speed. Then, each waypoint is interpolated using cubic splines, obtaining the position x_r, y_r, velocity $\dot{x}_r$, $\dot{y}_r$, acceleration $\ddot{x}_r$, $\ddot{y}_r$, and jerk $\dddot{x}_r$, $\dddot{y}_r$, needed to compute the reference car's state q_r, and inputs u_r, as in (8.1.1)-(8.1.2).

8.3.5 Configuration of the competitor schemes

Each competitor scheme has been configured to obtain the best tracking performance in the performed experiments. Specifically, the nonlinear MPC optimization (8.2.1) has been solved considering the following LQ cost matrices, $Q = diag([135,\ 135,\ 65,\ 65])$, $R = diag([0.3,\ 0.1])$, and a prediction horizon $N = 5$. The nonlinear optimization has been solved using *Sequential Quadratic Programming (SQP)* method. On the other hand, the backstepping algorithm developed in [61] has been tuned using: $d_x = 0.3$, $\sigma = 0.3$, $\alpha_c = 0.3$, $k_3 = 3.5$. The interested reader shall refer to [61] for a detailed explanation of the used parameters. It is worth mentioning that the steering command generated by the algorithm is subject to undesired chattering effects, typical of backstepping control algorithms [116]. In order to mitigate such an undesired effect, the control signal has been prefiltered using a low-pass filter with cut-off frequency $\omega_c = 100\frac{rad}{s}$.

8.3.6 Evaluation of the tracking performance

To evaluate the tracking performance of the proposed controller and alternative schemes, the Integral Square Error (ISE) ($\int_0^{T_f} e(t)^2 dt$) and Integral Time Squared Error (ITSE) ($\int_0^{T_f} te(t)^2 dt$) indexes have been used. In particular, for each performed experiment, three different error signals are measured: path distance error $e_{xy}(k) = \| [x(k),\ y(k)] - [x_r(k),\ y_r(k)] \|_2$, heading angle error $e_\theta(k) = \theta(k) - \theta_r(k))$, and steering angle error $e_\varphi(k) = \varphi(k) - \varphi_r(k))$. Then, for each collected error signal the ISE and ITSE indexes are computed, i.e., $ISE_{x,y}$, $ITSE_{x,y}$, ISE_θ, $ITSE_\theta$, ISE_φ, $ITSE_\varphi$.

8.3.7 Evaluation of computational times

To assess the computation complexity of the approaches Dual FL-MC and FL-MPC" and Nonlinear MPC, the computational times required by each algorithm have been measured. In the performed analysis, only the controller computation has been considered, i.e., the state-estimation algorithm as well as the sensor processing have been neglected.

8.3.8 Results

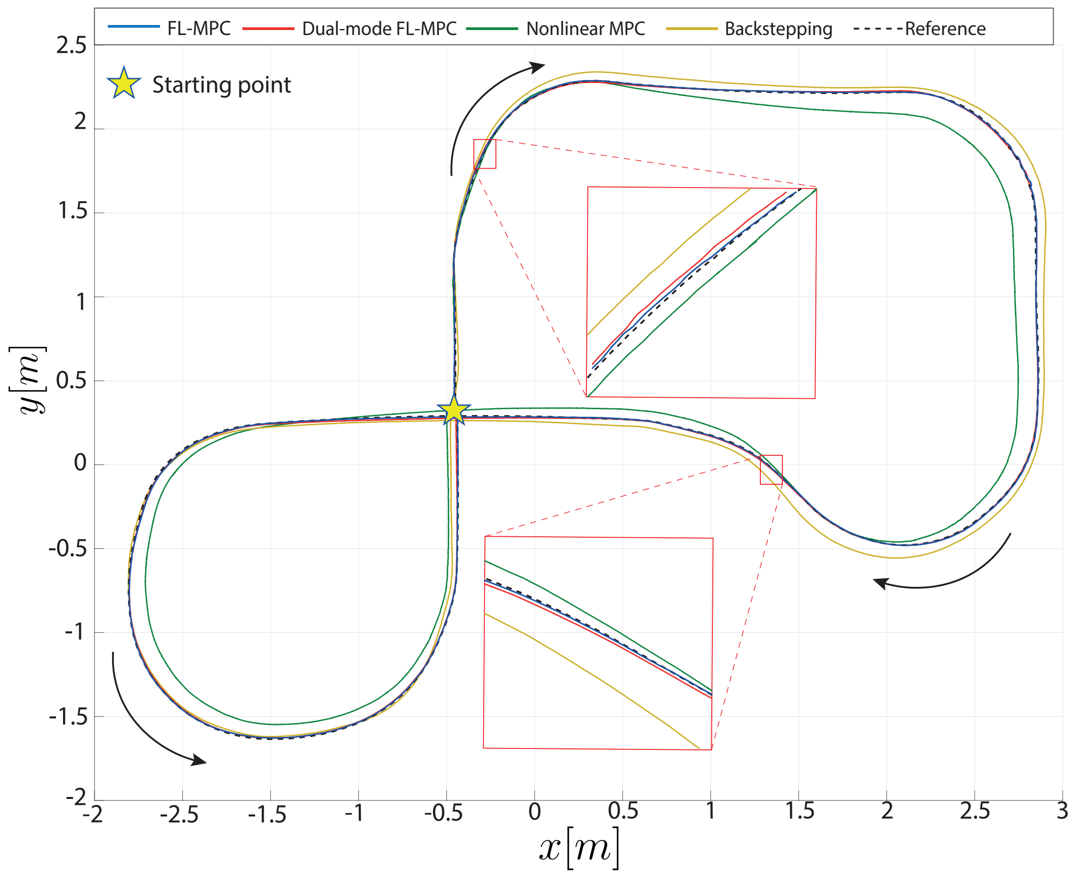


Figure 8.4: Experimental results: Trajectory

The obtained results are collected in Figs. 8.4-8.6 and and Tables 8.1-8.3. For the interested reader, videos of the performed experiments can be found at the following web link: `https://youtu.be/aeHZKyRfcEo`. The tracking performance has been evaluated considering two reference trajectories generated along the same path. The first requires a maximum speed of $0.6\frac{m}{s}$, while the second, a maximum speed of $0.75\frac{m}{s}$. It is worth mentioning that, for both the considered trajectories, several tests have been run, and the obtained results have been averaged in Table 8.1-8.2, respectively.

