Application of Nonlinear Model Predictive Control to Holonomic Mobile Robots without Terminal Constraints or Costs

by

Joseph Cenerini

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Mechanical and Mechatronics Engineering

Waterloo, Ontario, Canada, 2020

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

The Introduction, Background and Conclusion chapters (1, 2 and 7 respectively) do not present any contributions but were solely written by the author. Prof. Soo Jeon, Pr. William Melek and Dr. Mohamed Said provided feedback and suggested revisions to these and the remaining chapters in this thesis.

Chapter 3 is an excerpt from [67] for which the author of this thesis is a co-author. The author's contribution in this paper include the adaptation of a continuous time growth bound for the value function from the continuous to discrete time, which allows to prove the asymptotic stability of the system. Other contributions include showing that the asymptotic stability holds for the shortest possible horizon and confirming the theoretical work through simulation and experiment.

The contributions in Chapters 4 and 5 are in many ways extensions of the work done in Chapter 3, extending the work done for a point stabilization controller to an acceleration constrained and path following controllers. The algorithms in both of these cases were developed for a holonomic robot by the author. For Chapter 4, the author applies the theory of barrier developed in [24] to develop an admissible set for an acceleration constrained holonomic robot. Using this admissible set the author develops a growth function which bounds the value function for this algorithm. This allows us to show the system is asymptotically stable for a sufficiently long prediction horizon.

The formulation of the model predictive path following problem in Chapter 5 is an adaptation of the work done in [68] for a non-holonomic to the holonomic case. The author adapts the algorithm to take advantage of the extra degree of freedom provided by the holonomic robot. Again growth bounds which are adapted from [68] are developed to ensure the asymptotic stability of the system for a sufficiently long prediction horizon. The work in both of these Chapters was done in collaboration with Dr. Mohamed Said.

Finally work presented in Chapter 6 are entirely the authors own. By taking concepts of the minimum effort controller presented in [11] the author develops a novel nonlinear MPC algorithm which uses the Newton-Euler recursive dynamic model of a robotic arm. This controller, explicitly minimizes torque while also ensuring that the desired end effector position is reached, without terminal costs or constraints, using a time dependent preference function.

Abstract

The use of mobile robots greatly enhances the capability and versatility of industrial robots compared to their fixed counterpart. However, successfully deploying autonomous mobile robots is challenging and requires accurate mapping, localization, planning and control. Herein, we focus the application of nonlinear model predictive control to a holonomic mobile robot while ensuring closed loop asymptotic stability. This thesis presents three main contributions in this area.

We begin with a study of the regulation control of a holonomic mobile robots with limits on acceleration under a Model Predictive Control (MPC) scheme without stabilizing terminal conditions or costs. Closed-loop asymptotic stability is ensured by suitably choosing the prediction horizon length. We first compute a set of admissible states for a holonomic mobile robot with limits on acceleration using the theory of barriers. Then, by deriving a growth (envelope) function for the MPC value function, we determine a stabilizing prediction horizon length. Theoretical results are confirmed through numerical simulations.

Next the Model Predictive Path Following Control (MPFC) of holonomic mobile robots is considered. Here, the control objective is to follow a geometric path, where the time evolution of the path parameterization is not fixed a priori, but rather is left as an extra degree of freedom for the controller. Contrary to previous works, we show that the asymptotic stability can be ensured for the resulting closed-loop system under MPFC without terminal constraints or costs. The analysis is based on verifying the cost-controllability assumption by deriving an upper bound on the MPFC finite-horizon value function. This bound is used to determine a stabilizing prediction horizon. The analysis is preformed in the discrete-time setting and results are verified by numerical simulations.

Finally, a dual-objective MPC algorithm for an open chain manipulator is presented, which, as a primary objective takes the end effector to a desired position and as a secondary goal minimizes the effort required from the actuators. To achieve this, a recursive Newton-Euler algorithm is used to calculate the system dynamics and to determine the actuator torques. The proposed method is based on a time varying cost function which, as time goes on, reduces weight on the secondary objective. This allows the controller to minimize torque at the start of the maneuver without affecting the ability of the system to reach the desired end effector position.

By taking advantage of the inherent benefits of MPC such as the ability to naturally incorporate constraints and to manipulate the objective function, the algorithms proposed here offer control solutions applicable in a variety of mobile robotic applications.

Acknowledgements

I would like to thank and recognize the support and contributions of my supervisors Prof. Soo Jeon and Prof. William Melek who guided and advised me throughout my MASc. program. I would also like to extend a special thank you to Dr. Mohamed Said who guided and mentored me through a lot of my research.

I would like to thank Prof. Baris Fidan and Pr. Christopher Nielsen for taking the time to review this thesis and for showing interest in my research.

Additionally I would also like to recognize and thank the other lab members of the Mechanical Systems and Control Lab group, I'd like to especially thank Jeong Woo Han, Shiyi Yang and Mostafa Osman for their help and collaboration on the autonomous warehouse robot project.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

CARIMA Controlled Auto Regressive Integrated Moving Average , 7

CARMA Controlled Auto Regressive Moving Average , 7

DMC Dynamic Matrix Control , 7

EKF Extended Kalman Filter , 2

GPC General Predictive Control , 7

HVAC Heating Ventilation and Air Conditioning , 8

LQR Linear Quadratic Regulator , 28

MIMO Multi-Input Multi-Output , 7

MOOP Multi-objective optimization problem , 87

MPC Model Predictive Control iv, 1, 7, 31, 47, 66, 81

MPFC Model Predictive Path Following Control iv, 66

MPHC Model Predictive Heuristic Control , 7

NLP Nonlinear Programming Problem , 41, 75

OCP Optimal Control Problem , 6, 9, 41, 53, 68, 82, 86

PID Proportional-Integral-Derivative , 9, 82

POE Power of Exponential , 82

ROS Robot Operating System , 43

RRT Rapidly-Exploring Random Tree , 2

SISO Single-Input Single-Output , 7

List of Symbols

α Performance bound for an MPC problem.

β $\mathcal{KL}$ class function comparison function.

γ Bound on the value function for a discrete time MPC problem.

Γ Joint torques for robotic arm.

ϕ^{Δ} Superscript Δ denotes the change in a function $\phi^{\Delta}(x,u) = \phi(x^{+}) - \phi(x)$.

δ Continuous time control horizon.

η $\mathcal{K}$ class function comparison function.

Θ Rotation matrix where $\Theta \in SO(3)$.

θ Virtual state used to parametrize the desired path.

κ_f Control input under which terminal set X_f is forward invariant.

λ Ratio of the control weight to the position error weight, $\lambda = q_i/r_i$.

$\mu_{N,m}$ MPC feedback control law for prediction horizon N and control horizon m. If $m = 1$, we simply use μ_N.

τ Sampling time of a discrete time system where $\tau > 0$.

ϕ Preference function for muli-objective MPC.

ω Normalized angular velocity of a frame in space.

ξ Value of the adjoint equation for the derivation of the admissible set boundary.

a, b Dimensions of holonomic mobile robot.

$\mathcal{A}$ Admissible set of states, obtained through the theory of barriers.

$\mathbf{b}$ Position vector between two frames, where $\mathbf{b} \in \mathbb{R}^3$.

B Bound on the value function for a continuous time MPC problem.

$\mathbf{C}$ Matrix accounting for Coriolis forces for robotic arm dynamics.

c Growth sequence or function which bounds the running costs for MPC. Where $\gamma_i = \sum_{j=0}^{i-1} c_j$ in discrete time or $B(t) = \int_0^t c(s)ds$ in continuous time.

D Derivative of a function.

$\mathbf{e}_0$ Error between the initial position and the desired position on the path, $\mathbf{e}_0 = \mathbf{p}(\theta_0) = \mathbf{x}_0$.

F Cost function $F : \mathbb{R}^n \to \mathbb{R}$ on the final state of an OCP.

f System model $f : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^n$ relating state $\mathbf{x} \in \mathbb{R}^n$ and input $\mathbf{u} \in \mathbb{R}^p$ to the next state $\mathbf{x}^+ \in \mathbb{R}^n$.

g Constraint functions for OCP.

$\mathbf{G}$ Matrix accounting for gravitational forces for robotic arm dynamics.

H Lower level kinematic model for holonomic robot $H \in \mathbb{R}^{4\times3}$.

$\mathbb{I}$ Set of indices of active constraints.

$\mathbf{I}_i$ Identity matrix of size $i \times i$.

I_i Is the inertia mass matrix of link i about joint i.

J_N Objective function over the prediction horizon, N.

$\mathcal{K}$ A comparison function with properties in Definition A.2.1.

$\mathcal{KL}$ A comparison function with properties in Definition A.2.2.

$k_{m/n}$ Maneuver time for sub-optimal maneuver in discrete time.

k The time step index for a discrete time system, where $t = k\tau$ and $k \in \mathbb{N}$.

L Lie derivative of a function.

ℓ Running costs $\ell : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}$ for optimal control problem.

$\mathbf{M}$ Mass matrix for robotic arm dynamics.

M_i Transformation $\mathcal{T}_{i-1,i}$ when $q_i = 0$, $M_i \in SE(3)$.

m Control horizon in optimal control problem.

$\mathbb{N}$ The set of natural numbers.

n Number of states for a system.

N Prediction horizon length in an optimal control problem.

$\hat{N}$ Minimum prediction horizon length to guarantee closed loop MPC stability.

$\mathbf{p}$ System coordinates of the desired path.

$\mathcal{P}$ The desired path of the system.

$\mathcal{PC}(I, \mathbb{R})$ A piecewise continuous function, $c : I \to \mathbb{R}$.

p Number of control inputs.

$\hat{\mathbf{p}}$ Points on a piecewise continuous linear function which connect the via points $\tilde{\mathbf{p}}$.

$\tilde{\mathbf{p}}$ Via points given for the path following problem.

$\mathbf{q}$ Joint coordinates for robotic arm.

Q Positive semi-definite weight on the state error.

q_i Entry (i, i) in the Q matrix.

$\mathcal{Q}$ The set of allowable joint positions.

$\mathcal{R}$ Nonlinear system matrix model, $\mathcal{R} \in \mathbb{R}^{n \times p}$.

$\mathbb{R}_{\geq 0}$ The set of non-negative real numbers.

$\mathbb{R}^+$ The set of positive real numbers.

$\mathbb{R}$ The set of real numbers.

R Positive semi-definite weight on the control input.

r_i Entry (i, i) in the R matrix.

r Wheel radius of holonomic mobile robot.

$SE(3)$ The special Euclidean group.

$SO(3)$ The special orthogonal group.

S Positive semi-definite weight on an additional objective in the running costs.

$\mathcal{S}$ Screw axis.

$\hat{T}$ Minimum time horizon to guarantee stability for closed loop MPC.

T Time horizon over OCP.

$t_{k/m/n}$ Maneuver time for sub-optimal maneuver in continuous time.

t Time $t \in \mathbb{R}_{\geq 0}$.

$\mathcal{T}_{i,j}$ Transformation matrix between frames i and j, where $\mathcal{T}_{i,j} \in SE(3)$.

$\mathbf{u}_{ref}$ Control input required to exactly follow path $\mathcal{P}$.

$\mathbf{u}$ Control input for robot system, $\mathbf{u} \in \mathbb{R}^p$.

u General control input.

$\mathbf{u}_w$ Vector of wheel speeds for a holonomic mobile robot.

$\mathcal{U}^N(x_0)$ Set of admissible functions over horizon, N with starting position x_0.

U Set of allowable control inputs.

u^* Superscript $\_^*$ denotes the solution to an optimization problem. In this case u^* denotes the solution to an OCP.

$\mathbf{V}_i$ Is the generalized velocity of the link i frame in frame i coordinates.

$\mathbf{v}$ Linear velocity of frame in space.

v Virtual control input for path parameter, θ.

$\mathcal{V}$ Set of admissible path control inputs, v.

V Lyapunov function.

V_N Minimum of the objective function, J_N.

$\mathcal{W}^N(z_0)$ Set of allowable augmented control sequences over the horizon, N starting from state $\mathbf{z}_0$.

W Set of admissible augmented control inputs, $W := U \times \mathcal{V}$.

$\mathbf{w}$ Augmented system control input which includes the virtual control, v.

$\mathbf{W}_i$ Is the generalized force (or wrench) transmitter from link $i-1$ to link i.

x General system state.

x_0 Subscript $_0$ denotes the initial state of a system.

$\mathbf{x}$ Mobile robot system state, where $\mathbf{x} \in \mathbb{R}^n$.

$x^\star$ The superscript * denotes the desired state in a point stabilization problem.

$x_u(\cdot; x_0)$ A state trajectory resulting from input function u, starting from initial position x_0.

$\bar{x}$ Accent $\bar{\cdot}$ represents an upper limit of a state or input. Similarly $\underline{\cdot}$ denotes a lower limit.

x_i Subscript $_i$ denotes the index of a vector $\mathbf{x}$.

X Set of allowable states.

x^+ Superscript $^+$ denotes the next system state, $x^+ = f(x, u)$.

X_f Set to which the final state of an OCP is constrained.

$\mathbf{y}$ Value of the states at a point tangent to the boundary.

Z Set of allowable augmented states for path following problem, $Z := X \times [\bar{\theta}, 0]$.

$\mathbf{z}$ Augmented system state which includes the path parameter, θ.

Chapter 1

Introduction

In this thesis, we present nonlinear MPC algorithms designed to improve the control of mobile robots. These include an algorithm for the point to point stabilization of a holonomic robot, which is then extended to the case where the acceleration of the robot is constrained as well as to a model predictive path following controller. An MPC algorithm which aims to minimize the torque for a redundant robotic arm is also presented. These algorithms can be deployed in a broad range of applications such as manufacturing and warehousing where robust and stable control of mobile robots is required.

1.1 Motivation

Autonomous mobile robots have been and continue to be developed for a wide range of applications, whether it be in manufacturing or logistics as a way to deal with rising labour costs [22] or to to complete tasks in an environment which would be otherwise inaccessible to humans [87]. In order for autonomous mobile robots to gain widespread acceptance, they must be able to perform their desired function safely, effectively and consistently. Designing autonomous mobile robots which meet these criteria is no simple task and involves the integration of a number of complex bodies of knowledge. In order to achieve successful design and deployment of a mobile robot the following are required [87]:

- Perception - the process of interpreting information from various sensors, which could include cameras, lidar, odometers, accelerometers, etc. to extract features from the robots environment.

- Mapping - in order to successfully navigate in its environment, a mobile robot needs access to a map of its surroundings, which is either known apriori or built from the information extracted from sensors.
- Localization - the robot then needs to be able to localize or find itself on the map. Using the perception data, the robot can obtain a pose estimate using a localization algorithm such as an Extended Kalman Filter (EKF) or a particle filter.
- Planning - given an objective or command the robot needs to find a way or a path to reach this objective. Common path planning algorithms that achieve this for mobile robots include A*, Dijkstra or Rapidly-Exploring Random Trees (RRTs).
- Motion Control - finally a controller needs to provide the robots actuators with the necessary commands to follow the planned path. These controllers usually take into account the robot kinematics and dynamics and use the pose estimation as feedback to get the robot to the desired path.

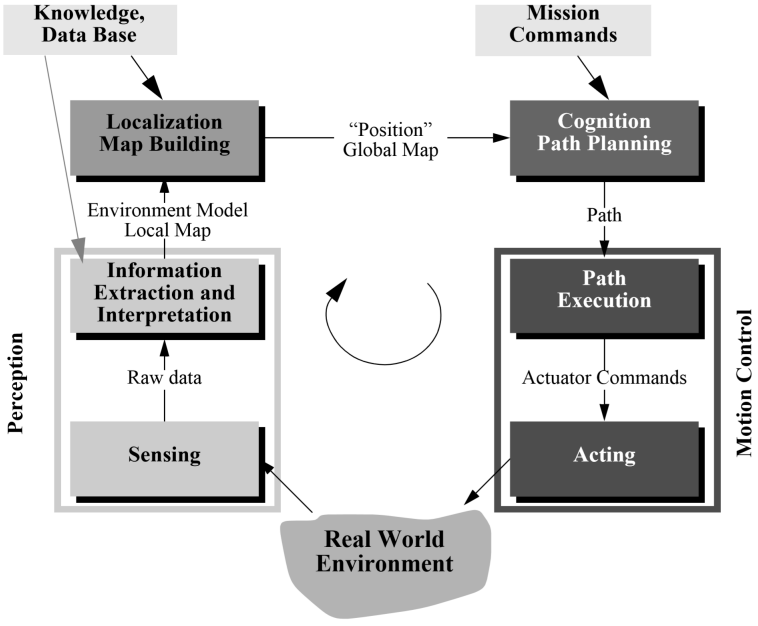

Figure 1.1: Scheme for autonomous mobile robot deployment, figure is reprinted from [87]

The relationship between the key capabilities for an autonomous mobile robot discussed above is shown in Fig. 1.1. Without effective and robust algorithms to provide the robot

with each of these aspects, an autonomous robot is unable to operate efficiently. From here on, this thesis focuses on the motion control portion of mobile robot deployment.

1.2 Robotic Platform

The work in this thesis focuses on the design of optimal control algorithms for a holonomic mobile robotic platform, specifically the Summit-XL-S from Robotnik shown in Fig. 1.2. This robot employs mecanum wheels, which have rollers mounted on its outer circumference and allow the wheel to slip in the free sliding direction, here at a ± 45 degree angle from the orientation of the robot. Rather than being steered, these wheels allow the robot to independently control its forward, lateral and angular velocities [59]. The ability to move in any direction independently gives holonomic mobile robots an advantage when working in tight spaces, which is beneficial when considering warehouses or manufacturing environments.

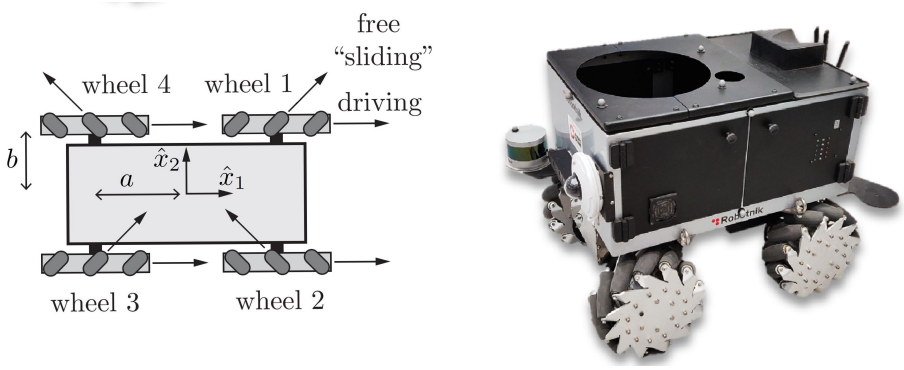

Figure 1.2: Holonomic mobile robot with mecanum wheels (*Summit-XL-Steel* [81]) with the driving and the free sliding directions for each wheel type is shown by corresponding vectors. The figure is partially reprinted from [59].

To benefit from the advantages offered by holonomic robots, a controller is required which can take into account the nonlinearities of the system, can guarantee asymptotic stability and is robust to disturbances. In cluttered environments, being able to apply constraints to the robot state, such as limiting the proximity to obstacles or boundaries is also important. Given that for the considered environments, the robot may be required to share a space with humans, applying constraints on the actuator output also becomes important for safety reasons. In this thesis, by taking advantage of the properties of model

predictive control, we develop control algorithms which achieve given control objectives as well as the aforementioned goals.

1.3 Motion Control

There have been several controllers developed for motion control of a holonomic robot. The control objectives for these controllers are typically categorized as follows:

- Point stabilization - involves stabilizing the robot to a desired pose. Planning the path required to reach the desired point is also sometimes included. Within the approaches based on optimal control, this may be achieved using real-time near-optimal trajectory generation [45] or MPC [7].
- Trajectory tracking - here the robot follows a desired trajectory with predetermined speed, either defined as a set of points or a function. One example presented in [20] uses a controller based on ideal reference velocities. Another in [43] uses an adaptive back-stepping approach to achieve stabilization and trajectory tracking.
- Path-following - similar to the trajectory tracking problem, the goal is to follow a function or set of points, however there is no time assignment to the desired positions for path following. A version of this controller is presented in [57] using the linearized inverse kinematic model. Another formulation using linearized MPC can be seen in [46].

This thesis will address the implementation of MPC for a point stabilization controller, which unlike previous works [7, 46, 48] guarantees stability without the use of terminal constraints or penalties. This controller is then extended to constrain the robot's acceleration while maintaining its guaranteed stability. The path-following problem presented in Section 2.4 is similarly considered, again the algorithm presented herein differs from previous applications of MPC for this purpose [47, 49] as it offers a proof of asymptotic stability for a model path predictive controller without terminal costs or constraints.

1.4 Structure of This Thesis

This thesis is organized as follows, in the next chapter some of the background information needed for the remainder of this thesis is presented. This will include an introduction

of the MPC algorithm in Section 2.1.2, including the motivation for using it in robotics applications. Since it is the subject of study in much of this paper, the history and current state of MPC stability is covered in the following section (2.2). Finally, the kinematic model for a holonomic robot and the path following problem formulation are presented in the last two sections.

In the chapters following the background, we present the main contributions of this thesis. Chapter 3, an excerpt from a joint journal paper published in Robotics and Autonomous Systems, presents an MPC scheme without terminal constraints or costs for point stabilization in discrete time. This chapter presents a rigorous proof of closed-loop asymptotic stability for this system, using the concept of cost controllability following the theory presented in 2.2. Furthermore, it is shown that the system will be stable for the shortest possible prediction horizon of $N = 2$. Finally, through numerical simulation and experiments, we verify these theoretical results.

The work from Chapter 3 is extend in Chapter 4, where the same system is considered, with an additional constraint on acceleration rather than only on velocity and position. The acceleration constraints ensure smoother operation of the robot and reduce wear on its components. Using the theory of barriers, an admissible set where the system is recursively feasible is constructed. With an initial state inside this admissible set, the proof of closed-loop asymptotic stability for this system is again ensured using cost controllability. In this case, stability is ensured for MPC with a sufficiently long prediction horizon.

Another extension of the work done in Chapter 3 is presented in Chapter 5, where a model predictive path following control (MPFC) algorithm is presented for the holonomic mobile robot. In this chapter, by parametrizing the path we allow the MPC algorithm to choose the optimal velocity along the path as an extra degree of freedom. With its ability to control its lateral, longitudinal positions and orientation independently, the holonomic robot has the ability to follow any arbitrary path accurately. For this formulation of the MPFC, using the concept of cost controllability, again we can prove closed-loop asymptotic stability for a sufficiently long prediction horizon.

Finally, Chapter 6 introduces a dual objective MPC control algorithm which minimizes the effort to move to a desired position while at the same time ensuring that the end effector does in fact reach that position with no steady state error. Reducing the effort required, defined here as the sum of the square of the joint torques, reduces the stress on the joints. In this chapter we apply this algorithm to a planar arm with three revolute joints. This system is modeled using a recursive Newton-Euler algorithm. We are able to show a significant reduction in torque when comparing this to a standard MPC formulation. By comparing the Newton-Euler model to a classical dynamic model obtained from the

Lagrangian of the system, we can see it produces similar results but that the Optimal Control Problem (OCP) can be solved significantly faster.

Chapter 2

Background

In this chapter, some of the background theory and concepts on which our contributions rely are presented. This includes background on nonlinear model predictive control, its stability analysis, the presentation of kinematic models for holonomic robots and the path following problem.

2.1 Model Predictive Control

Introduced in the late seventies, [16] MPC algorithms have evolved to be used in a wide range of applications and industries. Early iterations of the MPC algorithms were known as Dynamic Matrix Control (DMC) [23] and Model Predictive Heuristic Control (MPHC) [80]. These methods built upon previous work done in minimum time optimal control, linear programming and receding horizon optimal control [16]. These used a dynamic model based on the response of the system to a step or impulse input. Using an optimization algorithm, the inputs of the system were set to minimize the output error subject to some process constraints. Later, in a formulation called General Predictive Control (GPC) [19], to overcome stability problems, another penalty term was added to the control input, creating the quadratic value function we see in modern implementations of MPC. GPC used mostly Single-Input Single-Output (SISO) systems with polynomial based models which aimed to minimize the number of parameters, example of which include Controlled Auto Regressive Moving Average (CARMA) or Controlled Auto Regressive Integrated Moving Average (CARIMA) [91] algorithms. Straightforward extensions of GPC to Multi-Input Multi-Output (MIMO) systems are provided in [26] using a polynomial model and [52] with a state-space model.

Although this provides the basis of much of the MPC algorithms used in industry, it still remained difficult to prove stability and robustness of these algorithms. Much of the later research on MPC, including this thesis has been focused on proving stability [91]. We review some of this research in Section 2.2.

MPC includes a broad range of control strategies that consider the following criteria:

1. Explicit use of a model to predict the process output over a control horizon.
2. Calculation of an optimal control sequence, by minimizing some cost.
3. The control input is determined in a receding fashion. In other words, the first control input of the control sequence for a given horizon length is applied to the system. A new optimal control sequence is calculated for the next time instance.

These algorithms are applied in a broad range of industries, from its early adoption for chemical processes [75], to more recently extending among others to power electronics [92], Heating Ventilation and Air Conditioning (HVAC) systems [86], aircraft control [36] and robotics [34, 97].

2.1.1 Motivation

The widespread adoption of MPC is due to a number of distinct advantages which make it appealing [16, 91], such as

1. It can be used to control a wide variety of processes, from systems with nonlinear dynamics, long delay times, non-minimium phase and unstable ones.
2. Easily deals with multi-variable system.
3. Intrinsically compensates for dead times.
4. Naturally incorporates feed forward control to compensate for measurable disturbances.
5. The resulting controller is an easy to implement linear control law.
6. The treatment of constraints is conceptually simple and these can be systematically included in the design process.

7. Useful when future references are known.

8. All MPC algorithms are based on the same basic principles which are easily extended to more complex cases.

Even with these advantages MPC is not ideal in all situations as it has some drawbacks.

1. The derivation of the control law is more complex than a classical Proportional-Integral-Derivative (PID) controller. It can require the solution of a complex constrained optimization problem. This can be any issue for real time applications in robotics.

2. An appropriate system model is required, as the performance of the MPC algorithm depends on having an accurate model.

When considering mobile robotic applications, its ability to handle MIMO systems and the ability to naturally incorporate constraints makes MPC especially advantageous, as it restricts the position of the robot to a safe region and ensures that the actuator outputs remain within their limits [53]. The heavier computational burden also needs to be considered for a robotics applications as these applications typically require a high control frequency and solving the nonlinear constrained OCP may be infeasible in real time applications.

2.1.2 MPC Algorithm in Discrete Time

In this section we take a more in depth look at the three main components of MPC algorithms, the predictive model, the objective function and the control law.

Predictive Model

Unlike other controls methods, such as PID controllers, MPC explicitly considers the future implications of its control actions [83]. A model is needed to capture the dynamics of the system we are trying to control and to be able to obtain future system states for the length of the prediction horizon. Though there a number of different formulations of system models [16], here we focus on the non-linear system model without disturbances as this is what we use in the applications throughout this thesis. We formulate this system model as

$$\dot{x}(t) = f(x(t), u(t)), \quad x(0) = x_0 \tag{2.1}$$

where x is our system state, u is the input to the system and x_0 is the initial condition. Here $f(x(t), u(t))$ is the system model, that relates the input to the change in state. We can formulate the system dynamics in discrete time as

$$x(k+1) = f_d(x(k), u(k), \tau), \quad x(0) = x_0 \tag{2.2}$$

where k is time discretized by some constant time step τ. The discrete time system model is often more useful in practice, as control inputs are generally applied in discretized time steps.

The system dynamics are then propagated for the length of the prediction horizon, N. For some initial state x_0, the states over the prediction horizon can be defined as follows

$$x_u(k, x_0) = \begin{pmatrix} x(0) \\ x(1) \\ \vdots \\ x(N) \end{pmatrix} = \begin{pmatrix} x_0 \\ f_d(x_0, u(1), \tau) \\ \vdots \\ f_d(x(N-1), u(N), \tau) \end{pmatrix} \tag{2.3}$$

Having an accurate model is important as this determines the input to the objective function, the next element of the MPC algorithm.

Objective Function

Let us consider a cost function at each time step in the prediction horizon. The usual formulation of MPC penalizes the error between the predicted future states in (2.3) and some reference signal $x^\star(k)$ and the control input $u(k)$. This leads to the running costs

$$\ell_k(x(k), u(k)) = (x(k) - x^\star(k))^\top Q(k)(x(k) - x^\star(k)) + u(k)^\top R(k)u(k) \tag{2.4}$$

at each time step $k \in \{0, \ldots, N\}$. $Q(k)$ and $R(k)$ are positive definite weighing matrices, either constant or some function of the time step for adaptive weights. Considering these running costs over the entire prediction horizon N we obtain the objective function

$$J_N(x, u) = \sum_{k=0}^{N} \ell_k(x(k), u(k)). \tag{2.5}$$

Using the control sequence u as the optimization variable, we set up an optimal control problem to minimize the objective function

$$V_N(x_0) = \min_{u \in U^N} J_N(x_u(k, x_0), u). \tag{2.6}$$

In reality, the control problem will be subject to some constraints or limitations. This could include limitation on the actuator outputs, such as position, velocity or force limits for a robotic system. There can also be physical limits in the output space such as obstacles or limits in speed for safety reasons. The advantage of MPC is that these constraints can be directly accounted for as constraints on the optimization problem. We define the admissible set of states and controls as

$$X := [\underline{x}_1, \bar{x}_1] \times \ldots \times [\underline{x}_n, \bar{x}_n] \quad \text{and} \quad U := [\underline{u}_1, \bar{u}_1] \times \ldots \times [\underline{u}_m, \bar{u}_m] \tag{2.7}$$

for $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^m$. Furthermore, a control sequence $u = (u(0), u(1), \ldots, u(N-1)) \in U^N$ of length $N \in \mathbb{N}$ is said to be admissible, for state x_0, denoted by $u \in \mathcal{U}^N(x_0)$ if the state trajectory $x_u(\cdot; x_0)$ from (2.3), satisfies $x_u(k, x_0) \in X$ for all $k \in \{0, 1, \ldots, N\}$.

We can set up the constrained optimization problem as

$$\begin{aligned} V_N(x_k) = &\min_{u \in \mathcal{U}^N(x_k)} J_N(x_u(k, x_k), u) \\ \textbf{s.t.} \quad &\mathbf{x}(k) = \mathbf{x}_k, \\ &\mathbf{x}(i) \in X \quad \forall i \in [k, k+N] \\ &\mathbf{u}(i) \in U \quad \forall i \in [k, k+N) \end{aligned} \tag{2.8}$$

Note here that we a value function without terminal costs or constraints. Adding terminal costs or constraints, as seen in Section 2.2, makes guaranteeing asymptotic stability more straightforward. However these cause the algorithm to incur heavier computational costs and requires a large prediction horizon in order to have a large feasible region and is therefore not often used in practice [38].

Control Law

The solution to OCP (2.8) will be a control sequence u^* which minimizes the value function over the prediction horizon N, namely $V_N(x_0) = J_N(x_{u^*}(k, x_0), u^*)$. The control law μ applied to the system will be the first m inputs in the sequence, where m is the control horizon (setting $m = 1$ is common).

$$\mu_{N,m}(k + i, x_k) = u^*(i, x_k), \quad \text{for } i \in \{0, \ldots, m-1\} \tag{2.9}$$

where $x_k = x(k)$ is the current state of the system. The control law is then applied to system (2.2). We then increment k and repeat the process of solving OCP (2.8) and applying the first control input of the solution.

2.1.3 Multiple Shooting Method

For nonlinear problems, the propagation of the control inputs through the system as shown in (2.3) can become very complex especially with long prediction horizons. The method shown here is referred to as single shooting [27]. An alternative, more efficient way of propagating systems dynamics is to formulate the optimal control problem using the multiple shooting method [13]. Here the dynamics of the system are enforced through constraints. Adding the state as an optimization variable, the OCP becomes

$$
\begin{aligned}
V_N(x_k) = & \min_{u \in \mathcal{U}^N(x_k),\, x \in X} J_N(x_k, u) \qquad (2.10)\\
\textbf{s.t.} \quad & \mathbf{x}(k) = \mathbf{x}_k, \\
& \mathbf{x}(i+1) = f_d(x(i), u(i), \tau) \quad \forall i \in [k, k+N) \\
& \mathbf{x}(i) \in X \quad \forall i \in [k, k+N] \\
& \mathbf{u}(i) \in U \quad \forall i \in [k, k+N)
\end{aligned}
$$

In this manner, the optimization does not become increasingly complex further along the prediction horizon. This method of formulating the OCP is used throughout the later chapters of this thesis.

2.2 Stability

In the design of a controller, stability is an important consideration. Proving the stability of a controller can be a challenging undertaking and has been the subject of much research. Within this section we present some of the theories behind the stability of nonlinear MPC. Grune's work in [37, 39] on nonlinear MPC stability without terminal constraints or penalty presented in Subsection 2.2.2 is especially important, as we use the theorems presented therein to prove the stability of the nonlinear closed-loop MPC controllers in later chapters.

2.2.1 Stability of Constrained Nonlinear MPC

In this section we introduce the stability characteristics of systems under MPC feedback control, we refer to Appendix A for a more general review of stability for nonautonomous

nonlinear systems. Though the stability theorems used in this thesis are presented in Sections 2.2.2 and 2.2.3, this section is meant to give context to these theorems. Theory on constrained nonlinear MPC stability is preceded by significant contributions in stability analysis for unconstrained linear systems [65]. In this section, we focus on the stability of constrained non-linear systems with terminal constraints or costs.

Throughout this section, for a given function $\phi(x)$, $\phi^{\Delta}(x,u)$ denotes the change in $\phi(\cdot)$ as the state moves from x to $x^+ = f(x,u)$, namely, we say that

$$\phi^{\Delta}(x,u) := \phi(x^+) - \phi(x). \tag{2.11}$$

Lyapunov stability was first introduced as a tool to prove stability in non-linear MPC in [17], where the value function is used as a Lyapunov function to establish the stability of an unconstrained receding horizon control scheme with a terminal equality constraint. These results were extended by [64] [50] to discrete time non-linear systems, again using the value function as a Lyapunov function for constrained nonlinear systems. From then on using the value function as a Lyapunov function became the common method of showing stability for an MPC control scheme.

Another formulation which instead of putting a terminal constraint on the final value of the state, adds a terminal cost $F(\cdot)$ to the final state is introduced by [10]. The terminal cost therein were chosen to be $F(x) = 1/2x^\top P_f x$, and where P_f is the terminal value of the Riccati difference equation. Another proposal of terminal costs is presented in [78], where the terminal cost $F(\cdot)$ is chosen to be the infinite horizon value function associated with the stabilizing controller for a linear system subject to convex control constraints. This is equivalent to having infinite horizon costs with a finite control horizon. This allows us to take advantage of the benefits of infinite horizon control such as its robustness and stability [65].

The next formulation of the stability proof combines the terminal costs with a terminal constraint set, where ideally the terminal costs $F(\cdot)$ are the infinite horizon value function V_∞. Calculating these infinite costs for non-linear systems however is not possible. Instead $F(\cdot)$ is chosen to approximate the value function V_∞ in a suitable neighbourhood of the origin. This approach is introduced in [89] and is finally used to characterize the stability of constrained non-linear systems in [25] [18]. These are considered 'quasi-infinite horizon prediction control' because they use approximate infinite horizon costs for their finite horizon control problem.

Problem Formulation

In this section, since we review stability work for MPC schemes with terminal constraints and costs we consider an alternative value function to (2.5). Here terminal costs $F(\cdot)$ are added such that

$$J_N(x_0, u) = \sum_{k=0}^{N} \ell(x(k), u(k)) + F(x(N)). \tag{2.12}$$

This objective function is used in OCP (2.10) for some initial condition x_0, from which the optimal control sequence $u^*(x_0)$ and the corresponding value function $V_N(x_0)$ are obtained by solving (2.6). It follows that a corresponding optimal trajectory, $x_{u^*}(k, x_0)$ for $k \in \{0, 1, \ldots, N\}$, can be obtained by applying the optimal control sequence. We follow the typical formulation for a feedback law $\mu_N(x) = u^*(0, x)$ which is propagated through the system dynamics (2.2) abbreviated here as $x^+ = f(x, \mu_N(x))$.

Direct Method

Using the formulation in [65], we obtain conditions on the terminal costs $F(\cdot)$, the terminal set X_f and the control function $\kappa_f(\cdot)$ which renders the terminal set X_f forward invariant, such that

$$V_N^{\Delta}(x, \mu_N(x)) + \ell(x, \mu_N(x)) \leq 0, \tag{2.13}$$

In this method, rather than directly computing $V_N(x^+)$ an upper bound is calculated using a prolonged control sequence $\tilde{u}(x)$. Let us first define an admissible control sequence u which steers an initial state $x_0 \in X_n$ to a state $x_u(n, x_0) \in X_f$ in n steps or less. For this admissible sequence $u = \{u(0), \ldots, u(n-1)\}$, $u(k) \in U$ for $k \in \{0, 1, \ldots, n-1\}$ and $x_u(k, x_0) \in X$ for all $k \in \{0, 1, \ldots, n\}$. We define another set X_N as the set of states which can be controlled to X_f by the solution to the OCP (2.10). We now wish to determine an admissible (sub-optimal) control sequence $\tilde{u}$ for x^+, which is then used to find an upper bound for the optimal value function $V_N(x)$.

Truncating the solution to the OCP, $u^*(x)$, we obtain an admissible control sequence steering x^+ to $x_{u^*}(N, x_0) \in X_f$ as $\{u^*(1, x_0), \ldots, u^*(N-1, x_0)\}$. Now to obtain a N-step control law steering x^+ to $x_{\tilde{u}}(N, x^+) \in X_f$, we can simply add an additional control input to the aforementioned sequence, giving the sequence

$$\tilde{u} := \{u^*(1, x), \ldots, u^*(N-1, x), \kappa_f(x_{u^*}(N, x_0))\}$$

where a state $x \in X_f$ is forward invariant under the control input $\kappa_f(\cdot)$, i.e. $f(x, \kappa_f) \in X_f$, $\forall x \in X_f$. The trajectory generated from the control input $\tilde{u}$ is denoted as $\tilde{x}$ where $\tilde{x}(0) = x^+ = x_{u^*}(1, x_0)$.

The cost associated with $\tilde{u}$ and $\tilde{x}$ can be calculated as

$$J_N(\tilde{x}, \tilde{u}) = V_N(x_0) - \ell(x_0, \mu_N(x_0)) - F(x_{u^*}(N, x_0)) \\ + \ell(x_{u^*}(N, x_0), \kappa_f(x_{u^*}(N, x_0))) + F(f(x_{u^*}(N, x_0), \kappa_f(x_{u^*}(N, x_0))).$$

We can now express our upper bound on the value function $V_N(x^+)$ as

$$V_N(x^+) \leq J_N(\tilde{x}, \tilde{u}) \leq V_N(x_0) - \ell(x_0, \mu_N(x_0)) \tag{2.14}$$

if $F^\Delta(x, \kappa_f(x)) + \ell(x_{u^*}(N, x_0), \kappa_f(x_{u^*}(N, x_0))) \leq 0$. In other words, the change in the terminal costs plus the running costs for x to remain in X_f must be less than or equal to zero. This holds if the terminal cost is a Lyapunov function where $\inf_{u \in U}(F(f(x, u)) - F(x)) \leq 0$ in the neighborhood of the origin and $\kappa_f(\cdot)$ and X_f are chosen appropriately. If this holds, then (2.13) also holds for $x \in X_N$ which is enough to ensure asymptotic stability in that region given that the following assumptions hold:

A 1. *$X_f \in X$, X_f is closed, $0 \in X_f$ (state constraints satisfied in X_f)*

A 2. *$\kappa_f(x) \in U, \quad \forall x \in X_f$ (control constraints satisfied in X_f)*

A 3. *$f(x, \kappa_f(x)) \in X_f, \ \forall x \in X_f$ (X_f is forward invariant under κ_f)*

A 4. *$F^\Delta(x, \kappa_f(x)) + \ell(x_{u^*}(N, x_0), \kappa_f(x_{u^*}(N, x_0))) \leq 0$ ($F(\cdot)$ is a local Lyapunov function)*

Using Terminal Constraints

This variant introduced in [50] constrains the terminal value of the OCP $x(N) = 0$ or $X_f = \{0\}$. This formulation does not employ any terminal costs, so $F(\cdot) = 0$. To maintain an equilibrium point at X_f in this case we can say that no control input is required so, $\kappa_f = 0$. This gives us a definition for $F(\cdot)$ and $\kappa_f(\cdot)$ on X_f, which allows us to check assumptions A1-A4. Since $0 \in X$ and $0 \in U$, A1 and A2 are satisfied. It is also easy to see that $f(x, \kappa_f(x)) = f(0, \kappa_f(0)) = 0 \in X_f$, so assumption A3 is satisfied. Finally, since

$$F^\Delta(x, \kappa_f(x)) + \ell(x_{u^*}(N, x_0), \kappa_f(x_{u^*}(N, x_0))) = F(0, \kappa_f(0)) - F(x) + \ell(0, \kappa(0)) = 0$$

satisfies A4, asymptotic stability for $x \in X_N$ is ensured.

Using a Terminal Constraint Set

In this formulation introduced in [70], MPC steers a system to the set X_f, then once in X_f a local stabilizing controller $\kappa_f(x)$ is employed. This formulation is sometimes called *'dual mode MPC'*.

To steer the system to X_f we can set a set a similar constraint to the terminal state $x(N) = \{0\}$, but instead we use a set of terminal states X_f. Again here we use no terminal penalty, so $F(\cdot) = 0$. We can choose a local controller $\kappa_f(\cdot)$ and set X_f to satisfy assumptions A1-A3. To satisfy A4, $\ell(x, \kappa_f(x)) = 0 \in X_f$. Satisfying these assumptions will ensure that our controller satisfies (2.13) and that $x \in X_N$ will be steered by $\mu_N(\cdot)$ to X_f in finite time.

Using Terminal Costs and Constraint Set

Finally we consider an MPC scheme using both terminal costs and a constraint set for a non-linear constrained system [18, 25]. For this system we choose $\kappa_f(x) = K_f x$ to stabilize the linearized system $x^+ = Ax + Bu$. To satisfy assumptions A1 and A2, we choose $X_f \subset X$ and $\kappa_f \subset U$. Following the formulation in [18], X_f is chosen to be a level set of $F(\cdot) = (1/2)x^\top P x$, where $F(x)$ is a Lyapunov function of the linearized system satisfying

$$F(Ax + B\kappa_f(X)) - F(x) + \bar{\ell}(x, \kappa_f(X)) = 0$$

where

$$\bar{\ell}(x, u) = \beta \ell(x, u)$$

for $\ell(x, u) = (1/2)(|x|_Q^2 + |u|_R^2)$ and $\beta \in (1, \infty)$. In [18], this is achieved by replacing A with $A + \rho I$ for the linearized system dynamics where $\rho \in (0, 1)$. Using $\bar{\ell}$ provides sufficient margin to ensure that A3 and A4 are satisfied, when X_f is a sufficiently small level set of $F(\cdot)$. Satisfying these assumptions is enough to ensure asymptotic stability of the system. The prediction horizon N is chosen to be sufficiently large so that the approximation error is less than $\rho|x|_Q^2$ and

$$V_N^\Delta(x, \mu_N(x)) + (1 - \rho)\ell(x, \mu_N(x)) \leq 0$$

for all $x \in X_f$. We use this relaxed Lyapunov inequality rather than (2.13) to ensure closed-loop asymptotic stability.