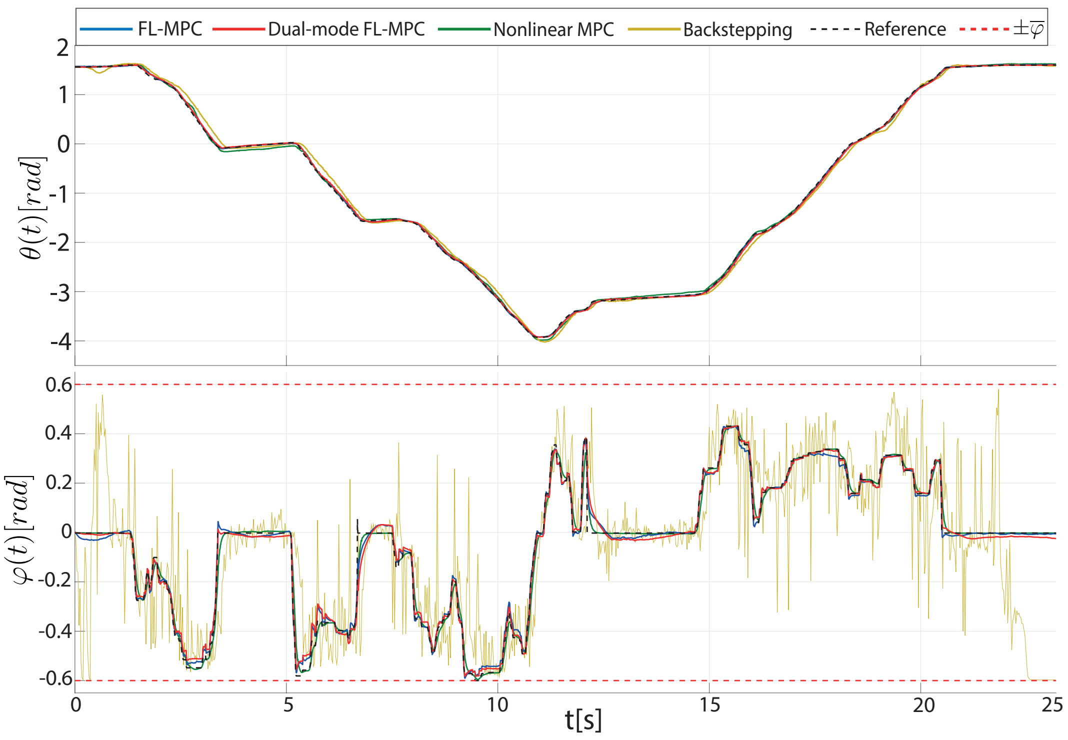


Figure 8.5: Experimental results: Heading and Steering Angles

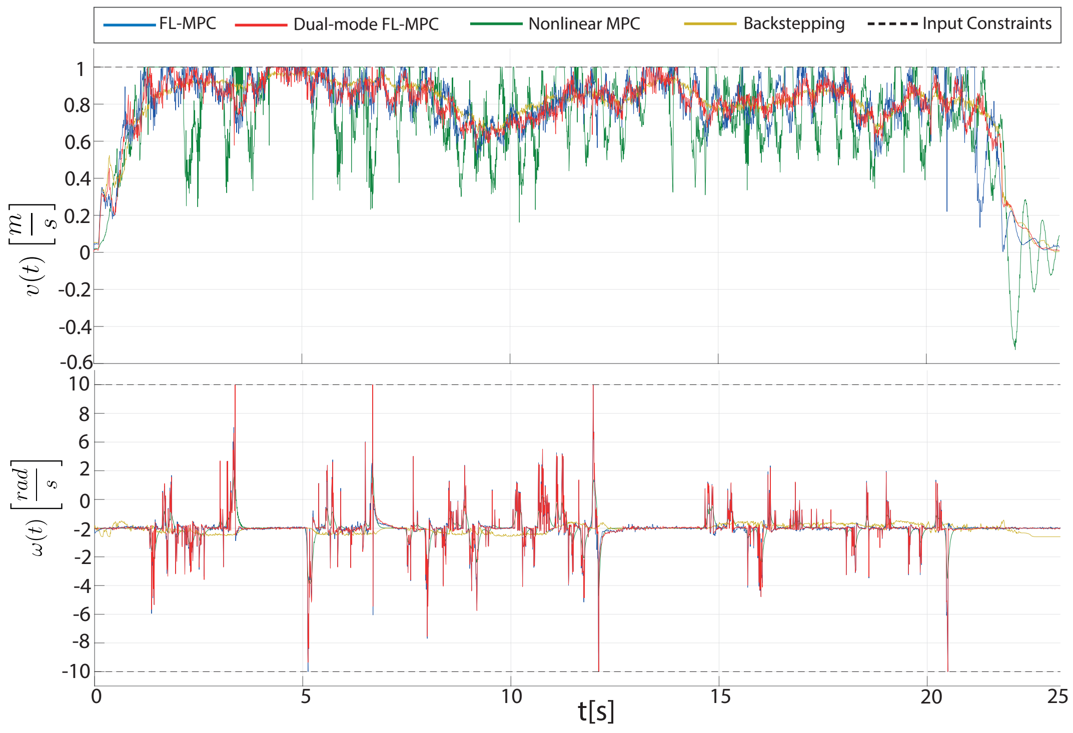


Figure 8.6: Experimental results: Control Inputs

Fig. 8.4 and Tables 8.1-8.2 show that for both trajectories, the proposed controller achieves better tracking when compared to the Nonlinear MPC and Backstepping controllers. In particular, the Nonlinear MPC showed poor tracking performance in all the performed experiments. This finds justification in the highly nonconvex nature of optimization (8.2.1), which converges to local minima and, consequently, to nonoptimal solutions. Moreover, as shown in Table 8.3, the reference tracking performance cannot be improved by increasing the prediction horizon. For example, for $N = 10$, the nonlinear optimization solver for (8.2.1) can take up to approximately $40\,ms$ to obtain a solution, which is far above the considered sampling time $T_s = 10\,ms$. On the other hand, the backstepping controller shows slightly better tracking performance of the Nonlinear MPC. However, a chattering phenomena affects the computed steering angle command (see the rapid and discontinuous switching of the control signal). Moreover, as shown in Fig. 8.6, the control inputs computed by the backstepping controller are conservative, i.e. the longitudinal and angular steering velocities never reach the prescribed limits, $\overline{v}$ and $\overline{\omega}$, respectively. The two above mentioned drawbacks justify why the tracking performance of the backstepping scheme are slightly worse of the one achieved with the proposed tracking controller degrades.

Table 8.1: Comparison of tracking performance: Trajectory 1

Algorithm	Distance Error		Heading Error		Steering Error	
	ISE_{xy}	$ITSE_{xy}$	ISE_{θ}	$ITSE_{\theta}$	ISE_{φ}	$ITSE_{\varphi}$
FL-MPC	0.0279	0.3191	0.0201	0.1797	0.0244	0.2834
Dual-mode FL-MPC	0.0323	0.4160	0.0246	0.3195	0.0333	0.3117
Nonlinear MPC	0.2703	4.4197	0.0978	1.4305	0.0135	0.1642
Backstepping [61]	0.1629	2.0904	0.1375	1.6539	1.8797	30.5802

Table 8.2: Comparison of tracking performance: Trajectory 2

Algorithm	Distance Error		Heading Error		Steering Error	
	ISE_{xy}	$ITSE_{xy}$	ISE_{θ}	$ITSE_{\theta}$	ISE_{φ}	$ITSE_{\varphi}$
FL-MPC	0.0321	0.4718	0.0127	0.1713	0.0132	0.1480
Dual-mode FL-MPC	0.0459	0.6006	0.0250	0.3571	0.0212	0.2439
Nonlinear MPC	0.2458	3.0141	0.0468	0.5247	0.0218	0.2568
Backstepping [61]	0.2634	3.2056	0.2378	2.6428	1.2098	20.9822

On the other hand, the performance obtained with the FL-MPC is slightly superior to the ones obtained with the dual-mode MPC. The reason behind such a result is that FL-MPC uses the RPI region (8.2.26) to ensure recursive feasibility, but it never directly relies on the associated controller, which is by nature more conservative. However, the dual-mode MPC implementation shows some computational advantage related to the fact that when the tracking error enters the terminal regions, then a simpler optimization problem (8.2.28) is solved. Moreover, for both performed experiments, the proposed FL-MPC controller fulfills the vehicle's input constraints, as shown in Figs. 8.4-8.6.

Table 8.3: Comparison of average and maximum computational times (in ms) of Nonlinear and FL-MPC

Algorithm	N=3		N=5		N=10	
	MAX (ms)	AVG(ms)	MAX(ms)	AVG(ms)	MAX(ms)	AVG(ms)
FL-MPC	3.3106	0.5416	3.8249	0.6455	3.1471	0.6954
Dual-mode FL-MPC	1.7726	0.3622	3.3520	0.3417	2.0981	0.3638
Nonlinear MPC	7.2283	3.1933	15.6261	5.2227	40.5351	6.8099

Concerning the computational analysis, the proposed Dual-mode FL-MPC and FL-MPC solutions have been compared with the Nonlinear MPC considering the prediction horizons $N = 3, N = 5, N = 10$. The obtained results are collected in Table 8.3. It can be appreciated how the computational times, both maximum and average ones, of the proposed solutions are significantly lower than the ones of the Nonlinear MPC, especially for larger values of N.

8.4 Conclusions

In this chapter, a novel Feedback Linearized Model Predictive Control strategy for input-constrained self-driving cars has been presented. The proposed strategy combines two main ingredients: 1) an input-output FL technique and 2) a dual-mode MPC framework. The obtained tracking controller has the peculiar capability of efficiently dealing with state-dependent input constraints acting on the feedback linearized car's model while ensuring recursive feasibility, stability, and velocity constraints fulfillment. Extensive experimental results and comparisons have been carried out to highlight the features and advantages of the proposed tracking controller.

9 Conclusions

This thesis introduces a novel predictive paradigm for autonomous vehicles and wheeled mobile robots. The proposed control paradigm is based on the combination of Model Predictive Control (MPC) and Feedback Linearization (FL) arguments and aims at solving challenging problems such as trajectory tracking, path following, obstacle avoidance, and formation control for input-constrained wheeled autonomous vehicles.

The control framework offers a promising solution to improve trajectory accuracy while considering nonholonomic constraints, input-saturation constraints, and bounded disturbances. The research demonstrates the benefits of exploiting feedback linearization for MPC predictions, characterizing the derived state-dependent input constraint set, and its analytical worst-case circular approximation. The proposed control solutions are designed to be computationally affordable, recursively feasible, and non-conservative, ensuring real-time implementation. Both simulations and laboratory experiments with the Khepera IV differential-drive robot and Quanser Qcar validate the effectiveness and advantages of the proposed approach compared to existing competitor schemes. This research paves the way for more efficient and reliable control strategies for autonomous wheeled vehicles, enabling their successful operation in real-world applications.

The thesis is structured to build a comprehensive understanding of the proposed control strategies and their applications. Chapter 2 provides the necessary theoretical foundation, discussing the kinematic modeling of wheeled autonomous vehicles, the principles of feedback linearization, and the fundamentals of predictive control. This background material sets the stage for the subsequent chapters, where the novel contributions are presented.