2.2.2 Stability of MPC without Terminal Constraints or Costs

In chapters 3 and 4 on point stabilization MPC and chapter 5 on path following MPC, some of the main contributions are the proofs of asymptotic stability for these algorithms. Both cases consider non-linear MPC schemes without terminal constraints or penalties. Each case presents a proof based on the work done in Grune et al. in [37, 39]. In this section we summarize the work presented in those papers.

Problem Formulation

Let us start by considering a nonlinear discrete time system given by (2.2) with $x(k) \in X$ and $u(k) \in U$ for all $k \in \mathbb{N}$. We denote the trajectory for some $u \in \mathcal{U}$ as $x_u(k)$ where $\mathcal{U}$ is the space of allowable control sequences and X and U are the spaces of admissible states and inputs, respectively. We begin by modifying the objective function presented in Section 2.1 to be over an infinite prediction horizon

$$J_\infty(x_0, u) = \sum_{k=0}^{\infty} \ell(x_u(k), u(k)) \tag{2.15}$$

where $\ell \in \mathbb{R}_{\geq 0}$ are the running costs in (2.4). The optimal value function obtained from this minimization problem is

$$V_\infty(x_0) = \inf_{u \in \mathcal{U}} J_\infty(x_0, u)$$

Knowing the optimal value function V_∞, using Bellman's optimality principle, we can show that the optimal feedback law is given by

$$\mu(x_0, \cdot) := \arg\min_{u \in \mathcal{U}^m} \left\{ V_\infty(x_u(m)) + \sum_{k=0}^{m-1} \ell(x_u(k), u(k)) \right\}. \tag{2.16}$$

Here μ is an m-step feedback control law. It is assumed that the minimum with respect to u is attained. Since in practice we use a receding horizon controller, we now consider the finite horizon costs where $N \in \mathbb{N}$ is the prediction horizon

$$J_N(x_0, u) = \sum_{k=0}^{N-1} \ell(x_u(k), u(k)) \tag{2.17}$$

with the optimal value function

$$V_N(x_0) = \inf_{u \in \mathcal{U}} J_N(x_0, u) \tag{2.18}$$

Note that we are not using any terminal constraints or penalties in this formulation. Based on this finite horizon optimal control problem, we define a feedback control $\mu_{N,m}$ where $m \leq N$, by picking the first m elements of the optimal control sequence u^*.

$$\mu_{N,m}(x_0, k) = u^*(k), \quad k = 0, \ldots, m-1$$

In [37] an estimate of the suboptimality of the feedback control law $\mu_{N,m}$ for the infinite horizon problem is derived. For an m-step feedback control law μ and its corresponding trajectory x_μ we can define the optimal costs

$$V_\infty^\mu(x_0) := \sum_{k=0}^{\infty} \ell(x_\mu(k), \mu(x_\mu(k), k))$$

The estimate of the degree of suboptimality for control feedback $\mu_{N,m}$ is derived by getting an upper bound for V_∞^μ. Using this estimate, results on the asymptotic stability of the closed-loop system are derived using V_N as a Lyapunov function. The results derived therein rely on results on relaxed dynamic programming [58], adapted to the MPC problem.

Considering an m-step feedback law $\tilde{\mu}$ and its corresponding trajectory $x_{\tilde{\mu}}(k)$ with $x_{\tilde{\mu}}(0) = x_0$ and a value function $\tilde{V} : X \to \mathbb{R}^+$ which satisfies

$$\tilde{V}(x_0) \geq \tilde{V}(x_{\tilde{\mu}}(m)) + \alpha \sum_{k=0}^{m-1} \ell(x_{\tilde{\mu}}(k), \tilde{\mu}(x_0, k)) \tag{2.19}$$

for some $\alpha \in (0, 1]$ and $\forall x_0 \in X$. Then for all $x \in X$ the following holds

$$\alpha V_\infty(x) \leq \alpha V_\infty^{\tilde{\mu}}(x) \leq \tilde{V}(x)$$

Herein we will present the key propositions for the proof of asymptotic stability. For full proofs on these propositions see [37].

Asymptotic Controllability and Optimal Values

This subsection introduces an asymptotic controllability assumption followed by several consequences for the optimal control problem. The controllability assumption here is presented not in terms of the trajectory but in terms of the running costs along the trajectory.

Here we recall the definitions of $\mathcal{K}$ and $\mathcal{KL}$ class functions presented in Definitions A.2.1 and A.2.2 respectively. We also define $\ell^*(x) := \min_{u \in U} \ell(x, u)$.

A 5. *We begin by making the assumption that for a given function $\beta \in \mathcal{KL}_0$, for each $x_0 \in X$ there exists a control function $u_{x_0} \in \mathcal{U}$ which satisfies*

$$\ell(x_{u_{x_0}}(k), u_{x_0}(k)) \leq \beta(\ell^*(x_0), k)$$

for all $k \in \mathbb{N}$.

Some special cases for $\beta \in \mathcal{KL}_0$ include exponential controllability,

$$\beta(r, k) = C\sigma^k r \tag{2.20}$$

for constant $C \geq 1$ and $\sigma \in (0, 1)$. And finite time controllability

$$\beta(r, k) = c_k r \tag{2.21}$$

for some sequence $(c_k)_{k\in\mathbb{N}}$ with $c_k \geq 0$ and $c_k = 0$ for all $k \geq k_0$. We also define the following property for β

$$\beta(r, k + m) \leq \beta(\beta(r, k), m) \quad \forall\, r \geq 0, \quad k, m \in \mathbb{N} \tag{2.22}$$

In other works, [51] [60] β is obtained using a suitable Lyapunov function as we see in Appendix A where we review comparison functions. In [37], however, a precise function β is not identified, rather the properties of β are used to guarantee the performance of the closed-loop MPC algorithm. Under the assumption A5, for any $r \geq 0$ and $N \geq 1$ we define

$$B_N(r) := \sum_{k=0}^{N-1} \beta(r, k) \tag{2.23}$$

which produces the following inequality

$$V_N(x_0) \leq B_N(\ell^*(x_0)). \tag{2.24}$$

This is a direct consequence of A5 as

$$V_N(x_0) \leq J_N(x_0, u_{x_0}) = \sum_{k=0}^{N-1} \ell(x_{u_{x_0}}(k), u_{x_0}(k)) \leq \sum_{k=0}^{N-1} \beta(\ell^*(x_0), k) = B_N(\ell^*(x_0))$$

Using the fact that the final piece of an optimal trajectory is also an optimal trajectory, we use (2.24) to create a bound on the finite horizon functional along the optimal trajectory.

Under Assumption 5, for an $x_0 \in X$ and a finite horizon optimal control sequence u^* on some prediction horizon $N \geq 1$, the following holds for B_N from (2.23)

$$J_{N-n}(x_{u^*}(n), u^*(n) \leq B_{N-n}(\ell^*(x_{u^*}(n)) \tag{2.25}$$

for each $n \in \{0, 1, \ldots, N-1\}$.

Similarly, considering a two part trajectory, we can obtain an expression for V_N using (2.24). Under assumption A5 we consider an $x_0 \in X$ and a finite horizon optimal control sequence u^* with an optimization horizon $N \geq 1$. For each $m = 1, \ldots, N-1$ and each $j = 0, \ldots, N-m-1$ the following holds for B_N from (2.23).

$$V_N(x_{u^*}) \leq J_j(x_{u^*}(m), u^*(m) + B_{N-j}(\ell^*(x_{u^*}(m+j)) \tag{2.26}$$

Computation of Performance Bounds

Using the results from the previous section, the constructive approach to computing α in (2.19) in [37] is presented here. To begin, we introduce a sequence $\lambda_0, \ldots, \lambda_{N-1} > 0$ which corresponds to an optimal running cost sequence $\ell(x_{u^*}(k), u^*(k))$ and a value $v > 0$ corresponding to the optimal value function $V_N(x_{u*}(m))$.

Under A5 we consider an $x_0 \in X$ with the finite horizon optimal control sequence $u^* \in \mathcal{U}$ for a prediction horizon $N \geq 1$ and a control horizon $m \in \{1, \ldots, N-1\}$. Given that

$$\lambda_k := \ell(x_{u^*}(k), u^*(k)), \quad k = 0, \ldots, N-1$$

following from (2.25)

$$\sum_{k=n}^{N-1} \lambda_k \leq B_{N-n}(\lambda_n), \quad n = 0, \ldots, N-2 \tag{2.27}$$

holds. Furthermore, given that

$$v = V_N(x_{u^*}(m)),$$

following from (2.26),

$$v \leq \sum_{k=0}^{j-1} \lambda_{k+m} + B_{N-j}(\lambda_{j+m}), \quad j = 0, \ldots, N-m-1 \tag{2.28}$$

holds. Using the expressions in (2.25) and (2.26), we can present an expression for inequality (2.19). This gives sufficient conditions for the suboptimality of the finite horizon MPC

feedback law μ_{N-m}. Assuming that the sequence $\lambda_k > 0, k = 0 \ldots, N-1$ and value $v > 0$, fulfilling expressions (2.25) and (2.26) satisfy

$$\sum_{k=0}^{N-1} \lambda_k - v \geq \alpha \sum_{k=0}^{m-1} \lambda_k \tag{2.29}$$

for some $\alpha \in (0, 1]$. Then the m-step MPC control feedback law $\mu_{N,m}$, which is the solution to optimal control problem, (2.18) satisfying A5 meets the conditions for (2.19) to hold for all $x \in X$. Explicitly, inequality

$$\alpha V_\infty(x) \leq \alpha V_\infty^{\mu_{N.m}}(x) \leq V_N(x) \tag{2.30}$$

holds. Examining (2.29), we can interpret α as a performance bound, which indicates how well the receding horizon MPC approximates the infinite horizon problem. We now focus on building an expression to compute α. An optimization approach to computing α can now be presented. Given that $\beta \in \mathcal{KL}_0$, $N \geq 1$ and $m \in \{1, \ldots, N-1\}$

$$\alpha := \inf_{\lambda_0, \ldots \lambda_{N-1}, v} \frac{\sum_{n=0}^{N-1} \lambda_k - v}{\sum_{n=0}^{m-1} \lambda_k} \tag{2.31}$$

subject to constraints (2.25) and (2.26) and $\lambda_0, \ldots \lambda_{N-1}, v > 0$.

For the remainder of this section, we follow the analysis from [39] to find an explicit expression for α. Therein, the analysis is done to a more general case where an additional weight w on the final term is included, described in [37, Remark 4.3]. The cost function (2.17) becomes

$$J_N^w(x_0, u) = \sum_{k=0}^{N-2} \ell(x_u(k), u(k)) + w\ell(x_u(N-1), u(N-1))$$

for some $w \geq 1$. Note that for the presentation herein, we will stick to the simpler case where $w = 1$ and we recover the original cost function in (2.17), in other words $J_N^1 = J_N$. To obtain an expression for α, we look at optimization problem (2.31), which is still in general a non-linear optimization problem. However, if we consider functions $\beta(r, n)$ which are linear in r, problem (2.31) becomes a linear program. If $\beta(r, t)$ is linear in r, then problem (2.31) as shown in [39] yields the same optimal value α as

$$\alpha := \min_{\lambda_0, \ldots \lambda_{N-1}, v} \sum_{k=0}^{N-1} \lambda_k - v \tag{2.32}$$

subject to constraints (2.25) and (2.26) which are now linear and

$$\lambda_0, \dots \lambda_{N-1}, v \geq 0, \quad \sum_{k=0}^{N-1} \lambda_k = 1.$$

We can now show that the optimal value for problem (2.31) can be found from the following. We first define a new variable $\gamma_k = B_k(r)/r$. Let $\beta(\cdot,\cdot)$ be linear in its first argument and satisfy property (2.22). The optimal value α of (2.31) for a given optimization horizon $N \geq 1$ and control horizon $m \in \{0, \dots, N-1\}$ satisfies $\alpha = 1$ if and only if $\gamma_{m+1} \leq 1$, otherwise

$$\alpha = 1 - \frac{(\gamma_{m+1} - 1) \prod_{i=m+2}^{N} (\gamma_i - 1) \prod_{i=N-m+1}^{N} (\gamma_i - 1)}{\left(\prod_{i=m+1}^{N} \gamma_i - (\gamma_{m+1} - 1) \prod_{i=m+2}^{N} (\gamma_i - 1) \right) \left(\prod_{i=N-m+1}^{N} \gamma_i - \prod_{i=N-m+1}^{N} (\gamma_i - 1) \right)} \tag{2.33}$$

Asymptotic Stability

Using the value of the performance bound α, conclusions can be drawn about the asymptotic stability of the closed-loop MPC algorithm. Following the method in [37], the following assumption is made.

A 6. *There exists a closed set $X_f \subset X$ satisfying:*

i) For each $x \in X_f$ there exists $u \in U$ with $f(x,u) \in X_f$ and $\ell(x,u) = 0$; i.e. there exists a state and input in X_f at which the system can stay at no cost, in the usual formulation this is at the desired state and with zero input.

ii) There exists $\mathcal{K}_\infty$ functions α_1, α_2 such that inequality

$$\alpha_1(\|x\|_{X_f}) \leq \ell^*(x) \leq \alpha_2(\|x\|_{X_f})$$

holds for each $x \in X$ where $\|x\|_{X_f} := \min_{y \in X_f} \|x - y\|$.

This assumption assures the global asymptotic stability of X_f under optimal feedback for the infinite horizon problem, provided $\beta(r,k)$ is summable. The assumptions presented build on A1 - A4 used in the previous section. Following these assumptions the main stability result can be presented.

Theorem 2.1. *We consider an optimal control problem (2.18), where the assumptions 5 and 6 hold. Then if $\beta \in \mathcal{KL}_0$, prediction horizon $N \geq 1$, control horizon $m \in \{1, \ldots, N-1\}$ and that $\alpha \in (0, 1]$ as obtained from (2.33). We can say that the MPC feedback law $\mu_{N,m}$ asymptotically stabilizes the set X_f and that V_N is a corresponding m-step Lyapunov function in the sense that*

$$V_N(x_{\mu_{N,m}}(m)) \leq V_N(x) - \alpha V_m(x). \tag{2.34}$$

This expression guarantees that V_N decreases with every m-step control input, [37] and it also decreases to 0 as $k \to \infty$.

The presented theorem, gives a conservative criterion for stability. It is possible for a system satisfying Assumptions 5 and 6 but for which $\alpha < 0$ to still be asymptotically stable. We can however say that for any system which violates the conditions in Theorem 2.2.2, we can always find a problem which is not stabilized by the MPC feedback law. This is expressed and proved in the [37][Theorem 5.3].

Asymptotic Stability for Control Horizon, $m = 1$

In later chapters, we use a simplified version of Theorem 2.1, where we only consider a control horizon of $m = 1$. The simplified theorem which we use to conclude the asymptotic stability of discrete time MPC schemes for a system with dynamics (2.2) without terminal conditions is presented below.

Theorem 2.2. *Let a monotonically increasing and bounded sequence $(\gamma_i)_{i\in\mathbb{N}}$ bound the value function V_i over a horizon i such that the inequality*

$$V_i(x_0) \leq \gamma_i \cdot \ell^*(x_0) \qquad \forall\, i \in \mathbb{N} \text{ and } x_0 \in X \tag{2.35}$$

holds. Additionally, let the performance index α_N for the prediction horizon N, be given by

$$\alpha_N := 1 - \frac{(\gamma_N - 1)\prod_{k=2}^{N}(\gamma_k - 1)}{\prod_{k=2}^{N}\gamma_k - \prod_{k=2}^{N}(\gamma_k - 1)}. \tag{2.36}$$

Then, if $\alpha_N > 0$ holds, the relaxed Lyapunov inequality

$$V_N(f(x, \mu_N(x))) \leq V_N(x) - \alpha_N \ell_\tau(x, \mu_N(x)) \tag{2.37}$$

holds for all $x \in X$ and the MPC closed-loop with prediction horizon N is asymptotically stable. Moreover, the performance bound

$$V_\infty^{\mu_N}(x) := \sum_{k=0}^{\infty} \ell_\tau(x_{\mu_N}(k,x), \mu_N(k,x)) \leq \alpha_N^{-1} V_\infty(x) \tag{2.38}$$

is guaranteed, where $\mu_N(k,x) := \mathbf{u}^\star(0, x_{\mu_N}(k,x))$ and $V_\infty^{\mu_N}(x)$ are the MPC closed-loop control and closed-loop costs on the infinite horizon, respectively. □

One of the key challenges in later chapters is to calculate the bounded sequence γ_i which allows us to calculate the performance bound α_N from which we can confirm asymptotic stability.

2.2.3 Stability of MPC Without Terminal Constraints or Penalty in Continuous Time

Without going through the same level of detail as presented in Section 2.2.2, in this section we present a theorem analogous to Theorem 2.1 in continuous time. The asymptotic stability of system (2.1) under the employed MPC scheme, without terminal conditions, can be summarized by the following theorem, see [79, Theorem 9] for details and the proof.

Theorem 2.3. *For the dynamics given by (2.1) and the running costs given by (4.8), let a given growth bound $B : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ be a monotonically increasing and bounded such that the estimate*

$$V_t(x_0) \leq B(t) \cdot \ell^*(x_0) \tag{2.39}$$

holds for the value function V_t over the horizon t for all $x_0 \in X_f$ and $t \geq 0$, where

$$\ell^\star(x_0) := \inf_{u \in \mathcal{U}^1(x_0)} \ell(x_0, u).$$

Furthermore, let there exist $\mathcal{K}_\infty$-functions $\underline{\eta}$, $\bar{\eta}$ satisfying

$$\underline{\eta}(\|x - x^\star\|) \leq \ell^\star(x) \leq \bar{\eta}(\|x - x^\star\|) \qquad \forall\, x \in X. \tag{2.40}$$

Additionally, for a given control horizon $\delta > 0$ and prediction horizon $T > \delta$ let the performance index $\alpha_{T,\delta}$ be given by

$$\alpha_{T,\delta} := 1 - \frac{e^{-\int_\delta^T B(t)^{-1}\,dt} \cdot e^{-\int_{T-\delta}^T B(t)^{-1}\,dt}}{\left[1 - e^{-\int_\delta^T B(t)^{-1}\,dt}\right]\left[1 - e^{-\int_{T-\delta}^T B(t)^{-1}\,dt}\right]}, \tag{2.41}$$

then, if $\alpha_{T,\delta} > 0$ holds, the relaxed Lyapunov inequality

$$V_T(x_\mu(t;x)) \leq V_T(x) - \alpha_{T,\delta} \int_0^\delta \ell(x_\mu(t;x), \mu(t;x))\, dt \tag{2.42}$$

holds for all $x \in X$ and the MPC closed-loop with control horizon δ and prediction horizon T is asymptotically stable. □

Inequality (2.39) relates the growth of the MPC value function and the running costs w.r.t. the initial condition x_0; the verification of this Inequality is referred to as cost controllability verification (a term coined in the study [21]), which is a sufficient condition for asymptotic stability.

2.3 Kinematics of Holonomic Robots

Chapters 3, 4 and 5 deal with the control of a holonomic mobile robot. Each of these chapters use a kinematic model of the robot for MPC, the various forms of the kinematic model used throughout this thesis are presented in this section.

2.3.1 Higher Level Kinematic Model

The holonomic robot state is described by the state vector $\mathbf{x} = (x_1, x_2, x_3)^\top \in X \subsetneq \mathbb{R}^3$, which consists of the (spatial) position $(x_1, x_2)^\top$ [m] and the orientation angle x_3 [rad] of the robot. Additionally, the robot control input is defined by the vector $\mathbf{u} = (u_1, u_2, u_3)^\top \in U \subsetneq \mathbb{R}^3$, where u_1 [m/s], u_2 [m/s], and u_3 [rad/s] are the linear, the lateral, and the angular speeds of the robot, respectively. At time $t \in \mathbb{R}_{\geq 0}$ [seconds], the continuous time kinematic model of a holonomic mobile robot is given by

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)) = \mathcal{R}(x_3(t)) \cdot \mathbf{u}(t), \tag{2.43}$$

where $f : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}^3$ is an analytic mapping and $\mathcal{R}(x_3) \in \mathbb{R}^{3\times 3}$ is the following matrix

$$\mathcal{R}(x_3) = \begin{pmatrix} \cos(x_3) & -\sin(x_3) & 0 \\ \sin(x_3) & \cos(x_3) & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{2.44}$$

2.3.2 Acceleration Control Kinematic Model

When considering the higher level kinematics of a holonomic robot, using velocity control defined in (2.43) is generally sufficient. However, in cases such as the acceleration constrained MPC controller presented in Chapter 4, an acceleration controlled model may be more useful. We can obtain this model by modifying (2.43) to give

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}, \mathbf{u}) = \begin{pmatrix} \mathbf{0}_{3\times3} & \mathcal{R}(x_3(t)) \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \end{pmatrix} \mathbf{x} + \begin{pmatrix} \mathbf{0}_{3\times3} \\ \mathbf{I}_{3\times3} \end{pmatrix} \mathbf{u}, \tag{2.45}$$

where $\mathcal{R}(x_3)$ is matrix (2.44). Here, the state of the robot is described by the vector $\mathbf{x} = (x_1, x_2, x_3, x_4, x_5, x_6)^\top \in \mathbb{R}^6$, which consists of the (spatial) position $(x_1, x_2)^\top$ [m], the orientation of the robot x_3 [rad], and the robot speeds x_4 [m/s], x_5 [m/s], and x_6 [rad/s]. Additionally, the robot control input is defined by the vector $\mathbf{u} = (u_1, u_2, u_3)^\top \in U \subsetneq \mathbb{R}^3$, where u_1 [m/s^2], u_2 [m/s^2], and u_3 [rad/s^2] are the linear, the lateral, and the angular accelerations of the robot, respectively.