Chapter 3 represents the first attempt to analytically characterize the time-varying input constraint set acting on input-output linearized wheeled mobile robots and its worst-case inner approximation. These characterizations were directly exploited to design an LMI-based receding-horizon controller capable of guaranteeing bounded tracking error, recursive feasibility, and input constraint fulfillment. The work was extended in Chapter 4 by introducing a set-theoretic receding horizon control strategy to enhance tracking performance and deal with time-varying constraints. The performance improvement is achieved by modeling the reference input associated with the trajectory as a bounded disturbance acting on the feedback linearized tracking error dynamics. The proposed set-theoretic control drives the tracking error within a small robust positive invariant region, ensuring that it is bounded in a neighborhood of the origin.

Additionally, Chapter 5 addressed formation control for mobile robot platoons, tailoring the set-theoretic control strategy derived in Chapter 4 to formally ensure collision avoidance and bounded tracking error for a platoon of differential-drive robots. This work demonstrates the flexibility of the proposed control framework in handling multiple autonomous vehicles in a coordinated manner.

Chapter 6 tackled waypoint tracking under obstacle-avoidance requirements for differential-drive robots using a receding horizon approach based on feedback linearization. This chapter highlights the capability of the proposed control paradigm to navigate complex static environments, avoiding obstacles while maintaining accurate tracking of the desired path.

Finally, the proposed predictive control paradigm was extended to deal with feedback-linearized car-like vehicles in Chapters 7 and 8, where the trajectory tracking problem for self-driving cars was considered. Similar to the differential-drive case, feedback linearization was used to derive an equivalent linear model describing the car's kinematics. Time-varying input constraints and their approximation were analytically characterized and exploited to design real-time affordable predictive controllers for self-driving cars. The tracking performance and low computational complexity were validated by means of laboratory experiments conducted on a Quanser Qcar. The obtained results were contrasted with several competitor control algorithms, including nonlinear MPC, showcasing the advantages of the proposed approach in terms of computational efficiency and robustness.

The extensive validation through simulations and real-time experiments underscores the practical applicability of the proposed control strategies. The research outcomes demonstrate significant improvements in tracking accuracy, obstacle avoidance, and formation control, making substantial contributions to the field of autonomous vehicle control.

Despite the promising results, several aspects require further investigation to enhance the proposed control framework:

- **State constraints:** State constraints have not been fully considered in the proposed control framework. Although geometrical and collision-avoidance constraints were addressed for differential-drive robot models, limits on the heading angle of the vehicle, steering angle of the car, or geometrical constraints on the car's position have not been directly tackled in this work. The integration of such constraints may not be trivial, as they might be nonconvex and time-varying for the feedback linearized system. The inherent problem is the difficulty in analytically characterizing inner approximations of such time-varying state constraints.

- **Model accuracy:** The proposed control framework makes use of simple kinematic models describing the robots and vehicles. Although it has been extensively validated considering small vehicles showing promising performance, further experiments are needed to validate such a framework when large-scale vehicles are considered. The proposed predictive control may suffer from model mismatches, especially at very high speeds, necessitating further refinement of the models to account for dynamic effects and higher-order nonlinearities.

- **Trajectory planning:** When dealing with the trajectory tracking problem, the proposed control framework presupposes the availability of reference trajectories compatible with the vehicle kinematic model, which enjoys properties such as continuity and smoothness. A kinodynamic planner may be needed to meet such an assumption, especially in unknown environments. The trajectory planning problem has not been directly tackled in this thesis. Integrating state-of-the-art algorithms based on neural networks to accomplish such a task and their incorporation within the proposed control framework may be an interesting subject for further studies.

- **Robustness to disturbances and uncertainties:** While the proposed control strategies have shown robustness to bounded disturbances, further work is needed to enhance robustness against a wider range of uncertainties, including sensor noise, actuator faults, and dynamic changes in the environment. Developing adaptive control mechanisms and incorporating robust optimization techniques could improve the resilience of the control framework.

- **Scalability to larger formations:** The formation control strategies proposed for platoons of mobile robots demonstrate the potential for cooperative control. However, scalability to larger fleets and more complex coordination tasks requires further exploration. Investigating decentralized and distributed control approaches could enable efficient management of larger groups of autonomous vehicles.

In conclusion, this thesis bridges the gap between nonlinear and linear MPC for autonomous wheeled vehicles by formulating recursively feasible predictive solutions based on feedback linearization, which are computationally affordable and do not introduce any approximation in the model prediction. These optimizations are convex by design and do not suffer from local minima problems typical of nonlinear MPC formulations. Their real-time effectiveness and performance have been extensively validated, showing promising results for their applicability in real-time scenarios. Further studies may be devoted to extending the predictive framework to deal with state constraints and integrating it with trajectory planning algorithms for unknown or partially known scenarios. This research lays a solid foundation for future advancements in the control of autonomous wheeled vehicles, contributing to their safe and efficient operation in diverse real-world environments.

References

[1] C. Badue, R. Guidolini, R. V. Carneiro, P. Azevedo, V. B. Cardoso, A. Forechi, L. Jesus, R. Berriel, T. M. Paixao, F. Mutz, *et al.*, "Self-driving cars: A survey," *Expert systems with applications*, vol. 165, p. 113816, 2021.

[2] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to autonomous mobile robots.* MIT press, 2011.

[3] L. S. Martins-Filho and E. E. Macau, "Trajectory planning for surveillance missions of mobile robots," in *Autonomous robots and agents*, pp. 109–117, Springer, 2007.

[4] M. S. Das, A. Samanta, S. Sanyal, and S. Mandal, "Support value-based nfsmc for wheeled mobile robot path tracking in unknown environments," *Wireless Personal Communications*, pp. 1–21, 2021.

[5] B. Ning, Q.-L. Han, and Q. Lu, "Fixed-time leader-following consensus for multiple wheeled mobile robots," *IEEE transactions on cybernetics*, vol. 50, no. 10, pp. 4381–4392, 2019.

[6] X. Gao, J. Li, L. Fan, Q. Zhou, K. Yin, J. Wang, C. Song, L. Huang, and Z. Wang, "Review of wheeled mobile robots' navigation problems and application prospects in agriculture," *IEEE Access*, vol. 6, pp. 49248–49268, 2018.

[7] S. Erfani, A. Jafari, and A. Hajiahmad, "Comparison of two data fusion methods for localization of wheeled mobile robot in farm conditions," *Artificial Intelligence in Agriculture*, vol. 1, pp. 48–55, 2019.

[8] A. Vasiliev and I. Dalyaev, "Simulation method for the transport system of a small-sized reconfigurable mobile robot," *Machines*, vol. 9, no. 1, p. 8, 2021.

[9] S. Perminov, N. Mikhailovskiy, A. Sedunin, I. Okunevich, I. Kalinov, M. Kurenkov, and D. Tsetserukou, "Ultrabot: Autonomous mobile robot for indoor uv-c disinfection," *arXiv preprint arXiv:2108.09772*, 2021.

[10] J. Guo, H. Gao, L. Ding, T. Guo, and Z. Deng, "Linear normal stress under a wheel in skid for wheeled mobile robots running on sandy terrain," *Journal of Terramechanics*, vol. 70, pp. 49–57, 2017.

[11] M. B. Alatise and G. P. Hancke, "A review on challenges of autonomous mobile robot and sensor fusion methods," *IEEE Access*, vol. 8, pp. 39830–39846, 2020.

[12] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Trans. on Intelligent Vehicles*, vol. 1, no. 1, pp. 33–55, 2016.

[13] S. Dixit, S. Fallah, U. Montanaro, M. Dianati, A. Stevens, F. Mccullough, and A. Mouzakitis, "Trajectory planning and tracking for autonomous overtaking: State-of-the-art and future prospects," *Annual Reviews in Control*, vol. 45, pp. 76–86, 2018.

[14] P. Stano, U. Montanaro, D. Tavernini, M. Tufo, G. Fiengo, L. Novella, and A. Sorniotti, "Model predictive path tracking control for automated road vehicles: A review," *Annual reviews in control*, vol. 55, pp. 194–236, 2023.

[15] Y. Q. Chen and Z. Wang, "Formation control: a review and a new consideration," in *2005 IEEE/RSJ International conference on intelligent robots and systems*, pp. 3181–3186, IEEE, 2005.

[16] C. Gonzalez and H. B. Schlegel, "An improved algorithm for reaction path following," *The Journal of Chemical Physics*, vol. 90, no. 4, pp. 2154–2161, 1989.

[17] A. De Luca and G. Oriolo, "Modelling and control of nonholonomic mechanical systems," in *Kinematics and dynamics of multi-body systems*, pp. 277–342, Springer, 1995.

[18] F. Kuhne, W. F. Lages, and J. G. da Silva Jr, "Model predictive control of a mobile robot using linearization," in *Mechatronics and robotics*, pp. 525–530, Citeseer, 2004.

[19] G. Oriolo, A. De Luca, and M. Vendittelli, "Wmr control via dynamic feedback linearization: design, implementation, and experimental validation," *IEEE Transactions on Control Systems Technology*, vol. 10, no. 6, pp. 835–852, 2002.

[20] Y. Kanayama, Y. Kimura, F. Miyazaki, and T. Noguchi, "A stable tracking control method for an autonomous mobile robot," in *IEEE Int. Conf. on Robotics and Automation*, pp. 384–389, 1990.

[21] Z.-P. JIANGdagger and H. Nijmeijer, "Tracking control of mobile robots: A case study in backstepping," *Automatica*, vol. 33, no. 7, pp. 1393–1399, 1997.

[22] S. Blažič, "A novel trajectory-tracking control law for wheeled mobile robots," *Robotics and Autonomous Systems*, vol. 59, no. 11, pp. 1001–1007, 2011.

[23] K. Shojaei, A. M. Shahri, A. Tarakameh, and B. Tabibian, "Adaptive trajectory tracking control of a differential drive wheeled mobile robot," *Robotica*, vol. 29, no. 3, pp. 391–402, 2011.

[24] B. d'Andréa Novel, G. Campion, and G. Bastin, "Control of nonholonomic wheeled mobile robots by state feedback linearization," *The Int. Journal of Robotics Research*, vol. 14, no. 6, pp. 543–559, 1995.

[25] G. Klančar and I. Škrjanc, "Tracking-error model-based predictive control for mobile robots in real time," *Robotics and Autonomous Systems*, vol. 55, no. 6, pp. 460–469, 2007.

[26] T. P. Nascimento, C. E. T. Dórea, and L. M. G. Gonçalves, "Nonlinear model predictive control for trajectory tracking of nonholonomic mobile robots: A modified approach," *International Journal of Advanced Robotic Systems*, vol. 15, no. 1, p. 1729881418760461, 2018.

[27] E. F. Camacho and C. B. Alba, *Model predictive control.* Springer science & business media, 2013.

[28] S. Gros, M. Zanon, R. Quirynen, A. Bemporad, and M. Diehl, "From linear to nonlinear mpc: bridging the gap via the real-time iteration," *Int. Journal of Control*, vol. 93, no. 1, pp. 62–80, 2020.

[29] S. J. Yoo, Y. H. Choi, and J. B. Park, "Generalized predictive control based on self-recurrent wavelet neural network for stable path tracking of mobile robots: adaptive learning rates approach," *IEEE Trans. on Circuits and Systems I: Regular Papers*, vol. 53, no. 6, pp. 1381–1394, 2006.

[30] T. P. Nascimento, C. E. Dórea, and L. M. G. Gonçalves, "Nonholonomic mobile robots' trajectory tracking model predictive control: a survey," *Robotica*, vol. 36, no. 5, pp. 676–696, 2018.

[31] D. Gu and H. Hu, "Receding horizon tracking control of wheeled mobile robots," *IEEE Trans. on Control Systems Technology*, vol. 14, no. 4, pp. 743–749, 2006.