2.3.3 Lower Level Kinematics

Though robots will often accept a control $\mathbf{u}$ which is based on the higher level kinematics, this is not always the case and understanding the lower level kinematics is still important. It also allows as seen in Chapter 3 to add constraints on wheel speed to ensure we do not exceed the motor saturation limits. We consider a holonomic robot with design parameters shown in Fig. 2.1. The lower level kinematics model relates the robot's speeds $\mathbf{u}$ and the robot wheels' speeds, $\mathbf{u}_w \in \mathbb{R}^4$ for robots with mecanum wheels. This relation is given by:

$$\mathbf{u}_w(t) = \frac{1}{r} \cdot H\mathbf{u}(t), \tag{2.46}$$

where

$$H = \begin{pmatrix} 1 & -1 & -a-b \\ 1 & 1 & a+b \\ 1 & -1 & a+b \\ 1 & 1 & -a-b \end{pmatrix}$$

for mecanum wheels robots. Here, r [m] is the radius of the robot wheels, and the dimensions a and b [m] are shown in Fig. 1.2, see Appendix B for the derivation of the matrix H

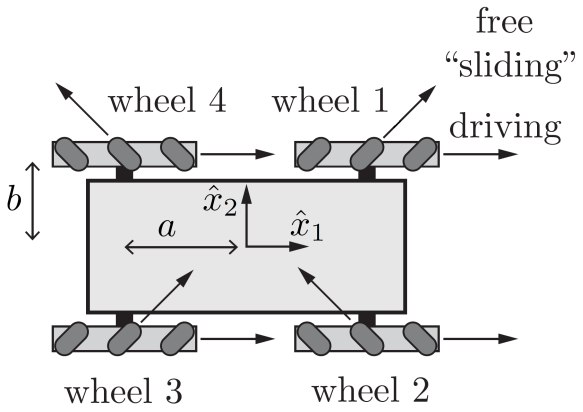

Figure 2.1: Holonomic mobile robot with mecanum wheels. The driving and the free sliding directions for each wheel type is shown by corresponding vectors. The figure is reprinted from [59].

2.3.4 Discrete Time Kinematics

Since in practice controllers are typically applied in a zero-order hold fashion, we consider the discrete time formulation of the holonomic robot kinematics. Here, model (2.43) is discretized by the sampling period $\tau > 0$, and assuming piecewise constant control during each sampling interval; thus, the resulting (*exact*) discrete-time model is written as

$$\mathbf{x}^+ = f_{e,\tau}(\mathbf{x}, \mathbf{u}) = \mathbf{x} + \mathcal{R}_e(\tau, x_3, u_3)\mathbf{u}, \tag{2.47}$$

where the superscript $(\cdot)^+$ denotes the next state and

$$\mathcal{R}_e(\tau, x_3, u_3) = \begin{pmatrix} \frac{\sin(x_3+\tau u_3)-\sin(x_3)}{u_3} & \frac{\cos(x_3+\tau u_3)-\cos(x_3)}{u_3} & 0 \\ \frac{\cos(x_3)-\cos(x_3+\tau u_3)}{u_3} & \frac{\sin(x_3+\tau u_3)-\sin(x_3)}{u_3} & 0 \\ 0 & 0 & \tau \end{pmatrix}. \tag{2.48}$$

We remark that for $u_3 = 0$ the model given by Eq. (2.47) reduces to

$$\mathbf{x}^+ = \mathbf{x} + \lim_{u_3 \to 0} \mathcal{R}_e(\tau, x_3, u_3)\mathbf{u} = \mathbf{x} + \tau \begin{pmatrix} u_1 \cos(x_3) - u_2 \sin(x_3) \\ u_1 \sin(x_3) + u_2 \cos(x_3) \\ 0 \end{pmatrix}.$$

2.4 Path Following Problem

The point stabilization problem can be extended to a trajectory tracking problem by making the desired reference point time dependent. It has been shown in [2, 3] that the trajectory tracking method has limitations which prevent it from achieving perfect tracking due to its unstable zero dynamics. In contrast to trajectory tracking, the path-following problem has an objective to follow a geometric path without a predefined speed assignment [2]. With the additional degree of freedom provided by the ability to select a timing law, the path following problem does not suffer from the same limitations as the trajectory tracking problem and the deviation from the path can be made arbitrarily small.

A common way of formulating the path following problem is by parametrizing the desired reference path with an additional (virtual) variable, referred to as the path parameter. This formulation was introduced in the form of maneuver regulation in [41]. Therein, a maneuver is defined as a curve in the state space that is consistent with system dynamics. A feedback control law is developed as a trajectory tracking problem, using a projection mapping which selects an appropriate trajectory timing for the maneuver; then, the maneuver regulation control law is obtained. This idea is applied to flight control in [4], where a Linear Quadratic Regulator (LQR) is used to control the nonminimum phase part of the system.

Another approach for path following based on nonlinear feedback control is proposed in [85] for mobile robots to follow a path parametrized by curvilinear coordinates. This method is extended to systems with large disturbances such as marine vehicles [28]. Building upon the previously highlighted work, a general approach for robust output maneuvering is presented in [88]. Therein, a path-following algorithm is divided into two tasks: first to force the output of the system to follow a geometric path, and second to satisfy the dynamic task of following the path with a given timing for velocity or acceleration. Further advancements in path-following algorithms are made in [72, 73]. Here, the path-following control problem for a magnetic levitation positioning system is divided into two parts: a transversal control which stabilizes the output to the path-following manifold (the set of geometric trajectories which satisfy the path following problem) and a tangential control of the motion along the desired path.

In the following section we present the path parametrization formulation of the path following which is used to set up an MPC controller in Chapter 5.

2.4.1 Problem Formulation

The path following problem requires steering system (2.47) along a geometric reference path $\mathcal{P} \in X$. Here, the negative parameterization $\mathbf{p} : [\bar{\theta}, 0] \to \mathbb{R}^3$ with $\mathbf{p}(0) = 0$ is used to define $\mathcal{P}$, i.e.

$$\mathcal{P} = \left\{ \mathbf{x} \in \mathbb{R}^3 | \exists \theta \in [\bar{\theta}, 0] : \mathbf{p}(\theta) = \mathbf{x} \right\}.$$

The scalar variable $\theta \in [\bar{\theta}, 0]$ is referred to as the path parameter, we note that $\theta \leq 0$ and the end of the path is reached at $\theta = 0$. For holonomic mobile robots, we can choose the path $\mathbf{p}(\theta)$ as three independent reference curves, one for each state in $\mathbf{x}$, as

$$\mathbf{p}(\theta) = \begin{pmatrix} p_1(\theta) & p_2(\theta) & p_3(\theta) \end{pmatrix}^\top, \tag{2.49}$$

where p_1, p_2 and p_3 are piece-wise continuous and differentiable functions of the path parameter θ. In path following, the path parameter θ is not given an explicit time dependency. However, the path parameter θ is treated as a *virtual* state and is given the dynamics $\dot{\theta} = v$, which when discretized, becomes

$$\theta^+ = \theta + \tau v, \tag{2.50}$$

where the control $v \in \mathcal{V} := [0, \bar{v}]$ governs the time evolution of θ and is treated as an extra degree of freedom for the controller. The chosen dynamics of the path parameter is a single integrator in the discrete time; we refer to [31, Chapter 4] for more details. The controls $\mathbf{u}$ and v can now be chosen such that the path $\mathcal{P}$ is followed as closely as possible. We note here that v is non-negative, ensuring forward motion along the path $\mathcal{P}$. The path following problem can be summarized, as described in [35], as

Problem 1. *(State-Space Path following)*

1. *Convergence to path: The robot state* $\mathbf{x}$ *converges to the path* $\mathcal{P}$ *such that*

$$\lim_{k \to \infty} ||\mathbf{x}(k) - \mathbf{p}(\theta(k))|| = 0. \tag{2.51}$$

2. *Convergence to end of path: The robot moves in the direction of increasing* θ *along* $\mathcal{P}$ *such that* $v \geq 0$ *and* $\lim_{k \to \infty} \theta(k) = 0$.

3. *Constraint satisfaction: The state and control constraints* X *and* U *are satisfied for all time steps.*

Using the path dynamics in Equation (2.50), we treat θ as a virtual state which we can add to the robot kinematics described in Equation (2.47). The path following problem can now be analyzed by the following augmented system [33]

$$\mathbf{z}^+ = f_{aug}(\mathbf{z}, \mathbf{w}) = \mathbf{z} + \mathcal{R}_{aug}(\tau, x_3, u_3)\mathbf{w}, \tag{2.52}$$

where

$$\mathbf{z} := \begin{pmatrix} \mathbf{x} \\ \theta \end{pmatrix}, \quad \mathbf{w} := \begin{pmatrix} \mathbf{u} \\ v \end{pmatrix}. \tag{2.53}$$

$\mathbf{z} \in Z$ is the augmented system state vector, which includes the state of the robot $\mathbf{x}$ and the virtual state θ. $\mathbf{w} \in W$ is the augmented control vector, which includes the robot control input $\mathbf{u}$ and the virtual control v. Here, the sets Z and W for the augmented system (2.52) are defined as

$$Z := X \times [\bar{\theta}, 0], \qquad \text{and} \quad W := U \times \mathcal{V}.$$

Moreover, $\mathcal{R}_{aug}(\cdot)$ is given by

$$\mathcal{R}_{aug}(\tau, x_3, u_3) = \begin{pmatrix} \mathcal{R}_e(\tau, x_3, u_3) & 0 \\ 0 & \tau \end{pmatrix}. \tag{2.54}$$

It is now straightforward to set up the path-following control problem as the point stabilization of the augmented system (2.52), see [33].

For the augmented system (2.52), we use $z_w(\cdot; z_0)$ to denote a trajectory emanating from $\mathbf{z}_0$ and driven by the control sequence

$$w = (\mathbf{w}(0), \mathbf{w}(1), \ldots, \mathbf{w}(N-1)) \in W^N, \qquad N \in \mathbb{N}$$

Moreover, a control sequence w is said to be admissible, denoted by $w \in \mathcal{W}^N(\mathbf{z}_0)$ if, for a given state $\mathbf{z}_0 \in Z$, the state trajectory

$$z_w(\cdot; \mathbf{z}_0) = (z_w(0; \mathbf{z}_0), z_w(1; \mathbf{z}_0), \ldots, z_w(N; \mathbf{z}_0))$$

generated from the system dynamics (2.52), where $z_w(0; \mathbf{z}_0) = \mathbf{z}_0$, satisfies $z_w(k; \mathbf{z}_0) \in Z$ for all $k \in \{0, 1, \ldots, N\}$.

Chapter 3

MPC without Terminal Constraints or Cost for Holonomic Mobile Robots

This chapter is an excerpt from Model predictive control without terminal constraints or costs for holonomic mobile robot [67], which was published in the journal of Robotics and Autonomous Systems. This excerpt though not exclusively authored by myself is meant to give context and to represent my contribution to the publication. These contributions include the adaptation of the stability proof for a holonomic robot and the development of growth bounds in discrete time. I am also responsible for developing the algorithms for and conducting simulations and experiments then analyzing the obtained results.

3.1 Introduction

In this chapter, we focus on the point-stabilization problem of holonomic mobile robot problem using MPC *without stabilizing constraints or costs*. Although several studies employed MPC for the control of holonomic mobile robots, only a few considered the closed-loop asymptotic stability of the system under MPC. In all of these studies, stabilizing terminal conditions were adopted, see, e.g. [7, 46, 48]. MPC schemes *without* stabilizing constraints are easier to design, require less computational effort in implementation, and are the preferred MPC variant in industrial applications when compared to the MPC schemes with stabilizing constraints. Moreover, for a fixed prediction horizon, MPC schemes without stabilizing constraints provide larger stability regions than schemes with stabilizing constraints. Finally, when the control effort in the MPC cost function is largely penalized,

schemes without stabilizing conditions lead to better closed-loop performance in contrast to schemes with stabilizing constraints; we refer to [38, Chapter 8] for a more thorough comparison of the variant MPC schemes.

In this chapter, the closed-loop asymptotic stability of MPC without stabilizing constraints or costs is analyzed for the regulation (or point stabilization) control of holonomic mobile robots. In particular, we verify cost controllability, i.e. a sufficient stability condition, which relates the growth of the MPC value function and the running costs uniformly w.r.t. the initial state; for details on the general framework, see [21]. This is achieved by deriving a growth function bounding the MPC value function. This growth function allows to precisely determine the length of the prediction horizon such that asymptotic stability of the desired set-point w.r.t. the MPC closed-loop is guaranteed and, in addition, a suboptimality estimate on the infinite horizon can be determined. The latter provides a relative measure on the holonomic robot's performance compared to the optimal performance provided from the infinite horizon case.

Similar to the work of Worthmann et al. [96], an open-loop control maneuver, which stabilizes the considered system to a reference set point is designed and then used to derive a growth function of the MPC value function and calculate a stabilizing prediction horizon length. The special structure of the resulting growth function is used to show that the point stabilization of holonomic mobile robots can be guaranteed to be asymptotically stable, under the employed MPC with the shortest prediction horizon length. Additionally, we derive an estimate on the MPC performance (suboptimality) index in this case. The analysis is performed in the discrete-time settings. We verify the theoretical results by numerical and real-time lab experiments.

The author of this thesis can claim at least in part the following contributions which are presented in this chapter: 1) the adaptation from continuous time to discrete time of a growth function for the MPC value function used for the point stabilization of holonomic mobile robots. 2) employing the derived growth function to rigorously prove that the shortest possible prediction horizon is sufficient to ensure closed-loop asymptotic stability using the employed MPC scheme. 3) verifying the theoretical results throughout numerical simulations and experiments.

This chapter is organized as follows: in Section 3.2, the kinematics model of holonomic mobile robots is presented. In Section 3.3, the proposed control scheme is introduced. Stability results are investigated in the discrete-time setting in Section 3.4, where it is shown that closed-loop asymptotic stability is guaranteed by the shortest prediction horizon, additinally the performance bound of MPC is computed. Simulation and experimental results employing a mecanum wheel holonomic robot are shown in Section 3.5. Finally,

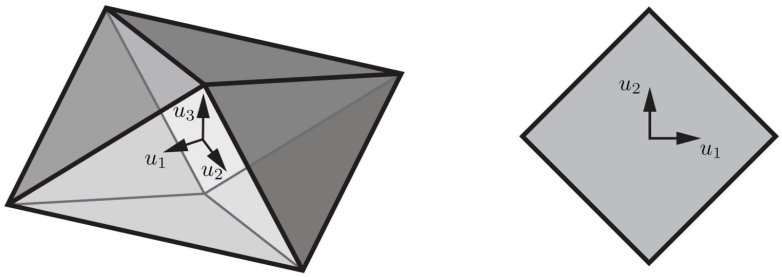

Figure 3.1: Visualization of the set U for mecanum wheels robots. Translation only constraints, for $u_3 = 0$, are shown on the right. The figure is reprinted from [59].

concluding remarks and the outlook of the contribution are discussed in Section 3.6.

3.2 State and Control Input Constraint Sets

Let us start by considering the kinematic model for a holonomic robot in (2.43). The compact and convex state constraint set X is defined as

$$X := [-\bar{x}_1, \bar{x}_1] \times [-\bar{x}_2, \bar{x}_2] \times S,$$

for $\bar{x}_1 > 0$ and $\bar{x}_2 > 0$, with the origin in its interior. S is the compact set of angles on circle.

In order to derive the control input set U, we must also consider the lower level kinematics (2.46). Now, for a given speed limit on the wheels, $\bar{\mathbf{u}}_w$, the control input set U can be defined as

$$U := \left\{ \mathbf{u} \in \mathbb{R}^3 |\ \ \|H\mathbf{u}\|_\infty \leq r\bar{\mathbf{u}}_w \right\}. \tag{3.1}$$

For robots with mecanum wheels, we have

$$\|H\mathbf{u}\|_\infty := |u_1| + |u_2| + (a+b)|u_3|.$$

The reason $\|H\mathbf{u}\|_\infty$ reduces to one term for robots with mecanum wheels is the symmetry of the robot configuration around $\hat{x}_1$-axis and $\hat{x}_2$-axis, see Fig. 1.2. The set U is convex with the origin in its interior, see Fig. 3.1. In summary, we have $(0,0,0)^\top \times (0,0,0)^\top \in \text{int}(X) \times \text{int}(U)$.

3.3 Model Predictive Control in Discrete Time

In this section, we present the discrete time formulation of the MPC Algorithm from Section 2.1.2 applied to the position regulation of a holonomic robot. As the proposed MPC controller is applied in a zero-order hold fashion, we consider the discrete time formulation of this problem. Going forward, we therefore consider the discrete time model of the holonomic robot kinematics in (2.47).

A control sequence $u = (\mathbf{u}(0), \mathbf{u}(1), \ldots, \mathbf{u}(N-1)) \in U^N$ of length $N \in \mathbb{N}$ is said to be admissible, for state $\mathbf{x}_0$, denoted by $u \in \mathcal{U}^N(\mathbf{x}_0)$ if the state trajectory

$$x_u(\cdot; \mathbf{x}_0) = (x_u(0; \mathbf{x}_0), x_u(1; \mathbf{x}_0), \ldots, x_u(N; \mathbf{x}_0)),$$

generated iteratively by the model (2.47) and stemming from $\mathbf{x}_0 =: x_u(0, \mathbf{x}_0)$, satisfies $x_u(k, \mathbf{x}_0) \in X$ for all $k \in \{0, 1, \ldots, N\}$.

Now, we present the MPC Algorithm in discrete-time. Here, the prediction horizon and the control horizon are defined by N and m. Moreover, we define the discrete-time running costs $\ell_\tau(\mathbf{x}, \mathbf{u}) : X \times U \to \mathbb{R}_{\geq 0}$ by

$$\ell_\tau(\mathbf{x}, \mathbf{u}) = \|\mathbf{x}\|_Q^2 + \|\mathbf{u}\|_R^2 \tag{3.2}$$

As a result, at a sampling step $k \in \mathbb{N}_0$, the discrete-time cost function $J_N : X \times U^N \to \mathbb{R}_{\geq 0}$ and the corresponding value function $V_N : X \to \mathbb{R}_{\geq 0}$ are

$$J_N(\mathbf{x}_k, u) := \sum_{n=k}^{k+N-1} \ell_\tau(x_u(n, \mathbf{x}_k), \mathbf{u}(n)) \tag{3.3}$$

$$V_N(\mathbf{x}_k) := \inf_{u \in \mathcal{U}^N(\mathbf{x}_k)} J_N(\mathbf{x}_k, u) \tag{3.4}$$

$$\begin{aligned} \textbf{s.t.} \quad & \mathbf{x}(k) = \mathbf{x}_k, \\ & \mathbf{x}(i+1) = f(\mathbf{x}(i), \mathbf{u}(i)) \quad \forall i \in [k, k+N) \\ & \mathbf{x}(i) \in X \quad \forall i \in [k, k+N] \\ & \mathbf{u}(i) \in U \quad \forall i \in [k, k+N) \end{aligned}$$

for $N \in \mathbb{N} \cup \{\infty\}$. For the further analysis in this section, we set $m = 1$. Thus, at the sampling step $k \in \mathbb{N}_0$, the MPC feedback law in the discrete-time setting, denoted by $\mu_N : X \to U$, is given by $\mu_N(k, \mathbf{x}_k) = \mathbf{u}^\star(0, \mathbf{x}_k)$, where $u^\star = u^\star(\cdot, \mathbf{x}_k) \in \mathcal{U}^N(\mathbf{x}_k)$ satisfies $V_N(\mathbf{x}_k) = J_N(\mathbf{x}_k, u^\star)$.

3.4 Stability Results in Discrete Time

Here we prove that the closed-loop asymptotic stability of the considered holonomic mobile robot is guaranteed for the shortest prediction horizon length. We achieve this by referring to Theorem 2.2 and developing a sequence γ_i which bounds the value function and for which α_N calculated from (2.36) is contained in interval (0,1). The system dynamics considered are that of the holonomic robot (2.47) starting from some $\mathbf{x}_0 \in X$.

We note that the required assumption on the boundedness of $\ell_\tau(\mathbf{x}, \mathbf{u})$ for Theorem 2.2 holds for the chosen running costs.

3.4.1 Growth Bounds γ_i

The growth bound γ_i in Inequality (2.35) can be obtained using $\gamma_i = \sum_{j=0}^{i-1} c_j$, $i \in \mathbb{N}_{\geq 2}$, where the sequence $c_j \geq 0$, $j \in \mathbb{N}_0$, is summable, i.e. $\sum_{j=0}^{\infty} c_j < \infty$. c_j is calculated such that the inequality

$$\frac{\ell_\tau(x_{u_{\mathbf{x}_0}}(j; \mathbf{x}_0), \mathbf{u}_{\mathbf{x}_0}(j))}{\|\mathbf{x}_0\|_Q^2} \leq c_j, \quad \forall j \in \mathbb{N}_0 \text{ and } \mathbf{x}_0 \in X \tag{3.5}$$

holds for an admissible sequence of control actions $u_{\mathbf{x}_0} = \mathbf{u}_{\mathbf{x}_0}(j), j \in \mathbb{N}_0$, see, e.g. [96] for more details. The following proposition summarizes the procedure of computing c_j in Inequality (3.5) and, thus, the growth bound γ_i in Theorem 2.2.