[32] J. Wei and B. Zhu, "Linear time-varying model predictive control for trajectory-tracking of a wheeled mobile robot," in *Int. Conf. on Neural Computing for Advanced Applications*, pp. 533–544, Springer, 2021.

[33] L. Lapierre and R. Zapata, "A guaranteed obstacle avoidance guidance system," *Autonomous Robots*, vol. 32, no. 3, pp. 177–187, 2012.

[34] M. Hoy, A. S. Matveev, and A. V. Savkin, "Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey," *Robotica*, vol. 33, no. 3, pp. 463–497, 2015.

[35] S. M. LaValle, *Planning algorithms.* Cambridge university press, 2006.

[36] X. Xiao, B. Liu, G. Warnell, and P. Stone, "Motion control for mobile robot navigation using machine learning: a survey," *arXiv preprint arXiv:2011.13112*, 2020.

[37] S. Yu, M. Hirche, Y. Huang, H. Chen, and F. Allgöwer, "Model predictive control for autonomous ground vehicles: a review," *Autonomous Intelligent Systems*, vol. 1, no. 1, pp. 1–17, 2021.

[38] A. S. Lafmejani and S. Berman, "Nonlinear mpc for collision-free and deadlock-free navigation of multiple nonholonomic mobile robots," *Robotics and Autonomous Systems*, vol. 141, p. 103774, 2021.

[39] J. Liu, P. Jayakumar, J. L. Stein, and T. Ersal, "Combined speed and steering control in high-speed autonomous ground vehicles for obstacle avoidance using model predictive control," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 10, pp. 8746–8763, 2017.

[40] A. Tahirovic, G. Magnani, *et al.*, *Passivity-based model predictive control for mobile vehicle motion planning.* Springer, 2013.

[41] M. Seder, M. Baotić, and I. Petrović, "Receding horizon control for convergent navigation of a differential drive mobile robot," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 2, pp. 653–660, 2016.

[42] G. Franzè and W. Lucia, "An obstacle avoidance model predictive control scheme for mobile robots subject to nonholonomic constraints: A sum-of-squares approach," *Journal of the Franklin Institute*, vol. 352, no. 6, pp. 2358–2380, 2015.

[43] H. Guo, C. Shen, H. Zhang, H. Chen, and R. Jia, "Simultaneous trajectory planning and tracking using an mpc method for cyber-physical systems: A case study of obstacle avoidance for an intelligent vehicle," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 9, pp. 4273–4283, 2018.

[44] S. Khan, J. Guivant, and X. Li, "Design and experimental validation of a robust model predictive control for the optimal trajectory tracking of a small-scale autonomous bulldozer," *Robotics and Autonomous Systems*, vol. 147, p. 103903, 2022.

[45] G. Franze and W. Lucia, "A receding horizon control strategy for autonomous vehicles in dynamic environments," *IEEE Trans. on Control Systems Technology*, vol. 24, no. 2, pp. 695–702, 2015.

[46] A. Sciarretta, A. Vahidi, *et al.*, *Energy-efficient driving of road vehicles.* Springer, 2020.

[47] V. Lesch, M. Breitbach, M. Segata, C. Becker, S. Kounev, and C. Krupitzer, "An overview on approaches for coordination of platoons," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 10049–10065, 2022.

[48] M. Pham and D. Wang, "A unified nonlinear controller for a platoon of car-like vehicles," in *American Control Conference*, vol. 3, pp. 2350–2355, 2004.

[49] I. M. Delimpaltadakis, C. P. Bechlioulis, and K. J. Kyriakopoulos, "Decentralized platooning with obstacle avoidance for car-like vehicles with limited sensing," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 835–840, 2018.

[50] G. Klančar, D. Matko, and S. Blažič, "A control strategy for platoons of differential drive wheeled mobile robot," *Robotics and Autonomous Systems*, vol. 59, no. 2, pp. 57–64, 2011.

[51] W.-J. Liu, H.-F. Ding, M.-F. Ge, and X.-Y. Yao, "Cooperative control for platoon generation of vehicle-to-vehicle networks: a hierarchical nonlinear mpc algorithm," *Nonlinear Dynamics*, vol. 108, no. 4, pp. 3561–3578, 2022.

[52] F. Eiras, M. Hawasly, S. V. Albrecht, and S. Ramamoorthy, "A two-stage optimization-based motion planner for safe urban driving," *IEEE Transactions on Robotics*, vol. 38, no. 2, pp. 822–834, 2022.

[53] G. Franzè, G. Fedele, A. Bono, and L. D'Alfonso, "Reference tracking for multiagent systems using model predictive control," *IEEE Transactions on Control Systems Technology*, vol. 31, no. 4, pp. 1884–1891, 2023.

[54] D. Angeli, A. Casavola, G. Franzè, and E. Mosca, "An ellipsoidal off-line mpc scheme for uncertain polytopic discrete-time systems," *Automatica*, vol. 44, no. 12, pp. 3113–3119, 2008.

[55] M. Liu, F. Zhao, J. Yin, J. Niu, and Y. Liu, "Reinforcement-tracking: an effective trajectory tracking and navigation method for autonomous urban driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6991–7007, 2021.

[56] L. Li, J. Li, and S. Zhang, "State-of-the-art trajectory tracking of autonomous vehicles," *Mechanical Sciences*, vol. 12, no. 1, pp. 419–432, 2021.

[57] L. Nie, J. Guan, C. Lu, H. Zheng, and Z. Yin, "Longitudinal speed control of autonomous vehicle based on a self-adaptive pid of radial basis function neural network," *IET Intelligent Transport Systems*, vol. 12, no. 6, pp. 485–494, 2018.

[58] H. T. Abatari and A. D. Tafti, "Using a fuzzy pid controller for the path following of a car-like mobile robot," in *2013 First RSI/ISM international conference on robotics and mechatronics (ICRoM)*, pp. 189–193, IEEE, 2013.