Proposition 1. *Given positive definite diagonal weighting matrices $Q = diag(q_1, q_2, q_3)$ and $R = diag(r_1, r_2, r_3)$ in Eq. (3.2). Let the following assumptions*

$$r_1 = r_2, \quad q_1 = q_2, \quad r_1 \leq \lambda q_1/2, \text{ and } r_3 \leq \lambda q_3, \tag{3.6}$$

hold for some constant λ. Then, there exists a sufficiently long maneuver length k_m calculated from the maneuver time in Eq. (3.8) such that cost controllability, i.e. Inequality (2.35), holds with $\gamma_i = \sum_{j=0}^{i-1} c_j$, $i \in \mathbb{N}_{\geq 2}$, where c_j is defined by

$$c_j := \left[\left(\frac{k_m - j}{k_m} \right)^2 + \frac{\lambda}{\tau^2 k_m^2} \right] \tag{3.7}$$

for $j \in \{0, 1, \ldots, k_m - 1\}$. □

Proof. We choose the following straight trajectories for each initial state $\mathbf{x}_0 = (x_{0,1}, x_{0,2}, x_{0,3})^\top$ in X to the origin:

$$\mathbf{x}(j) = \left(\frac{k_m - j}{k_m}\right)\mathbf{x}_0, \quad \mathbf{x}_0 := \mathbf{x}(0), \quad \forall j \in \{0, \dots, k_m - 1\},$$

where $k_m \in \mathbb{N}_{\geq 2}$ is chosen sufficiently large such that the constraints U given by Eq. (3.1) are satisfied for all $\mathbf{x}_0 \in X$.

If we consider a maneuver time, t_m lower bounded by $\bar{t}_m$ corresponding to the initial condition $\mathbf{x}_0 = (\bar{x}_1, \bar{x}_2, \pi)^\top$, i.e. the farthest initial condition in the state set X from the origin $\mathbf{x}^\star = (0, 0, 0)^\top$. $\bar{t}_m$ can be calculated via the following formula

$$\bar{t}_m = \frac{\max_{x_3(t)\in[0,\pi]} \|t_m \cdot H\mathbf{u}(t)\|_\infty}{r \cdot \bar{\mathbf{u}}_w}, \tag{3.8}$$

where $\mathbf{u}(t)$ is given by Eq. (3.9). For simplicity, maximizing the numerator of Eq. (3.8) can be performed via a line-search. In the special case of $\bar{x}_1 = \bar{x}_2$, $\bar{t}_m$ reduces to

$$\bar{t}_m = \frac{\bar{x}_1 + \bar{x}_2 + \pi \cdot (a + b)}{r \cdot \bar{\mathbf{u}}_w},$$

for a holonomic robot with mecanum wheels. Choosing the maneuver length k_m such that $k_m \geq \lceil \bar{t}_m/\tau \rceil$ will satisfy the control constraints for any initial condition $\mathbf{x}_0 \in X$. The resulting open-loop control actions required to achieve the maneuver can be calculated by

$$\mathbf{u}_{\mathbf{x}_0}(j) = \mathcal{R}_e(\tau, x_3(j), u_3(j))^{-1}\left(-\frac{1}{k_m}\mathbf{x}_0\right), \tag{3.9}$$

for all $j \in \{0, \dots, k_m - 1\}$, where

$$\mathcal{R}_e(\tau, x_3, u_3)^{-1} = \frac{u_3}{4}\begin{pmatrix} \frac{\sin(x_3+\tau u_3)-\sin(x_3)}{\sin^2(0.5\tau u_3)} & \frac{\cos(x_3+\tau u_3)-\cos(x_3)}{\sin^2(0.5\tau u_3)} & 0 \\ \frac{\cos(x_3)-\cos(x_3+\tau u_3)}{\sin^2(0.5\tau u_3)} & \frac{\sin(x_3+\tau u_3)-\sin(x_3)}{\sin^2(0.5\tau u_3)} & 0 \\ 0 & 0 & \frac{4}{\tau u_3} \end{pmatrix};$$

moreover, later in this proof, we drop the subscript $\mathbf{x}_0$ from all the elements of $\mathbf{u}_{\mathbf{x}_0}(j)$ to keep the presentation technically simple. By inspecting Eq. (3.9), it is straightforward to obtain $u_3(j)$ as

$$u_3(j) = -\frac{x_{3,0}}{\tau k_m}, \ \forall j \in \{0, \dots, k_m - 1\}.$$

Now, applying the open-loop control actions defined by Eq. (3.9) and using the first assumption in (3.6) on the weighting matrix R, lead to the following running costs along the resulting open-loop trajectories

$$\ell_\tau(x_{u_{\mathbf{x}_0}}(j;\mathbf{x}_0),\mathbf{u}_{\mathbf{x}_0}(j)) = \left(\frac{k_m-j}{k_m}\right)^2 \sum_{i=1}^{3} q_i x_{0,i}^2 + r_1\left(u_1(j)^2+u_2(j)^2\right) + r_3\frac{x_{0,3}^2}{\tau^2 k_m^2},$$

where the term $r_1\left(u_1(j)^2+u_2(j)^2\right)$ can be calculated by squaring then adding the first two equations in (3.9); thus, we have

$$r_1\left(u_1(j)^2+u_2(j)^2\right) = \frac{r_1 u_3(j)^2}{4\sin^2(0.5\tau u_3(j))}\left(\frac{x_{0,1}^2}{k_m^2}+\frac{x_{0,2}^2}{k_m^2}\right).$$

Using the Taylor series expansion of $\sin^2(0.5\tau u_3(j))$, we have

$$\sin^2(0.5\tau u_3(j)) \geq \left(\frac{\tau^2 u_3(j)^2}{4}-\frac{\tau^4 u_3(j)^4}{48}\right) = \frac{\tau^2 u_3(j)^2}{4}\left(1-\frac{\tau^2 u_3(j)^2}{12}\right) > 0,$$

which is valid for all $u_3(j)^2 := (x_{0,3}/\tau k_m)^2 \leq (\pi/\tau k_m)^2$. This leads to the estimate

$$r_1\big(u_1(j)^2+u_2(j)^2\big) \leq \frac{12\cdot r_1}{12-\left(\frac{x_{0,3}^2}{k_m^2}\right)}\left(\frac{x_{0,1}^2}{\tau^2 k_m^2}+\frac{x_{0,2}^2}{\tau^2 k_m^2}\right) \leq \frac{12\cdot r_1}{12-\left(\frac{\pi^2}{k_m^2}\right)}\left(\frac{x_{0,1}^2}{\tau^2 k_m^2}+\frac{x_{0,2}^2}{\tau^2 k_m^2}\right). \quad (3.10)$$

The term $12/\left(12-\frac{\pi^2}{k_m^2}\right)$ has an upper bound of 1.26 for $k_m \geq 2$. Choosing now the weight r_1 as in Assumption (3.6), leads to the estimate

$$r_1\left(u_1(j)^2+u_2(j)^2\right) \leq \lambda\cdot q_1\left(\frac{x_{0,1}^2}{\tau^2 k_m^2}+\frac{x_{0,2}^2}{\tau^2 k_m^2}\right). \quad (3.11)$$

We note that the term $u_3(j)^2/4\sin^2(0.5\tau u_3(j))$ can be defined for $u_3(j)=0$ as

$$\lim_{u_3(j)\to 0}\frac{u_3(j)^2}{4\sin^2(0.5\tau u_3(j))} = \frac{1}{\tau^2}.$$

Therefore, the estimate (3.11) holds for $u_3(j)=0$ too.

As a result, using the assumption $r_3 \leq \lambda q_3$ from Eq. (3.6), the running costs $\ell_\tau(x_{u_{\mathbf{x}_0}}(j;\mathbf{x}_0),\mathbf{u}_{\mathbf{x}_0}(j))$ can be estimated by

$$\ell_\tau(x_{u_{\mathbf{x}_0}}(j;\mathbf{x}_0),\mathbf{u}_{\mathbf{x}_0}(j)) \leq \left[\left(\frac{k_m-j}{k_m}\right)^2+\frac{\lambda}{\tau^2 k_m^2}\right]\underbrace{\sum_{i=1}^{3} q_i x_{0,i}^2}_{\|\mathbf{x}_0\|_Q^2},$$

for all $j \in \{0, \dots, k_m\}$. The sequence c_j in Inequality (3.5) can now be obtained as in Eq. (3.7), where the $c_j \equiv 0$ for all $j > k_m$. □

Now the growth bound γ_k, $k \in \mathbb{N}_0$ can be computed as

$$\gamma_k := \sum_{j=0}^{k-1} c_j = \frac{1}{k_m^2} \sum_{j=0}^{k-1} \left[j^2 - 2k_m \cdot j + k_m^2 + \frac{\lambda}{\tau^2} \right] = \frac{1}{k_m^2} \left[\left(k_m^2 + \frac{\lambda}{\tau^2} \right) k + \sum_{j=0}^{k-1} j^2 - 2k_m \cdot \sum_{j=0}^{k-1} j \right].$$

Using the sum formulas

$$\sum_{j=0}^{k-1} j = \frac{k(k-1)}{2} \text{ and } \sum_{j=0}^{k-1} j^2 = \frac{k(k-1)(2k-1)}{6},$$

γ_k reduces to

$$\gamma_k = \frac{1}{k_m^2} \left[\frac{k^3}{3} - \left(k_m + \frac{1}{2} \right) k^2 + \left(k_m^2 + k_m + \frac{\lambda}{\tau^2} + \frac{1}{6} \right) k \right]. \tag{3.12}$$

Similar to the continuous-time case, γ_k only depends on the number of steps k_m, the sampling time τ, and the weight ratio λ.

3.4.2 Asymptotic Stability for the Shortest Prediction Horizon

The shortest possible prediction horizon of discrete-time MPC without terminal constraints or costs is $N = 2$. In this case, one control and the successor state are dominating the cost function given by Eq. (3.3). As a result, the special structure of γ_k, given by Eq. (3.12), leads to the following theorem.

Theorem 3.1. *Given $N = 2$, the weight ratio λ of Assumption (3.6) and the sampling time τ, there exists a maneuver length $k_m^\star$ such that for all $k_m > k_m^\star$, the performance index $\alpha_N = \alpha_2$ given by (2.36), is ensured to be positive; thus, the closed-loop asymptotic stability of the considered holonomic mobile robot under MPC without terminal conditions is guaranteed for the shortest possible prediction horizon. Moreover, for the same maneuver length an upper bound on the performance index α_2, denoted by $\bar{\alpha}_2$, can be computed and, thus, a lower bound on the MPC closed-loop performance is estimated via Inequality (2.38) as*

$$V_\infty^{\mu_2}(\mathbf{x}) \leq \bar{\alpha}_2^{-1} V_\infty(\mathbf{x}). \tag{3.13}$$

□

Proof. For $N = 2$, the performance estimate α_N in Eq. (2.36) reduces to

$$\alpha_2 = 1 - (\gamma_2 - 1)^2. \tag{3.14}$$

Thus, the asymptotic stability condition can be written as

$$0 < \gamma_2 < 2. \tag{3.15}$$

From Eq. (3.12), γ_2 is given by

$$\gamma_2 = \frac{1}{k_m^2}\left[2k_m^2 - 2k_m + \left(\frac{2\lambda}{\tau^2} + 1\right)\right]. \tag{3.16}$$

Then, as $\gamma_2 > 0$ for all $k_m \geq 1$, the left inequality of (3.15) is ensured. Moreover, choosing k_m as

$$k_m > k_m^\star := \left\lceil \frac{\lambda}{\tau^2} + \frac{1}{2} \right\rceil \tag{3.17}$$

satisfies the right inequality of (3.15).

The upper bound on α_2 can be computed by finding k_m such that $\frac{\partial \alpha_2}{\partial k_m} = 0$. This reduces to finding k_m such that $\frac{\partial \gamma_2}{\partial k_m} = 0$. Differentiating γ_2 given by Eq. (3.16) with respect to k_m results in

$$\frac{\partial \gamma_2}{\partial k_m} = \frac{2}{k_m^2}\left[1 - \frac{1}{k_m}\left(\frac{2\lambda}{\tau^2} + 1\right)\right].$$

For $\frac{\partial \gamma_2}{\partial k_m} = 0$, we have

$$k_m = \frac{2\lambda}{\tau^2} + 1 > k_m^\star.$$

Using this value, we can then compute an upper bound on α_2 and, thus, a lower bound on the MPC closed-loop performance, via Eq. (3.14) as

$$\bar{\alpha}_2 := \frac{\lambda/\tau^2 + 1/4}{(\lambda/\tau^2 + 1/2)^2}. \tag{3.18}$$

□

It is therefore shown by Theorem 3.1 that the closed-loop asymptotic stability can be guaranteed by sufficiently increasing the number of open-loop maneuver steps k_m. Moreover, it can be inferred from the lower bound on the MPC performance $\bar{\alpha}_2$ given by Eq. (3.18) that the index is always positive and converges to 1 as λ/τ^2 is chosen small. Therefore, by means of Inequality (3.13), infinite horizon MPC performance can be approached by reducing λ/τ^2 despite using the shortest prediction horizon, i.e. $N = 2$.

We remark that prolonging the open-loop maneuver time/steps leads to more conservative growth bound γ_i in Eq. (2.35). Therefore, prolonging the open-loop time does not affect the MPC algorithm.

3.5 Numerical and Experimental Results

In this section, we show the numerical and the experimental results of implementing the MPC algorithm presented in Section 3.3. Herein, the kinematic equation of the holonomic robot shown in (2.47) is used for state prediction. The mecanum wheels robot (*Summit-XL-Steel* [81]) is used in the experiments, see Fig. 1.2. States are constrained to

$$X := [-2, 2]^2 \times [-\pi, \pi].$$

The input constraints are as described in Eq. (3.1), where $r = 0.127$ m is the wheel radius, $\bar{\mathbf{u}}_w = 12$ rad/s is the maximum wheel speed, $a = 0.223$ m is half the wheel base length and $b = 0.205$ m is half the wheel base width, see Fig. 1.2 and the specification of the experimental platform [81]. This results in the maneuver time $\bar{t}_m = 3.5$ s, which ensures control constraints satisfaction for all initial conditions in X, by means of Eq. (3.8).

The simulations as well as the experiments were conducted until the following condition is met

$$||\mathbf{x}(n) - \mathbf{x}^\star|| \leq 0.05,$$

where $\mathbf{x}(n)$ is the state measurement at time step $n \in \mathbb{N}$ and $\mathbf{x}^\star = (0, 0, 0)^\top$ is the reference set point.

3.5.1 Numerical Results

The simulations were run for a series of eight different initial conditions, which were all then stabilized to the origin. The initial orientation was set to $x_{0,3} = \pi$ for each of the initial conditions, as this is the maximum deviation form the reference orientation $x_3^\star = 0$.

Simulations were conducted using MATLAB; the OCP (3.4) was set up symbolically using CasADi toolbox [5]. Here, the multiple-shooting discretization method [55] is used to convert the OCP to a Nonlinear Programming Problem (NLP). The NLP is then solved using the Interior-Point method implemented in IPOPT [93]. The running costs are defined by the way of using Eq. (3.2) with $\lambda = 0.05$. Additionally, the prediction horizon is set to $N = 2$ and the sampling time to $\tau = 0.025$ s. Here, the time shift $\delta = m\tau = \tau$; the maneuver length $k_m^\star$ required for asymptotic stability is $k_m^\star = 81$ by means of Theorem 3.1; and, by means of Eq. (3.18), we have the performance index $\bar{\alpha}_2 = 0.01238$.

The resulting closed-loop paths can be seen in Fig. 3.2 (left). It can be noted from these results that the optimal path for the holonomic mobile robot in each case is to take an almost straight path to the origin while the orientation of the robot changes as the robot moves towards the desired position.

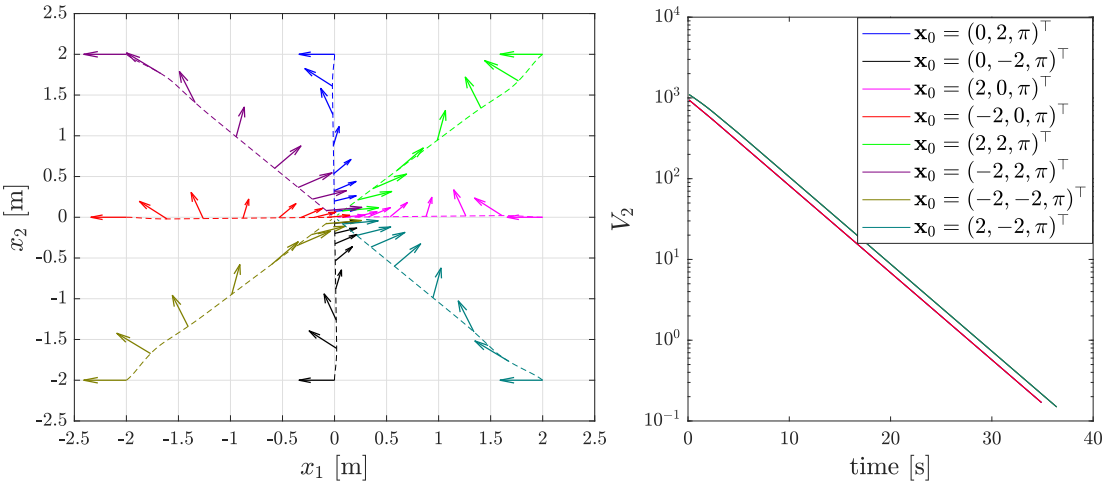

Figure 3.2: Left: Closed-loop paths for the simulation results for $N = 2$; the dashed line represents the robot's path and the arrow represent its orientation along the path. Right: value function evolution in simulation for $N = 2$.

The maximum wheel speed for two of the initial conditions is shown in Fig. 3.3 (left). The results confirm that the input constraints are not violated, i.e. Eq.(3.1) is satisfied for all time steps; the same behavior is observed for the remaining initial conditions.

To show that the system is asymptotically stable, we can observe that the evolution of the value function is monotonically decreasing with time, see Fig. 3.2 (right). Therefore, we can verify that the conditions for Theorem 2.2 are met for each initial condition in the simulation. Moreover, as the prediction horizon length is $N = 2$, we can find for each initial

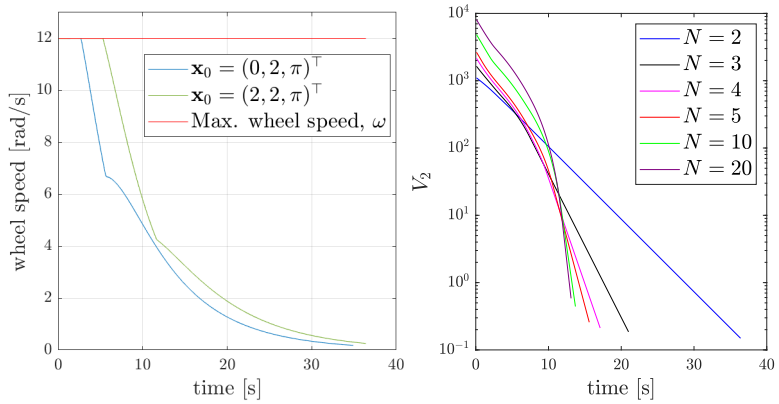

Figure 3.3: Left: maximum wheel speed for two initial conditions in simulation. Right: evolution of value function at different prediction horizon length N where $\lambda = 0.05$, $\tau = 0.025$ s, and $\mathbf{x}_0 = (2, 2, \pi)^\top$.

condition that $V_2(\mathbf{x}_0)/\ell_\tau^\star(\mathbf{x}_0) \leq \gamma_2 < 2$, see Conditions (2.35) and (3.15). In Table 3.1, we see that this holds for all initial conditions in the simulation.

Table 3.1: $V_2(\mathbf{x}_0)/\ell_\tau^*(\mathbf{x}_0)$ for each initial condition in simulations

$\mathbf{x}_0^\top$	$\frac{V_2(\mathbf{x}_0)}{\ell_\tau^\star(\mathbf{x}_0)}$	$\mathbf{x}_0^\top$	$\frac{V_2(\mathbf{x}_0)}{\ell_\tau^\star(\mathbf{x}_0)}$
$(0, 2, \pi)$	1.9877	$(2, 2, \pi)$	1.9887
$(0, -2, \pi)$	1.9877	$(-2, 2, \pi)$	1.9887
$(2, 0, \pi)$	1.9877	$(-2, -2, \pi)$	1.9887
$(-2, 0, \pi)$	1.9877	$(2, -2, \pi)$	1.9887

Through running these simulations, we observed some interesting results of varying parameters such as λ, the prediction horizon N, and the sampling time τ on the evolution of the value function V_N. First, a lower λ resulted in a faster convergence of the mobile robot to the set point, which is fairly intuitive; a lower λ means a lower ratio on the weights for the control inputs compared to the weights for the state. This will result in a solution of the OCP (3.4) with less conservative control actions, and therefore faster convergence of the robot to the set point. What is less intuitive is the effect of the prediction horizon and the sampling time, we explore both here. For these simulations, we used an initial condition of $\mathbf{x}_0 = (2, 2, \pi)^\top$. Fig. 3.3 (right) shows the effect of the prediction horizon length. Here, we see that larger prediction horizons produce a faster convergence of the robot to the set point. We also see that the evolution of the value function for $N = 2$ is roughly linear in the logarithmic scale, where for longer prediction horizon lengths the

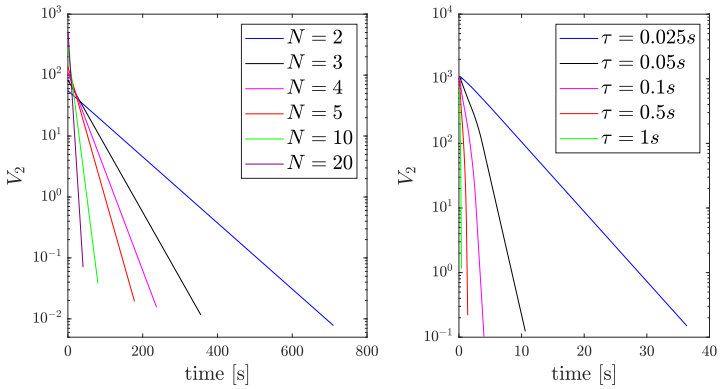

Figure 3.4: Left: evolution of the value function for different prediction horizon lengths N, where $\lambda = 1$ and $\tau = 0.025$ s. Right: evolution of the value function for different sampling intervals τ, where $\lambda = 0.05$ and $N = 2$. For the two figures, we have $\mathbf{x}_0 = (2, 2, \pi)^\top$.

value function evolution becomes steeper. Finally, we can also observe diminishing returns in terms of faster convergence to the set point for increasing the prediction horizon; going from $N = 2$ to $N = 3$ reduces convergence time by about 15 s, while going from $N = 10$ to $N = 20$ has a difference of less than 1 s.

Now, we investigate the evolution of the value function for $\lambda = 1$. In Fig. 3.4 (left), we see the effect of having $\lambda = 1$ for different prediction horizon lengths; convergence to the set point takes significantly longer time for smaller N. Additionally, we can see that the evolution of V_N in each case is nearly linear, not only for $N = 2$. Fig. 3.4 (right) shows the results of our investigation for the effect of τ on the evolution of the value function V_N. Here, $\lambda = 0.05$ and $N = 2$ are kept constant and τ is varied. There is an apparent tendency that increasing τ reduces the convergence time.

3.5.2 Experimental Results

Algorithm 1 was validated experimentally using a *Summit-XL-Steel* holonomic mobile platform, see Fig. 1.2. In the experiments, the mobile platform is stabilized to the origin from eight different initial positions. The experimental platform runs through the Robot Operating System (ROS) [76]; thus, a ROS node was developed in Python to implement Algorithm 1 in which CasADi toolbox was used to symbolically setup the MPC controller with the same settings used in the simulations. Here, the running costs in Eq. (3.2) is used with $\lambda = 0.05$. Additionally, the prediction horizon was set to $N = 2$ and the sampling time to $\tau = 0.025$ s.

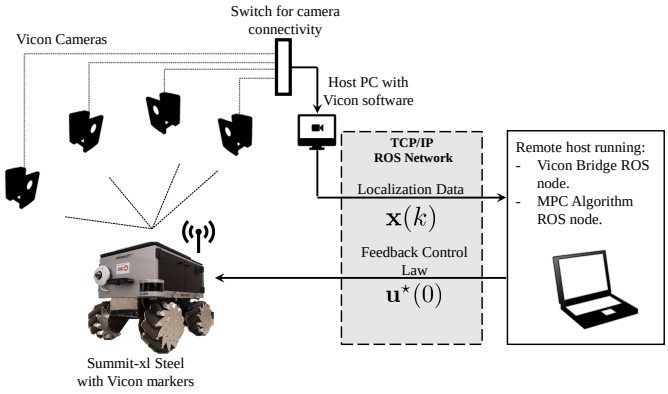

Figure 3.5: Block diagram of the experimental setup used to validate Algorithm 1.

To ensure that the localization of the robot would not affect our results, a *Vicon* motion capture system adopting 12 cameras was used to localize the mobile platform. The position of the robot from the Vicon system was transmitted to ROS using a Vicon bridge [1]. The block diagram of the experimental setup is shown in Fig. 3.5.

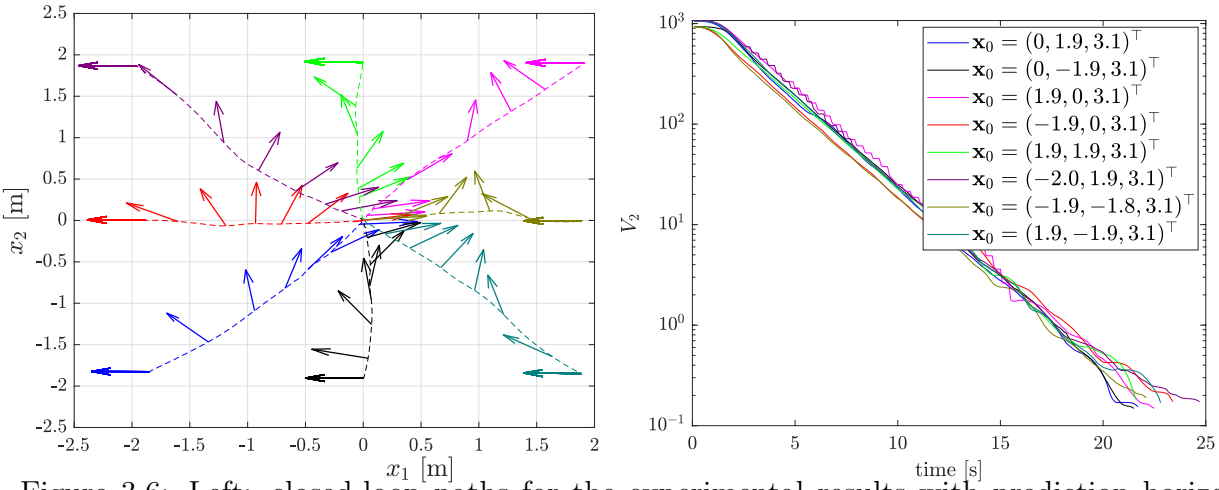

Figure 3.6: Left: closed-loop paths for the experimental results with prediction horizon length $N = 2$; the dashed line represents the robot's path and the arrow represent its orientation. Right: value function evolution in experiments for $N = 2$.

The results for eight different initial conditions can be seen in Fig. 3.6 (left). The value function V_2 evolution was tracked for each case. Moreover, a comparison of the closed-loop paths in experiments and their simulations counterparts is presented in Fig. 3.7. In Fig. 3.6

(right), we see that, in each case, V_2 is monotonically decreasing towards zero until the experiment is stopped, except for few moments for only three initial conditions; this is because of the unmodeled dynamics of the robot and/or floor surface irregularities. The results indicates the (inherent) robustness of the proposed controller.

Additionally, as can be seen in Fig. 3.7, the closed-loop paths of the experiments are very similar to that for the simulations especially for the initial conditions on the horizontal and vertical axes. Finally, we verify that the Conditions (2.35) and (3.15) are met for each initial condition in the experiments. In Table 3.2, we see that $V_2(\mathbf{x}_0)/\ell_\tau^\star(\mathbf{x}_0) \leq \gamma_2 < 2$ holds for all initial conditions in the experiments. In summary, experimental results show very good alignment with theory and simulations.

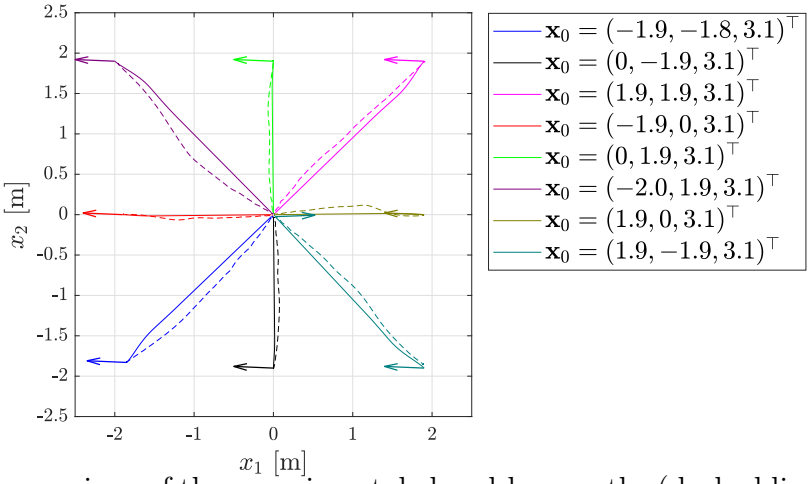

Figure 3.7: Comparison of the experimental closed-loop paths (dashed line) and simulation closed-loop paths (solid line); initial and final orientations are represented by arrows.

3.6 Conclusion

In this chapter, we rigorously show cost controllability in order to ensure asymptotic stability of a desired set point for holonomic mobile robots using MPC without stabilizing terminal conditions. To this end, we construct a growth function by using suitably designed open-loop control maneuvers. On the one hand, the conducted analysis allows to conclude asymptotic stability for the shortest possible prediction horizon. On the other hand, we also provide a bound on the MPC closed-loop costs in comparison to the infinite-horizon optimal controller. Results are verified by a series of numerical simulations and real-time lab experiments confirming the theoretical findings.

Table 3.2: $V_2(\mathbf{x}_0)/\ell_\tau^\star(\mathbf{x}_0)$ for each initial condition in experiments

$\mathbf{x}_0^\top$	$\frac{V_2(\mathbf{x}_0)}{\ell_\tau^\star(\mathbf{x}_0)}$
$(0.00, 1.91, 3.11)$	1.9876
$(0.00, -1.90, 3.14)$	1.9877
$(1.89, -0.01, 3.14)$	1.9886
$(-1.88, 0.00, 3.13)$	1.9869
$(1.91, 1.90, 3.13)$	1.9875
$(-1.96, 1.87, 3.13)$	1.9877
$(-1.85, -1.83, 3.14)$	1.9857
$(1.89, -1.85, 3.13)$	1.9887

Chapter 4

Acceleration Constrained Model Predictive Control

4.1 Introduction

In this chapter, we consider the regulation control of holonomic mobile robots with limits on acceleration. Such constraints ensure the smooth operation of the robot and reduce inertial forces on its components and payloads from sudden motions. This can be achieved by using the dynamic model of the robot and constraining the torque applied by the wheel motors [82] or by building time scaled trajectory to a desired point using segments with restricted acceleration [15]. This has also been done by including an additional constraint on acceleration in an MPC scheme [8]. The ability of MPC to handle constrained non-linear systems makes it an ideal candidate for our acceleration constrained regulation control problem.

The motion of mobile robots is normally analyzed by means of their kinematic models. Under MPC, the acceleration limits of such systems can be considered by expanding the kinematic model to include the actuator dynamics, which characterize the acceleration limit [69]. Alternatively, acceleration constraints can be considered by limiting the change in consecutive velocity commands of the MPC optimal control sequence [8]. For MPC algorithms which only consider kinematic models with a velocity control input such as Chapter 3, the optimal solution will often be to initially apply the maximum control effort to quickly drive the system closer to the desired set point. This results in a large initial acceleration of the system and, thus, an undesirable behavior.

In this chapter, we study the closed-loop asymptotic stability of the acceleration-constrained regulation of a holonomic mobile robot under MPC *without stabilizing constraints or costs*. This is an extension of the work presented in Chapter 3, where limits on accelerations are not considered. Here, we provide a detailed analysis of the effect of acceleration limits on the stability proof. This is achieved by, first, expanding the kinematic model of the robot to consider the vehicle's acceleration. Second, by establishing an admissible subset of the state space, where constraints on inputs and states are satisfied for all times. This is achieved using the theory of barriers presented in [24].

The rest of the chapter is organized as follows: In Section 4.2, we describe the kinematic model of holonomic mobile robots; the acceleration limit formulation; and the derivation of the admissible states set via the theory of barriers; and the MPC algorithm. Using the MPC stability results presented in Section 2.2.3, the open-loop control maneuvers are developed, and an expression for the value function bound (growth function) is derived in Section 4.4. It is then shown that this growth function can be used to find a stabilizing prediction horizon. Simulation results are presented in Section 4.5. Finally, the implications of the results of this paper as well as future works are discussed in Section 4.6.

4.2 Problem Formulation and the Admissible Set

In this section, using the acceleration controlled kinematic model of a holonomic robot, we define constraints on the states and inputs. We then show the detailed process (via the theory of barriers) of deriving an admissible state set of the resulting kinematic model. Finally, we show the set-point regulation problem for which we propose a model predictive control scheme in continuous-time without stabilizing constraints or costs.

4.2.1 Holonomic Mobile Robot Kinematic Model

As we are designing an acceleration constrained model predictive controller, it becomes advantageous to use the acceleration controlled kinematic model in (2.45) with $\mathbf{x}(t_0) = \mathbf{x}_0$ in order to directly constrain the acceleration.

We consider state constraints defined by the following functions

$$g_i(\mathbf{x}(t)) \leq 0, \quad \forall t \in [t_0, \infty] \quad \text{for} \quad i = 1, \ldots, p. \tag{4.1}$$

Here, we have $p = 12$ constraints on the states, as each state in $\mathbf{x}$ has both an upper and a lower bound. The bounds can be expressed as follows

$$\begin{aligned} g_{2j-1}(\mathbf{x}(t)) &= x_j(t) - \bar{x}_j, && \text{for} \quad j = 1, \dots, 6, \\ g_{2j}(\mathbf{x}(t)) &= -x_j(t) - \bar{x}_j, && \text{for} \quad j = 1, \dots, 6. \end{aligned}$$

for $\bar{x}_j \geq 0$, $j = 1, \dots, 6$.

Moreover, the control input (acceleration) constraints are defined by the set

$$U = [-\bar{u}_1^a, \bar{u}_1^a] \times [-\bar{u}_2^a, \bar{u}_2^a] \times [-\bar{u}_3^a, \bar{u}_3^a] \tag{4.2}$$

with the saturation limits $\bar{u}_1^a, \bar{u}_2^a, \bar{u}_3^a > 0$.

We remark that, unlike the case in which acceleration constraints are not considered [67], if the robot starts with an initial position on the boundaries of the allowable states, for example where $g_1(\mathbf{x}) = 0$ with an initial velocity away from the allowable states, $x_4 > 0$, a control function $\mathbf{u}(t) \in U$ for all $t \geq 0$ for which the constraints are satisfied, i.e. $g_i(\mathbf{x}) \leq 0$ for all $i = 1, \dots, p$ does not exist. Therefore, it is essential to construct a set of admissible states (formally defined later) which prohibits the aforementioned initial condition. In the following subsection, we achieve this by applying the theory of barriers.

4.2.2 Theory of Barriers

Here, we briefly summarize the theory of barriers as presented in [24]. We look at this theory as it applies to the considered holonomic mobile robot with acceleration constrains. Herein, we denote by $\mathbf{x}_\mathbf{u}(\cdot, \mathbf{x}_0)$ the trajectory generated by the control input $\mathbf{u} \in U$ via the system dynamics (2.45) for the initial condition $\mathbf{x}_0$. Furthermore, we define the following sets

$$\begin{aligned} X &:= \{\mathbf{x} \in \mathbb{R}^6 : g_i(\mathbf{x}) \leq 0, \quad \forall i \in \{1, \dots, p\}\} \\ X_- &:= \{\mathbf{x} \in \mathbb{R}^6 : g_i(\mathbf{x}) < 0, \quad \forall i \in \{1, \dots, p\}\} \\ X_0 &:= \{\mathbf{x} \in X : \exists i \in \{1, \dots, p\} \quad \text{s.t. } g_i(\mathbf{x}) = 0\}, \end{aligned}$$

and denote by

$$\mathbb{I}(\mathbf{x}) = \{i \in \{1, \dots, p\} : g_i(\mathbf{x}) = 0\}$$

the indices of active constraints at a state $\mathbf{x}$.

We can now define the set of admissible initial conditions denoted by $\mathcal{A}$. This is the set of states $\mathbf{x}$ for which there exists a control function $\mathbf{u} \in U$ such that the solution of system (2.45) with the initial condition $\mathbf{x}_0$ satisfies the constraints (4.1) for all future time. $\mathcal{A}$ is formally expressed as

$$\mathcal{A} := \{\mathbf{x}_0 \in X : \exists \mathbf{u} \in U, \mathbf{x}_\mathbf{u}(t, \mathbf{x}_0) \in X \quad \forall t \in [t_0, \infty)\}$$

The boundary of the admissible set $\mathcal{A}$ is denoted by $\partial\mathcal{A}$. Additionally, we introduce the barrier set

$$[\partial\mathcal{A}]_- = \partial\mathcal{A} \cap X_-.$$

For every point $\mathbf{x} \in [\partial\mathcal{A}]_-$, there exists an input $\bar{\mathbf{u}} \in U$, such that the resulting integral curve runs along $[\partial\mathcal{A}]_-$, and satisfies a minimum/maximum like principle. If this curve intersects the boundary of the constraint set X_0, it must do so tangentially. These conditions are expressed by the following theorem, see [24, Theorem 7.1] for the proof. In viability theory, the admissible set $\mathcal{A}$ is called the viability kernel, see [84].

Theorem 4.1. *Every integral curve $\mathbf{x}_{\bar{\mathbf{u}}}(\cdot, \mathbf{x}_0)$ with $\mathbf{x}_0$ on the boundary $[\partial\mathcal{A}]_-$ satisfies the following necessary conditions.*
There exists a continuous maximal solution to $\xi_{\bar{u}}$ to the adjoint equation

$$\dot{\xi}_{\bar{\mathbf{u}}}(t) = -\left(\frac{\partial f}{\partial \mathbf{x}}(\mathbf{x}_{\bar{\mathbf{u}}}(t), \bar{\mathbf{u}}(t))\right)^\top \xi_{\bar{\mathbf{u}}}(t), \tag{4.3}$$

where if D denotes the differential derivative, we have

$$\xi_{\bar{\mathbf{u}}}(\bar{t}) = (Dg_{i^*}(\mathbf{y}))^\top$$

such that

$$\min_{\mathbf{u} \in U} \left\{\xi_{\bar{\mathbf{u}}}(t)^\top f(\mathbf{x}_{\bar{\mathbf{u}}}, \mathbf{u})\right\} = \xi_{\bar{\mathbf{u}}}(t)^\top f(\mathbf{x}_{\bar{\mathbf{u}}}, \bar{\mathbf{u}}(t)) = 0 \tag{4.4}$$

at every point $t \geq 0$. In Equation (4.3), $\bar{t}$ denotes the time at which the states of the system are tangent to a boundary, i.e. $\mathbf{x}^{\bar{\mathbf{u}}}(\bar{t}) = \mathbf{y}$, and $\mathbf{y} \in X_0$, so $\mathbf{y}$ satisfies

$$g_i(\mathbf{y}) = 0, \quad i \in \mathbb{I}(\mathbf{y})$$

and

$$\min_{\mathbf{u} \in U} \max_{i \in \mathbb{I}(\mathbf{y})} L_f g_i(\mathbf{y}, \mathbf{u}) = L_f g_{i^*}(\mathbf{y}, \bar{\mathbf{u}}(\bar{t})) = 0. \tag{4.5}$$

Here, $L_f g(\mathbf{x}, \mathbf{u}) := Dg(\mathbf{x}) f(\mathbf{x}, \mathbf{u})$ is the Lie derivative of the function $g : \mathbb{R}^n \to \mathbb{R}$ with respect to $f(\cdot, \mathbf{u})$ at the point $\mathbf{x}$.

These conditions can be used to build the admissible set $\mathcal{A}$. We start by identifying the point $\mathbf{y}$ tangent to the constraint located on X_0 via Condition (4.5). Then, using Condition (4.4), the input $\bar{u}$ corresponding to the integral curve along the barrier function $[\partial\mathcal{A}]_-$ can be determined. Finally, the barrier function $[\partial\mathcal{A}]_-$ is calculated from the backward integration of the system (2.45) and the adjoint dynamics (4.3), with starting points at $\mathbf{x}(\bar{t})$ and $\xi(\bar{t})$, respectively.

4.2.3 The Admissible Set $\mathcal{A}$

Using Theorem 4.1, we can now derive the admissible set $\mathcal{A}$ of initial conditions $\mathbf{x}_0$. To simplify the analysis of the derivation of $\mathcal{A}$ and the upcoming stability analysis, we consider the bounds $\bar{x}_3 = \bar{x}_6 = 0$ on the constraints functions given by Inequality (4.1). This implies that $x_3(t) = x_6(t) = 0$, giving a fixed orientation for all $t \geq 0$. As a result, the mapping $f(\mathbf{x}, \mathbf{u})$ of System (2.45) becomes

$$f(\mathbf{x}, \mathbf{u}) = \begin{pmatrix} x_4 & x_5 & u_1 & u_2 \end{pmatrix}^\top, \tag{4.6}$$

where $\mathbf{x} = (x_1, x_2, x_4, x_5)^\top$ and $\mathbf{u} = (u_1, u_2)$. The resulting system is a 2-dimensional version of the double integrator system considered in [29]. Moreover, we redefine the set U to be

$$U = [-\bar{u}_1^a, \bar{u}_1^a] \times [-\bar{u}_2^a, \bar{u}_2^a].$$

Let us now consider the constraint $g_1(\mathbf{x}) = x_1(t) - \bar{x}_1 \leq 0$ as the active constraint; thus, the active constraint set is $\mathbb{I}(\mathbf{x}) = \{1\}$. Using Condition (4.5), we can calculate the state tangent to this boundary as $x_4 = 0$. This corresponds to the set of states with $\dot{x}_1 = 0$, i.e. zero velocity perpendicular to the constraint $g_1(\mathbf{x})$. We can then obtain the tangent point $\mathbf{y} = \mathbf{x}(\bar{t})$ as $\mathbf{y} = (\bar{x}_1, x_2(\bar{t}), 0, x_5(\bar{t}))^\top$, where $x_2(\bar{t}) \in [-\bar{x}_2, \bar{x}_2]$ and $x_5(\bar{t}) \in [-\bar{x}_5, \bar{x}_5]$. Intuitively, this indicates that the position along x_2 and velocity x_5 can be chosen arbitrarily, which is possible for a holonomic mobile robot as the control along each direction can be controlled independently. Therefore, we set $x_5(\bar{t}) = 0$ as this is independent of the boundary we are considering. In conclusion, looking at one point $x_2(\bar{t}) = 0$ along the boundary of the considered function $g_1(\mathbf{x})$, i.e. give us the tangential point $\mathbf{y} = (\bar{x}_1, 0, 0, 0)^\top$.

In a similar fashion, we can find a tangential point on the negative x_1 constraint, i.e. $g_2(x(t)) = -x_1 - \bar{x}_1$. This gives the admissible set of x_1 and x_4 on $\mathbb{R}^2$, see Fig. 4.1 (left). Furthermore, it is straightforward to extend this analysis to the set of allowable states in $\mathbb{R}^4$ over x_1, x_2, x_4 and x_5. The resulting admissible set $\mathcal{A}$ is shown in Fig. 4.1 (right).