[59] Y. Wu, L. Wang, J. Zhang, and F. Li, "Path following control of autonomous ground vehicle based on nonsingular terminal sliding mode and active disturbance rejection control," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 7, pp. 6379–6390, 2019.

[60] M. Shirzadeh, M. H. Shojaeefard, A. Amirkhani, and H. Behroozi, "Adaptive fuzzy nonlinear sliding-mode controller for a car-like robot," in *2019 5th conference on knowledge based engineering and innovation (kbei)*, pp. 686–691, IEEE, 2019.

[61] J. Hu, Y. Zhang, and S. Rakheja, "Adaptive trajectory tracking for car-like vehicles with input constraints," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 3, pp. 2801–2810, 2021.

[62] S. Kuutti, R. Bowden, Y. Jin, P. Barber, and S. Fallah, "A survey of deep learning applications to autonomous vehicle control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 2, pp. 712–733, 2020.

[63] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *Journal of field robotics*, vol. 37, no. 3, pp. 362–386, 2020.

[64] P. F. Lima, G. C. Pereira, J. Mårtensson, and B. Wahlberg, "Experimental validation of model predictive control stability for autonomous driving," *Control Engineering Practice*, vol. 81, pp. 244–255, 2018.

[65] E. F. Camacho and C. Bordons, "Nonlinear model predictive control," in *Model Predictive Control*, pp. 249–288, Springer, 2007.

[66] P. Falcone, F. Borrelli, J. Asgari, H. E. Tseng, and D. Hrovat, "Predictive active steering control for autonomous vehicle systems," *IEEE Transactions on control systems technology*, vol. 15, no. 3, pp. 566–580, 2007.

[67] F. Borrelli, P. Falcone, T. Keviczky, J. Asgari, and D. Hrovat, "Mpc-based approach to active steering for autonomous vehicle systems," *International journal of vehicle autonomous systems*, vol. 3, no. 2-4, pp. 265–291, 2005.

[68] H. Pang, M. Liu, C. Hu, and N. Liu, "Practical nonlinear model predictive controller design for trajectory tracking of unmanned vehicles," *Electronics*, vol. 11, no. 7, p. 1110, 2022.

[69] M. Cho, Y. Lee, and K.-S. Kim, "Model predictive control of autonomous vehicles with integrated barriers using occupancy grid maps," *IEEE Robotics and Automation Letters*, vol. 8, no. 4, pp. 2006–2013, 2023.

[70] W. Schwarting, J. Alonso-Mora, L. Paull, S. Karaman, and D. Rus, "Safe nonlinear trajectory generation for parallel autonomy with a dynamic vehicle model," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 9, pp. 2994–3008, 2017.

[71] M. Nezami, D. S. Karachalios, G. Schildbach, and H. S. Abbas, "On the design of nonlinear mpc and lpvmpc for obstacle avoidance in autonomous driving," in *2023 9th International Conference on Control, Decision and Information Technologies (CoDIT)*, pp. 1–6, IEEE, 2023.

[72] F. Eiras, M. Hawasly, S. V. Albrecht, and S. Ramamoorthy, "A two-stage optimization-based motion planner for safe urban driving," *IEEE Trans. on Robotics*, vol. 38, no. 2, pp. 822–834, 2021.

[73] J. Funke, M. Brown, S. M. Erlien, and J. C. Gerdes, "Collision avoidance and stabilization for autonomous vehicles in emergency scenarios," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 4, pp. 1204–1216, 2016.

[74] C. E. Beal and J. C. Gerdes, "Model predictive control for vehicle stabilization at the limits of handling," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 4, pp. 1258–1269, 2012.

[75] K. Majd, M. Razeghi-Jahromi, and A. Homaifar, "A stable analytical solution method for car-like robot trajectory tracking and optimization," *IEEE/CAA Journal of Automatica Sinica*, vol. 7, no. 1, pp. 39–47, 2019.

[76] H. K. Khalil, "Nonlinear systems third edition," *Patience Hall*, vol. 115, 2002.

[77] M. A. Kamel and Y. Zhang, "Linear model predictive control via feedback linearization for formation control of multiple wheeled mobile robots," in *IEEE Int. Conf. on Information and Automation*, pp. 1283–1288, 2015.

[78] S. Bouzoualegh, E.-H. Guechi, and R. Kelaiaia, "Model predictive control of a differential-drive mobile robot," *Electr Mech Eng*, vol. 10, no. 2018, pp. 20–41, 2018.

[79] M. Yue, C. An, and J.-Z. Sun, "An efficient model predictive control for trajectory tracking of wheeled inverted pendulum vehicles with various physical constraints," *Int. Journal of Control, Automation and Systems*, vol. 16, no. 1, pp. 265–274, 2018.

[80] V. T. Nguyen, C. Sentouh, P. Pudlo, and J.-C. Popieul, "Path following controller for electric power wheelchair using model predictive control and transverse feedback linearization," in *IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC)*, pp. 4319–4325, 2018.

[81] J. Deng, V. Becerra, and R. Stobart, "Input constraints handling in an mpc/feedback linearization scheme," *Int. Journal of Applied Mathematics and Computer Science*, vol. 19, no. 2, pp. 219–232, 2009.

[82] D. Simon, J. Löfberg, and T. Glad, "Nonlinear model predictive control using feedback linearization and local inner convex constraint approximations," in *European Control Conference (ECC)*, pp. 2056–2061, IEEE, 2013.

[83] K. Margellos and J. Lygeros, "A simulation based mpc technique for feedback linearizable systems with input constraints," in *IEEE Conf. on Decision and Control (CDC)*, pp. 7539–7544, 2010.

[84] C. Tiriolo, G. Franzè, and W. Lucia, "A receding horizon trajectory tracking strategy for input-constrained differential-drive robots via feedback linearization," *IEEE Transactions on Control Systems Technology*, vol. 31, no. 3, pp. 1460–1467, 2023.

[85] C. Tiriolo and W. Lucia, "A set-theoretic control approach to the trajectory tracking problem for input-output linearized wheeled mobile robots," *IEEE Control Systems Letters*, vol. 7, pp. 2347–2352, 2023.