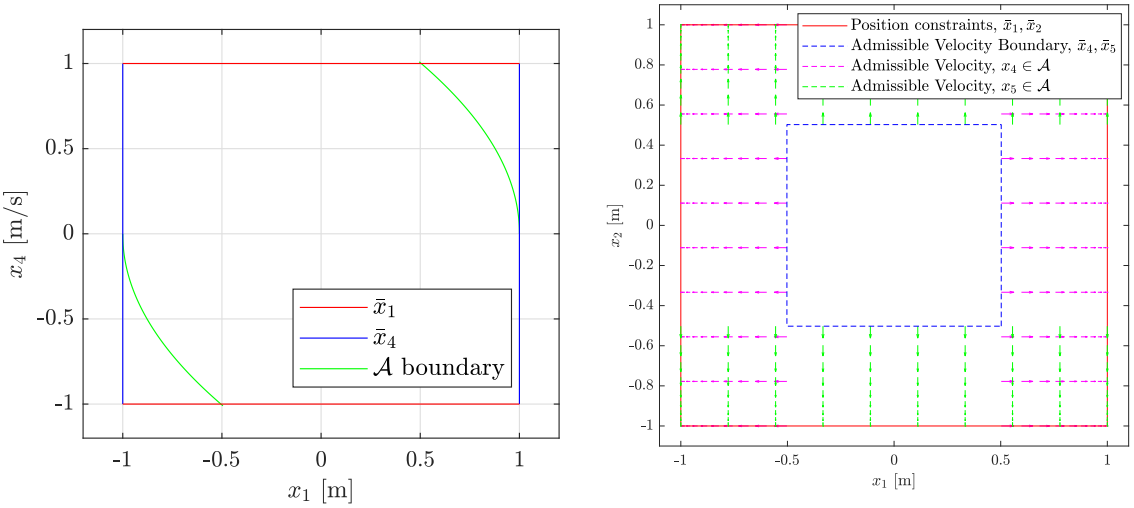

Figure 4.1: Left: Set of admissible initial states over x_1 and x_4 in $\mathbb{R}^2$. Right: Set of admissible initial states $\mathcal{A}$ for the states x_1, x_2, x_4, and x_5. Here the arrows represent the maximum allowable velocity in a given direction and location, the largest arrows representing a velocity of $x_i = 1$ [m/s] for $i = 4, 5$.

With the admissible set $\mathcal{A}$ now defined, we denote by $u|_{[0,T]} \in \mathcal{U}_T(x_0)$ the admissible control function for $t \in [0, T]$, where $u \in \mathcal{PC}([0, T, U)$, $T \geq 0$ and

$$\mathcal{U}_T(\mathbf{x}_0) := \{\mathbf{u}(t) \in U | \ \mathbf{x}_\mathbf{u}(t, x_0) \in \mathcal{A} \quad \forall t \in [0, T]\}, \tag{4.7}$$

where $\mathbf{x}_0 \in \mathcal{A}$. Additionally, if the admissible control function $\mathbf{u}|_{[0,T]} \in \mathcal{U}_T(\mathbf{x}_0)$ satisfies (4.7) for all $T > 0$, the resulting set of all admissible control functions $\mathbf{u} \in \mathcal{PC}([0, \infty], U)$ is denoted by $\mathcal{U}_\infty(\mathbf{x}_0)$.

Now, the control objective is to regulate the robot posture to a (controlled) equilibrium $\mathbf{x}^\star$, where $f(\mathbf{x}^\star, 0) = \mathbf{x}^\star$. Without loss of generality, we choose the origin as the set-point, i.e. $\mathbf{x}^\star = 0_{\mathbb{R}^4}$. To achieve this objective, we use a model predictive control (MPC) scheme without terminal constraints or costs presented in the next section.

4.3 MPC without Stabilizing Terminal Constraints or Costs

In this Section, we propose a model predictive control (MPC) scheme without stabilizing constraints or costs to regulate the system to the set point $\mathbf{x}^\star$. We then use the stability results of Theorem 2.3 in which the conditions for guaranteeing the closed-loop asymptotic stability are stated. As standard in MPC, we use the quadratic stage costs $\ell : \mathbb{R}^4 \times \mathbb{R}^2 \to \mathbb{R}_{\geq 0}$

$$\ell(\mathbf{x}, \mathbf{u}) = \|\mathbf{x} - \mathbf{x}^\star\|_Q^2 + \|\mathbf{u}\|_R^2 \tag{4.8}$$

as a performance measure, where

$$Q = \text{diag}(q_1, q_2, q_4, q_5) \text{ and } R = \text{diag}(r_1, r_2)$$

are positive definite weighting matrices. Note here that the indices of the weights match that for the states and controls. By integrating the running costs (4.8) over a prediction horizon $T = N\tau$, $N \in \mathbb{N}_{\geq 2}$ with the sampling period $\tau > 0$, we obtain the MPC cost function

$$J_T(\mathbf{x}_k, \mathbf{u}) := \int_{t_k}^{t_k+T} \ell(\mathbf{x}_\mathbf{u}(t; \mathbf{x}_k), \mathbf{u}(t))dt, \tag{4.9}$$

where $t_k = k\tau$, $k \in \mathbb{N}_0$ is the sampling instance and $\mathbf{x}_k = \mathbf{x}(t_k)$. The MPC scheme control input is determined by repeatedly the following OCP at each time instance t_k.

$$V_T(\mathbf{x}_k) := \inf_{\mathbf{u} \in \mathcal{U}_T(\mathbf{x}_k)} J_T(\mathbf{x}_k, \mathbf{u}) \tag{4.10a}$$

$$\begin{aligned}
\textbf{s.t.} \quad & \mathbf{x}(t_k) = \mathbf{x}_k \\
& \dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)) \quad \forall t \in [t_k, t_k + T) && \text{(4.10b)} \\
& \mathbf{x}(t) \in \mathcal{A} \quad \forall t \in [t_k, t_k + T] && \text{(4.10c)} \\
& \mathbf{u}(t) \in U \quad \forall t \in [t_k, t_k + T) && \text{(4.10d)}
\end{aligned}$$

The solution to OCP (4.10) is the optimal control function $\mathbf{u}^\star(t - t_k; \mathbf{x}_k)$, $t \in [t_k, t_k + T)$ with the optimal value function $V_T : \mathcal{A} \to \mathbb{R}_{\geq 0}$ given by $V_T(\mathbf{x}_k) = J_T(\mathbf{x}_k, u^\star)$. In the OCP, the system dynamics (2.45) are enforced by constraint (4.10b) and the state and input constraints are enforced by (4.10c) and (4.10d) respectively. We now define a control horizon $\delta = m\tau$, $m \in [1, N - 1]$, on which we apply the MPC feedback law

$$\mu(t, \mathbf{x}_k) = \mathbf{u}^\star(t - t_k; x_k), \quad t \in [t_k, t_k + \delta)$$

By applying the feedback law μ to system (2.45) we obtain the closed-loop trajectory governed by

$$\dot{\mathbf{x}}_\mu(t, \mathbf{x}_0) = f(\mathbf{x}_\mu(t; \mathbf{x}_0), \mu(t - t_\delta, x_\delta)) \tag{4.11}$$

with $t_\delta = \delta \lfloor t/\delta \rfloor$ and $\mathbf{x}_\delta = \mathbf{x}_\mu(t_\delta, x_0)$, i.e. $t - t_\delta \in [0, \delta)$. The proposed controller is summarized in Algorithm 1. We remark that Algorithm 1 does not employ any stabilizing terminal constraints or costs. The recursive feasibility of Algorithm 1 is assured if the

Algorithm 1 MPC Scheme

Given: sampling period τ, prediction horizon T, control horizon δ and initial state $\mathbf{x}_0 \in \mathcal{A}$
Set: $k = 0$ and $\mathbf{x}_k := \mathbf{x}_0$.

1: Compute a minimizing control function $\mathbf{u}^\star \in \mathcal{U}_T(\mathbf{x}_k)$ such that $J_T(\mathbf{x}_k, \mathbf{u}^\star) = V_T(\mathbf{x}_k)$ holds.

2: Apply $\mathbf{u}^\star(t, \mathbf{x}_k), t \in [0, \delta)$ to the mobile robot.

3: Set $\mathbf{x}_{k+1} := \mathbf{x}_\mu(t_{k+1}; \mathbf{x}_0)$, increment k, and go to step 1.

initial state $\mathbf{x}_0$ is contained within the admissible set $\mathcal{A}$ defined in Section 4.2.2 according to the theory of boundaries. The recursive feasibility of the closed-loop MPC algorithm for a constrained linear-quadratic case is addressed in [29, Section 4].

Now, to ensure the asymptotic stability of the system, we must show that there exists some function $\beta \in \mathcal{KL}$, where $\mathcal{KL}$ is defined in A.2.2, such that

$$\|\mathbf{x}_\mu(t; x_0)\| \leq \beta(\|\mathbf{x}_0\|, t), \quad \forall t \geq 0 \tag{4.12}$$

is satisfied for all $x_0 \in \mathcal{A}$ [51]. Where the closed-loop trajectory $\mathbf{x}_\mu(t; x_0)$ is as defined by Eq. (4.11). The asymptotic stability of system (4.6) under the employed MPC scheme, without terminal conditions, can be summarized by Theorem 2.3. Note that where Theorem 2.3 refers to the set of allowable states, X, here we consider the admissible set $\mathcal{A}$. In the following section, we derive a growth bound $B(\cdot)$ such that inequality (2.39) and condition 2.42 holds for the considered holonomic mobile robot with acceleration constraints.

4.4 Growth Bound

As shown by [66], a growth function $B(t)$ which satisfies the cost controllability condition given by Inequality (2.39), for all $x_0 \in \mathcal{A}$, can be obtained using an integrable and bounded

function $c(t)$, i.e. $\int_0^\infty c(t) < \infty$, as

$$B(t) = \int_0^t c(s)\mathrm{d}s. \tag{4.13}$$

Here, we ensure that the function $c(\cdot)$ bounds the running costs at each time instance $t \geq 0$, i.e.

$$\ell(\mathbf{x}_{\mathbf{u}_{\mathbf{x}_0}}(t;\mathbf{x}_0), \mathbf{u}_{\mathbf{x}_0}(t)) \leq c(t) \cdot \ell^*(\mathbf{x}_0) \quad \forall \mathbf{x}_0 \geq \mathcal{A}, \tag{4.14}$$

for an admissible control function $\mathbf{u}_{\mathbf{x}_0} = \mathcal{U}_\infty(\mathbf{x}_0)$ steering the robot from the initial condition $\mathbf{x}_0$ to the reference $\mathbf{x}^\star = (0,0,0,0)^\top$.

4.4.1 Trajectory Generation

The calculation of the growth bound $B(\cdot)$ is based on designing an open-loop control maneuver from which the running costs ℓ can be evaluated and the function $c(\cdot)$ from Inequality (4.14) can be estimated. For the initial condition $\mathbf{x}_0 = (x_{0,1}, x_{0,2}, x_{0,4}, x_{0,5})^\top \in \mathcal{A}$, we design an open-loop control maneuver that steers the robot to the reference $\mathbf{x}^\star$ in a finite time. The chosen input function $\mathbf{u}_{x_0}$, yields sub-optimal running costs $\ell(\mathbf{x}_{\mathbf{u}_{\mathbf{x}_0}}(t;\mathbf{x}_0), \mathbf{u}_{\mathbf{x}_0}(t))$, $t \geq 0$, ensuring Inequality (4.14) holds when the optimal control input $\mathbf{u}^\star_{x_0}$ is applied, where $\mathbf{u}^\star_{x_0}$ is the applied control input from Algorithm 1. Since the considered system is holonomic (and also differentially flat), it's enough to choose independent trajectories $x_i(\cdot)$, $i \in \{1,2\}$, while deducing the trajectories $x_i(\cdot)$, $i \in \{4,5\}$ and $u_{x_0}(\cdot)$ by inverting the dynamics of the considered system. We refer to [56] for more details on differential flatness. The chosen open-loop trajectory is defined over 3 segments:

I a segment with constant acceleration $u(t) = u_I \in U$, for $t \in [0, t_k)$ in the direction of $x^\star$ resulting in the trajectories

$$\begin{aligned} x_i(t) &= x_{0,1} + x_{0,i+3}t + 0.5u_{I,i}t^2 \\ x_{i+3}(t) &= x_{0,i+3} + u_{I,i}t, \qquad i \in \{1,2\}, \end{aligned}$$

II a segment with constant velocity $x_i(t) \in \mathcal{A}$ for $i \in \{4,5\}$ and $t \in [t_k, t_k + t_m)$ in the direction of $x^\star$ resulting in the trajectories

$$x_i(t) = x_i(t_k) + x_{i+3}(t_k)(t - t_k), \quad i \in \{1,2\},$$

III a segment with constant deceleration $u(t) = u_{III} \in U$, for $t \in [t_k + t_m, t_k + t_m + t_n)$ to slow the robot to a stop, i.e. $x(t_k + t_m + t_n) = x^\star$; this results in the trajectories

$$\begin{aligned} x_i(t) &= x_i(t_k + t_m) + x_{i+3}(t_k)(t - t_k - t_m) + 0.5u_{III,i} \cdot (t - t_k - t_m)^2 \\ x_{i+3}(t) &= x_{i+3}(t_k + t_m) + u_{III,i} \cdot (t - t_k - t_m), \qquad i \in \{1, 2\}. \end{aligned}$$

By solving for the maximum velocity $x_{\max,i+3}$, $i \in \{1, 2\}$ in a trajectory, we can find the maneuver times. The maximum velocity is calculated as

$$x_{\max,i+3} = -\operatorname{sgn}(x_{0,i}) \cdot \min\left\{\sqrt{\bar{u}_i^a |x_{0,i}| + 1/2 x_{0,i+3}^2}, \bar{x}_{i+3}\right\}$$

We note that if $x_{\max,i+3} < \bar{x}_{i+3}$, $i \in \{1, 2\}$, then $t_m = 0$. Using this expression for $x_{\max,i+3}$, we calculate the maneuver times as

$$\begin{aligned} t_k &= \max_{i \in \{1,2\}, x_0 \in \mathcal{A}} \left(\frac{x_{\max,i+3} - x_{0,i+3}}{\bar{u}_i^a} \right), \\ t_m &= \max_{i \in \{1,2\}} \left(\frac{x_i(t_k + t_m) - x_i(t_k)}{x_{\max,i+3}} \right), \\ t_n &= \max_{i \in \{1,2\}} \left(\frac{-x_{\max,i+3}}{\bar{u}_i^a} \right) \end{aligned} \tag{4.15}$$

This choice of maneuver times ensures that the state and control input constraints will not be violated. Three such trajectories are shown in Fig. 4.2 (right). The acceleration (control) inputs for these trajectories are shown in Fig. 4.2 (left).

4.4.2 Growth Bound Derivation

Using the open-loop trajectory presented in the previous subsection, we summarize by the following proposition the derivation of the function $c(t)$ such that Inequality (4.14) is satisfied.

Proposition 2. *With the maneuver times developed in the previous subsection and the following assumptions on the weighting matrices Q and R of the running costs (4.8)*

$$q_{i+3} = q_i, \quad \lambda = \frac{r_i}{q_i}, \qquad i \in \{1, 2\},$$

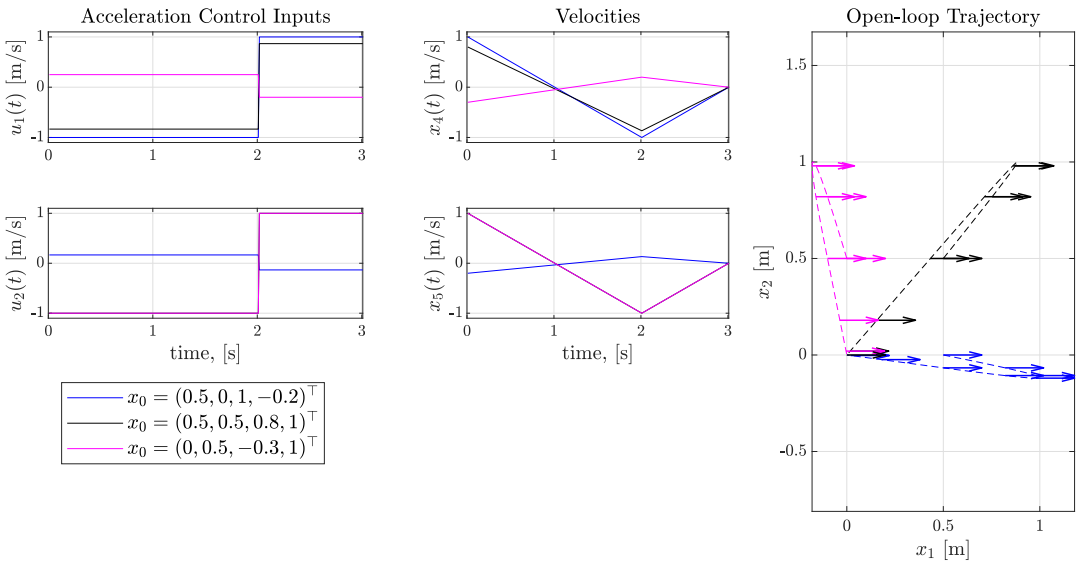

Figure 4.2: Left: Acceleration control inputs $u_1(t)$ and $u_2(t)$ and Mid: velocities $x_4(t)$ and $x_5(t)$ for open loop maneuver with $t_k = 2[s]$, $t_m = 0[s]$ and $t_n = 1[s]$. Right: Open-loop maneuver based on the applied controls, with arrow representing the orientation of the robot (which is fixed) and the dashed lines representing the path.

Inequality (2.39) holds with $B(t)$ given by Equation (4.13) with the function $c(t)$ given as

$$c(t) = \begin{cases} 2 \cdot \max\left\{ \left[\left(1 - \frac{\psi t^2}{t_k}\right)^2 + \left(\frac{-\psi t}{t_k}\right)^2 + \left(\frac{-\psi}{t_k}\right)\lambda \right], \left[t^2 \left(1 - \frac{\psi t}{4} - \frac{t}{2t_k}\right)^2 + \left(1 - \frac{t}{t_k} - \frac{\psi t}{2}\right)^2 \right. \right. \\ \qquad \left. \left. + \frac{1}{4}\left(\psi + \frac{2}{t_k}\right)^2 \lambda \right] \right\}, \quad t \in [0, t_k) \\ \kappa \left[\left(1 - \psi\left(\frac{t_k}{2} + (t - t_k)\right)\right)^2 + \psi^2 \right], \quad t \in [t_k, t_k + t_m) \\ \kappa \left[\left(1 - \psi\left(t - \frac{1}{2}t_k - \frac{(t - t_k - t_m)^2}{2t_n}\right)\right)^2 + \psi^2 \left(\frac{t - t_k - t_m}{t_n} - 1\right)^2 + \frac{\psi^2}{t_n^2}\lambda \right], \\ \qquad t \in [t_k + t_m, t_k + t_m + t_n) \\ 0 \qquad\qquad\qquad\qquad\qquad\qquad \textit{otherwise}, \end{cases} \tag{4.16}$$

where the constants κ and ψ are defined as

$$\psi = \left(\frac{t_k}{2} + t_m + \frac{t_n}{2}\right)^{-1}, \qquad \kappa = \max\left\{2, \frac{t_k^2}{2}\right\}$$

□

Proof. The objective here, is to show that the growth function $c(t)$ from Proposition 2 will satisfy inequality (4.14) for all time instances $t \geq 0$. We begin by noting that the trajectory developed in Section 4.4.1 is sub-optimal in terms running costs (4.8) and therefore the costs associated with this trajectory can be treated as an upper bound on the costs for the optimal trajectory obtained from Algorithm 1. We, therefore, proceed to develop a bound function $c(t)$ for each part of the sub-optimal trajectory.

Part *I*

We begin by solving for the final velocity of trajectory I, $x_{i+3}(t_k)$ and the acceleration, $u_{I,i}$ for this part of the trajectory. Using the fact that

$$u_{I,i} = \frac{x_{i+3}(t_k) - x_{0,i+3}}{t_k}, \quad \text{and} \quad u_{III,i} = \frac{-x_{i+3}(t_k)}{t_n}$$

it follows from

$$x_i(t) = x_{0,1} + x_{0,i+3}t + 0.5\left(\frac{x_{i+3}(t_k) - x_{0,i+3}}{t_k}\right)t^2$$

that

$$x_i(t_k) = x_{0,1} + 0.5x_{0,i+3}t_k + 0.5x_{i+3}(t_k)t_k. \tag{4.17}$$

We also show that since in trajectory III,

$$x_i(t) = x_i(t_k + t_m) + x_{i+3}(t_k)(t - t_k - t_m) + 0.5\left(\frac{-x_{i+3}(t_k)}{t_n}\right)\cdot(t - t_k - t_m)^2$$

and that $x_i(t_k + t_m + t_n) = 0$

$$x_i(t_k + t_m) = -0.5x_{i+3}(t_k)t_n$$

And finally looking at trajectory II we determine that

$$x_i(t_k + t_m) = x_i(t_k) + x_{i+3}(t_k)t_m$$

which implies that

$$x_i(t_k) = -x_{i+3}(t_k)\left(t_m + 0.5t_n\right) \tag{4.18}$$

Now equating (4.17) and (4.18) we can solve for

$$x_{i+3}(t_k) = \frac{-\left(x_{0,i} + \frac{1}{2}x_{0,i+3}t_k\right)}{\left(\frac{1}{2}t_k + t_m + \frac{1}{2}t_n\right)} = -\psi\left(x_{0,i} + x_{0,i+3}\frac{t_k}{2}\right)$$
$$u_{I,i} = \frac{x_{i+3}(t_k) - x_{0,i+3}}{t_k} = -\frac{x_{0,i}\psi}{t_k} - \frac{x_{0,i+3}}{2}\left(\psi + \frac{2}{t_k}\right)$$

for $i \in \{1, 2\}$. using this expression for $u_{I,i}$, we can find expressions for $x_i(t)$ and $x_{i+3}(t)$, $i \in \{1, 2\}$, for $t \in [0, t_k)$ as

$$x_i(t) = x_{0,i}\left(1 - \frac{\psi t^2}{2t_k}\right) + x_{0,i+3}t\left(1 - \frac{\psi t}{4} - \frac{t}{2t_k}\right)$$
$$x_{i+3}(t) = -x_{0,i}\frac{\psi t}{t_k} + x_{0,i+3}\left(1 - \frac{t}{t_k} - \frac{1}{2}\psi t\right)$$

Note that these expressions are now entirely dependent on the initial state x_0 and the maneuver time. The associated costs with this trajectory will be

$$\ell(\mathbf{x}_{\mathbf{u}_{\mathbf{x}_0}}(t; \mathbf{x}_0), \mathbf{u}_{\mathbf{x}_0}(t)) = \sum_{i=1}^{2} q_i x_i(t)^2 + \sum_{i=4}^{5} q_i x_i(t)^2 + \sum_{i=1}^{2} r_i u_{I,i}^2$$

for $t \in [0, t_k)$. In the interest of simplifying the presentation of the rest of the proof, we use $\ell(\cdot)$ to denote $\ell(\mathbf{x}_{\mathbf{u}_{\mathbf{x}_0}}(t; \mathbf{x}_0), \mathbf{u}_{\mathbf{x}_0}(t))$.