[86] S. Rajkumar, C. Tiriolo, and W. Lucia, "Collision-free platooning of mobile robots through a set-theoretic predictive control approach," *arXiv preprint arXiv:2403.08942*, 2024.

[87] C. Tiriolo, G. Franzè, and W. Lucia, "An obstacle-avoidance receding horizon control scheme for constrained differential-drive robot via dynamic feedback linearization," in *2023 American Control Conference (ACC)*, pp. 1116–1121, IEEE, 2023.

[88] C. Tiriolo and W. Lucia, "On the design of control invariant regions for feedback linearized car-like vehicles," *IEEE Control Systems Letters*, vol. 7, pp. 739–744, 2022.

[89] A. De Luca, G. Oriolo, and M. Vendittelli, "Control of wheeled mobile robots: An experimental overview," *Ramsete*, pp. 181–226, 2001.

[90] R. W. Brockett *et al.*, "Asymptotic stability and feedback stabilization," *Differential geometric control theory*, vol. 27, no. 1, pp. 181–191, 1983.

[91] A. D. Luca, G. Oriolo, and C. Samson, "Feedback control of a nonholonomic car-like robot," *Robot motion planning and control*, pp. 171–253, 1998.

[92] J.-J. E. Slotine, W. Li, *et al.*, *Applied nonlinear control*, vol. 199. Prentice hall Englewood Cliffs, NJ, 1991.

[93] D. Wang and G. Xu, "Full-state tracking and internal dynamics of nonholonomic wheeled mobile robots," *IEEE/ASME Trans. on Mechatronics*, vol. 8, no. 2, pp. 203–214, 2003.

[94] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[95] F. Allgöwer and A. Zheng, *Nonlinear model predictive control*, vol. 26. Birkhäuser, 2012.

[96] F. Borrelli, A. Bemporad, and M. Morari, *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.

[97] S. V. Rakovic, E. C. Kerrigan, K. I. Kouramas, and D. Q. Mayne, "Invariant approximations of the minimal robust positively invariant set," *IEEE Transactions on automatic control*, vol. 50, no. 3, pp. 406–410, 2005.

[98] A. Kurzhanski and I. Vályi, *Ellipsoidal calculus for estimation and control*. Springer, 1997.

[99] M. V. Kothare, V. Balakrishnan, and M. Morari, "Robust constrained model predictive control using linear matrix inequalities," *Automatica*, vol. 32, no. 10, pp. 1361–1379, 1996.

[100] B. Pluymers, M. Kothare, J. Suykens, and B. De Moor, "Robust synthesis of constrained linear state feedback using lmis and polyhedral invariant sets," in *American Control Conference (ACC)*, IEEE, 2006.

[101] F. Blanchini and S. Miani, "Any domain of attraction for a linear constrained system is a tracking domain of attraction," *SIAM Journal on Control and Optimization*, vol. 38, no. 3, pp. 971–994, 2000.

[102] C. Tiriolo, G. Franzè, and W. Lucia, "A receding horizon control strategy for constrained differential-drive robots moving in static unknown environments," in *IEEE Conference on Control Technology and Applications (CCTA)*, pp. 261–266, 2020.

[103] C. Tiriolo, G. Franzè, and W. Lucia, "A receding horizon trajectory tracking strategy for input-constrained differential-drive robots via feedback linearization," *IEEE Transactions on Control Systems Technology*, vol. 31, no. 3, pp. 1460–1467, 2023.

[104] E. A. Wan and R. Van Der Merwe, "The unscented kalman filter," *Kalman filtering and neural networks*, pp. 221–280, 2001.

[105] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[106] F. Duchoň, A. Babinec, M. Kajan, P. Beňo, M. Florek, T. Fico, and L. Jurišica, "Path planning with modified a star algorithm for a mobile robot," *Procedia Engineering*, vol. 96, pp. 59–69, 2014.

[107] S. M. LaValle, J. J. Kuffner, B. Donald, *et al.*, "Rapidly-exploring random trees: Progress and prospects," *Algorithmic and computational robotics: new directions*, vol. 5, pp. 293–308, 2001.

[108] W. Lucia, G. Franzè, and M. Sznaier, "A hybrid command governor scheme for rotary wings unmanned aerial vehicles," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 2, pp. 361–375, 2018.

[109] W.-J. Mao, "Robust stabilization of uncertain time-varying discrete systems and comments on "an improved approach for constrained robust model predictive control"," *Automatica*, vol. 39, no. 6, pp. 1109–1112, 2003.

[110] J. Petereit, T. Emter, C. W. Frey, T. Kopfstedt, and A. Beutel, "Application of hybrid a* to an autonomous mobile robot for path planning in unstructured outdoor environments," in *ROBOTIK 2012; 7th German Conference on Robotics*, pp. 1–6, VDE, 2012.

[111] B. D. Anderson and J. B. Moore, *Optimal control: linear quadratic methods.* Courier Corporation, 2007.

[112] F. Blanchini and S. Miani, *Set-theoretic methods in control*, vol. 78. Springer, 2008.

[113] I. Kolmanovsky and E. G. Gilbert, "Theory and computation of disturbance invariant sets for discrete-time linear systems," *Mathematical problems in engineering*, vol. 4, no. 4, pp. 317–367, 1998.

[114] A. De Luca, G. Oriolo, and C. Samson, "Feedback control of a nonholonomic car-like robot," *Robot motion planning and control*, pp. 171–253, 2005.

[115] R. Van Der Merwe and E. A. Wan, "The square-root unscented kalman filter for state and parameter-estimation," in *2001 IEEE international conference on acoustics, speech, and signal processing. Proceedings (Cat. No. 01CH37221)*, vol. 6, pp. 3461–3464, IEEE, 2001.

[116] H. G. Tanner and K. J. Kyriakopoulos, "Backstepping for nonsmooth systems," *Automatica*, vol. 39, no. 7, pp. 1259–1265, 2003.