Now, using the identity $(a + b)^2 \leq 2(a^2 + b^2)$ for $a, b \in \mathbb{R}$, an upper bound on the running costs $\ell(\cdot)$ can be obtained, for $t \in [0, t_k)$, as

$$\begin{aligned}\ell(\cdot) \leq & \sum_{i=1}^{2} 2q_i x_{0,i}^2 \left[\left(1 - \frac{\psi t^2}{t_k}\right)^2 + \left(\frac{-\psi t}{t_k}\right)^2 + \left(\frac{-\psi}{t_k}\right)^2 \lambda\right] \\ & + \sum_{i=4}^{5} 2q_i x_{0,i}^2 \left[t^2\left(1 - \frac{\psi t}{4} - \frac{t}{2t_k}\right)^2 + \left(1 - \frac{t}{t_k} - \frac{\psi t}{2}\right)^2 + \frac{1}{4}\left(\psi + \frac{2}{t_k}\right)^2 \lambda\right] \qquad \text{for } t \in [0, t_k).\end{aligned}$$

This leads to $c(t)$ for $t \in [0, t_k)$ given by Eq. (4.16).

Part II

The second part of the trajectory is with a constant velocity, i.e. $u_{II,i} = 0$ and $x_{i+3}(t) = x_{i+3}(t_k)$, $i \in \{1, 2\}$ for $t \in [t_k, t_k + t_m)$, where we have the position

$$x_i(t_k) = \left(x_{0,i} + x_{0,i+3}\frac{t_k}{2}\right)\left(1 - \frac{t_k\psi}{2}\right).$$

Thus, the position trajectory over $t \in [t_k, t_k + t_m)$ can be defined as

$$x_i(t) = \left(x_{0,i} + x_{0,i+3}\frac{t_k}{2}\right)\left(1 - \psi\left(\frac{t_k}{2} + (t - t_k)\right)\right)$$

In a similar fashion to the first part of the trajectory, we obtain the following estimate on the running costs for $t \in [t_k, t_k + t_m)$

$$\begin{aligned}\ell(\cdot) \leq \sum_{i=1}^{2} 2q_i x_{0,i}^2 \left[\left(1 - \psi\left(\frac{t_k}{2} + (t - t_k)\right)\right)^2 + \psi^2\right] \\ + \sum_{i=4}^{5} q_i x_{0,i}^2 \frac{t_k^2}{2}\left[\left(1 - \psi\left(\frac{t_k}{2} + (t - t_k)\right)\right)^2 + \psi^2\right]\end{aligned}$$

for $t \in [t_k, t_k + t_m)$, which results in $c(t)$ in Eq. (4.16) for the same time interval.

Part III

For the final part of the trajectory, we have a constant deceleration u_{III} for $t \in [t_k + t_m, t_k + t_m + t_n]$, where the system stops at $x^\star$. u_{III} is given by

$$u_{III,i} = \frac{-x_{i+3}(t_k)}{t_n} = \left(x_{0,i} + x_{0,i+3}\frac{t_k}{2}\right)\frac{\psi}{t_n} \tag{4.19}$$

for $i \in \{1, 2\}$. This input results in the following velocity and position

$$\begin{aligned}x_{i+3}(t) &= \left(x_{0,i} + x_{0,i+3}\frac{t_k}{2}\right)\psi\left(\frac{t - t_k - t_m}{t_n} - 1\right) \\ x_i(t) &= \left(x_{0,i} + x_{0,i+3}\frac{t_k}{2}\right)\left(1 - \psi\left(t - \frac{1}{2}t_k - \frac{(t - t_k - t_m)^2}{2t_n}\right)\right)\end{aligned}$$

Following similar to the first two parts, we have the estimate

$$\ell(\cdot) \leq \sum_{i=1}^{2} 2q_i x_{0,i}^2 \left[\left(1 - \psi \left(t - \frac{1}{2}t_k - \frac{(t - t_k - t_m)^2}{2t_n} \right) \right)^2 + \psi^2 \left(\frac{t - t_k - t_m}{t_n} - 1 \right)^2 + \frac{\psi^2}{t_n^2}\lambda \right]$$
$$+ \sum_{i=4}^{5} q_i x_{0,i} \frac{t_k^2}{2} \left[\left(1 - \psi \left(t - \frac{1}{2}t_k - \frac{(t - t_k - t_m)^2}{2t_n} \right) \right)^2 + \psi^2 \left(\frac{t - t_k - t_m}{t_n} - 1 \right)^2 + \frac{\psi^2}{t_n^2}\lambda \right]$$

for $t \in [t_k + t_m, t_k + t_m + t_n)$, which gives $c(t)$ for the same time interval, in Eq. (4.16). □

The calculated bound $B(t)$, given by Equation (4.13) is used via Theorem 2.3 to find a stabilizing horizon length $T > \delta$ for a control horizon $\delta > 0$ such that the closed-loop acceleration constrained control problem is asymptotically stable under the MPC scheme presented in Algorithm 1. This will be shown via numerical simulation in the next section.

4.5 Numerical Analysis

In this section, the bound $B(t)$, is used to determine a prediction horizon length T such that the closed-loop MPC algorithm is asymptotically stable. The resulting prediction horizon is then used in a closed-loop simulation where asymptotic stability conditions are verified.

4.5.1 Computation of a Stabilizing Prediction Horizon

The objective here is to find conditions under which the relaxed Lyapunov inequality (2.42) from Theorem 2.3 holds, this will ensure the asymptotic stability of the system under closed-loop MPC. To satisfy this inequality, we require that $\alpha_{T,\delta} > 0$. Given the bounds $B(\cdot)$ from Proposition 2, we find a minimum prediction horizon length $\hat{T}$, such that

$$\hat{T} = \min \left\{ T \in \mathbb{R}_{\geq \delta} : \alpha_{T,\delta} > 0 \right\}, \tag{4.20}$$

where $\alpha_{T,\delta}$ is calculated from Equation (2.41) by adopting the bounds $B(t)$, given by Equation (4.13).

We study the effects of varying parameters t_k, t_m, t_n and λ on the performance index $\alpha_{T,\delta}$. Fig. 4.3 shows the effect of changing t_k and λ, here we see that a shorter maneuver time t_k gives a smaller $\hat{T}$ and a smaller λ has the same effect. We note that changing the maneuver lengths t_m and t_n have little effect on the value of $\hat{T}$.

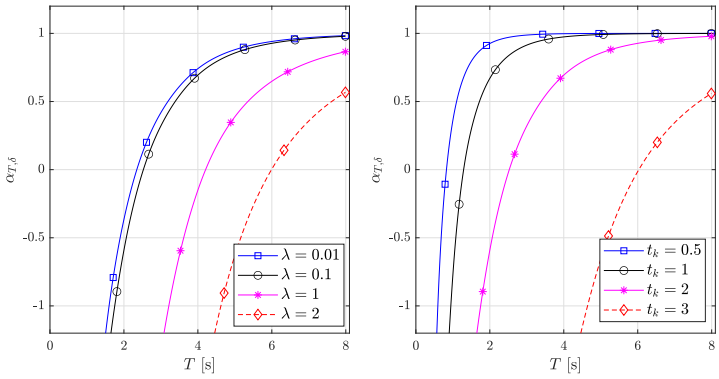

Figure 4.3: Value of $\alpha_{T,\delta}$ for various values of t_k (left) and λ (right), where the $t_m = t_n = 1$ s and $\delta = 0.1$ s.

4.5.2 Closed-loop Simulations

In this section, we show the numerical results of implementing Algorithm 1. Herein, the kinematic equation of the holonomic robot shown in (4.6) is used for state prediction. The acceleration constraint for this simulation was set to $\bar{u}_i^a = 1$ [m/s^2], $i \in \{1, 2\}$. The allowable states are set $\bar{x}_i = 1$m for $i \in \{1, 2\}$, $\bar{x}_i = 1$m/s for $i \in \{4, 5\}$. The simulations are conducted until condition $||\mathbf{x}(t_f) - \mathbf{x}^\star|| \leq 0.01$m is met, where $\mathbf{x}(t_f)$ is the state measurement at time t_f and $\mathbf{x}^\star = (0, 0, 0)^\top$ is the reference set point.

The simulations are performed from 4 different initial conditions, $\mathbf{x}_0 \in \mathcal{A}$. Simulations are conducted using MATLAB; OCP (4.10) is set up symbolically using CasADi toolbox [5] and the multiple-shooting discretization method [55]. The resulting nonlinear programming problem is solved with the Interior-Point method implemented in IPOPT [93]. The running costs are defined by (4.8) with $\lambda = 0.01$.

In order to determine the prediction horizon length which guarantees asymptotic stability for all $\mathbf{x}_0 \in \mathcal{A}$, maneuver times t_k, t_m and t_n are calculated according to Equation (4.15) for $\mathbf{x}_0$ as far as possible from $\mathbf{x}^\star$ in $\mathcal{A}$. The resulting values are

$$t_k = 2\text{s}, \qquad t_m = 0.5\text{s} \quad \text{and} \quad t_n = 1\text{s}$$

Moreover, using Equation (4.20), we obtain $\hat{T} = 2.45$ s. The resulting closed-loop paths can be seen in Fig. 4.4. As can be seen, the robot position converged to the reference $\mathbf{x}^\star$ for all the considered initial conditions. Additionally, the closed-loop paths do not leave the admissible set $\mathcal{A}$ presented in 4.2.3.

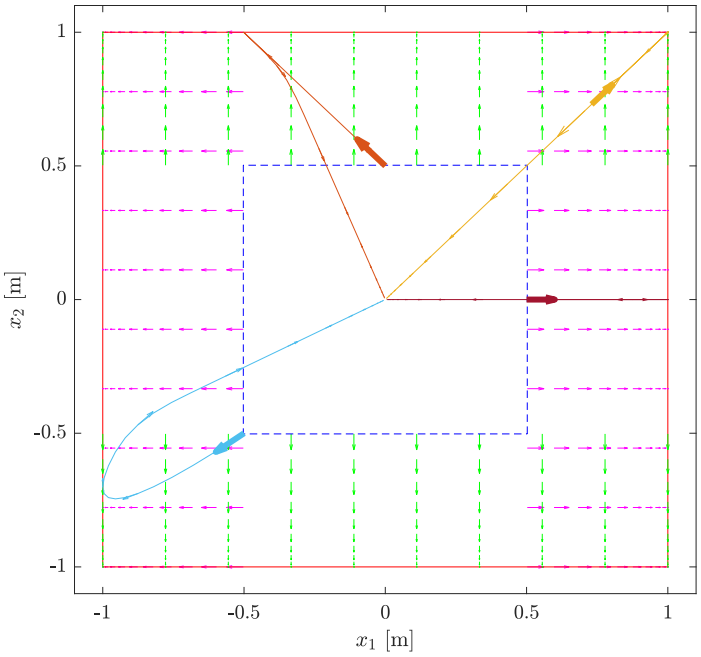

Figure 4.4: MPC closed-loop paths with initial states highlighted with bold arrows representing the overall initial speed of the robot. The paths are projected on the admissible set $\mathcal{A}$ shown in Fig. 4.1.

The purpose of constraining the acceleration is to generate velocity commands which produce a smoother motion. In Fig. 4.5 (left) we see that we we obtain smooth velocity profiles for the system. In Fig. 4.5 we also verify that the input constraints, which in this case is acceleration, are not violated.

Now that we have shown our constraints hold in simulation, it is also important to verify that the system is asymptotically stable. In Section 4.4, we've developed conditions for which Theorem 2.3 holds, to verify this is true in simulation, we look at the value function and ensure that the relaxed Lyapunov inequality (2.42) holds. In Fig. 4.6 we see that the value function is indeed monotonically decreasing for all cases shown in Fig. 4.4 and therefore Inequality (2.42) is verified.

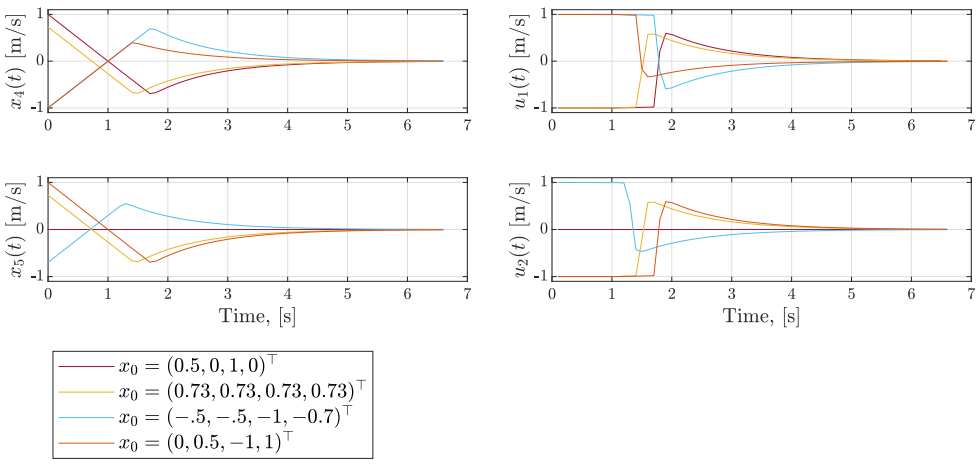

Figure 4.5: Left: Velocity profile and Right: acceleration profiles for the simulated trajectories in Fig. 4.4.

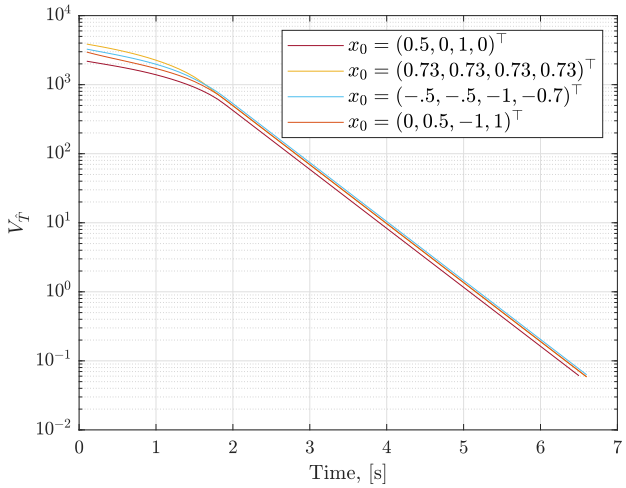

Figure 4.6: Time evolution of the value function $V_{\hat{T}}$ for the trajectories presented in Fig. 4.4.

4.6 Conclusion

In this chapter, the regulation of holonomic mobile robots with constraints on the acceleration is studied. The control objective is achieved using MPC without stabilizing constraints or costs. Herein, an admissible set of initial conditions for the considered system is derived first using the theory of barriers. Then, the closed-loop asymptotic stability under MPC is rigorously proven by verifying the cost controllability of the MPC controller. This is accomplished by developing a growth function bounding the MPC value function from an open-loop time scaled maneuver. With this growth function, a stabilizing prediction horizon is determined. We verified the theoretical results by several numerical simulations.
In the future, this work will be extended to the case where the orientation of the holonomic mobile robot is not constrained to zero values.

Chapter 5

Model Predictive Path Following Control

5.1 Introduction

In this chapter we apply a model predictive path following controller to tackle the path following problem presented in Section 2.4. The solution of the path-following control problem using MPC is introduced in [33] for constrained nonlinear systems. The resulting control scheme, which we use in this chapter, is referred to as MPFC.

MPFC is used for holonomic mobile robots in [49]. Here, a decoupled version of the kinematic model, where orientation and heading angles are considered separately, is used; an MPFC algorithm is then developed for the linearized model. Using the idea of a virtual robot, which perfectly follows the desired path, MPFC running costs penalize the deviation of the controlled robot from the virtual robot. The authors build on this paper in [47], where they solve the same problem for the nonlinear model. In these studies, terminal constraints and costs are utilized to ensure the closed-loop asymptotic stability.

In this chapter, we develop a discrete-time MPFC controller [54] for holonomic mobile robots. Rather than using terminal constraints or costs to guarantee the closed-loop asymptotic stability of the MPFC scheme, we use the concept of cost controllability similar to [68], which is based on the analysis developed in Section 2.2.2 and the generalization in [21]. The cost controllability method does not employ terminal constraints or penalties and is based on finding a growth function of the MPFC value function. The growth function is then used to compute a stabilizing prediction horizon and a performance index of

the finite-horizon MPFC algorithm in relation to the infinite-horizon MPFC. The path-following control of holonomic mobile robots presented here is an extension to the work in Chapter 3, in which the analysis is dedicated to the point-stabilization control problem.

The rest of the chapter is organized as follows: in Section 5.2, we recall the exact discrete-time kinematic model and the path-following control problem as they apply to holonomic mobile robots. This is followed by formulating the MPFC scheme. In Section 5.3, we revisit the stability results shown in Section 2.2.2. We follow this, in Section 5.4, with the derivation of a growth bound for the MPFC value function for a holonomic mobile robot, which allows us to compute a stabilizing prediction horizon. In Section 5.5, we show results of our simulations, study the effect of the parameters in the MPFC algorithm and present a path following formulation for a series of via points.

5.2 Problem Formulation

In this section, we consider the discrete time kinematic model for a holonomic robot (2.47) and the path following problem presented in Problem 1, Section 2.4.. We then introduce a model predictive path following control (MPFC) algorithm without terminal constraints or costs.

The compact and convex state constraint set $\mathbf{x}$ is defined as

$$X := [\underline{x}_1, \bar{x}_1] \times [\underline{x}_2, \bar{x}_2] \times S,$$

for $\underline{x}_1, \underline{x}_2 \leq 0$ and $\bar{x}_1, \bar{x}_2 \geq 0$. S is the compact set of angles on circle. Moreover, the saturation limits on the control inputs is defined by the set

$$U := [-\bar{u}_1, \bar{u}_1] \times [-\bar{u}_2, \bar{u}_2] \times [-\bar{u}_3, \bar{u}_3] \tag{5.1}$$

with $\bar{u}_1, \bar{u}_2, \bar{u}_3 > 0$.

5.2.1 Model Predictive Path-Following Control (MPFC)

Model predictive control (MPC) is proposed in [33] to tackle the path-following control Problem 1; we apply this control scheme here for holonomic mobile robots. For output path-following we refer to [32, 35]. To this end, we consider running costs $\ell : \mathbb{R}^4 \times \mathbb{R}^4 \to \mathbb{R}_{\geq 0}$ encoding the control objective and satisfying

$$\ell(\mathbf{z}^\star, 0) = 0 \quad \text{and} \quad \inf_{w \in W} \ell(\mathbf{z}, \mathbf{w}) > 0 \quad \forall \mathbf{z} \in Z \setminus \mathbf{z}^\star, \tag{5.2}$$

where $\mathbf{z}^\star$ is the end point of the path at which system (2.52) should be stabilized; $\mathbf{z}^\star$ and is given by

$$\mathbf{z}^\star := \left(\mathbf{p}(0)^\top, 0\right)^\top \tag{5.3}$$

We choose ℓ as a quadratic function as

$$\ell(\mathbf{z}, \mathbf{w}) = \left\| \begin{matrix} \mathbf{x} - \mathbf{p}(\theta) \\ \theta \end{matrix} \right\|_Q^2 + \left\| \begin{matrix} \mathbf{u} - \mathbf{u}_{ref}(\theta, v) \\ v \end{matrix} \right\|_R^2, \tag{5.4}$$

where $Q = \mathrm{diag}(q_1, q_2, q_3, \hat{q})$ and $R = \mathrm{diag}(r_1, r_2, r_3, \hat{r})$ are positive definite weighing matrices. $\mathbf{u}_{ref} \in U$ is the reference control input required to exactly follow the path $\mathcal{P}$. Since the considered system given by Equation (2.47) is affine in the control $\mathbf{u}$, the control $\mathbf{u}_{ref}$ can be obtained via

$$\mathbf{u}_{ref}(\theta, v) = \mathcal{R}_e^{-1}(\tau, x_3, u_3) \left(\left(\mathbf{p}(\theta^+) - \mathbf{p}(\theta)\right), \right. \tag{5.5}$$

where

$$\mathbf{p}(\theta^+) - \mathbf{p}(\theta) = \int_0^\tau \dot{\mathbf{p}}(\theta) dt = \frac{\partial \mathbf{p}(\theta)}{\partial \theta} \dot{\theta} \int_0^\tau dt = \frac{\partial \mathbf{p}(\theta)}{\partial \theta} v\tau,$$

where $\dot{\theta} = (\theta^+ - \theta)/\tau = v$ is used, see the path dynamics model given by Equation (2.50). Since $\mathbf{u}_{ref}$ is a function of v, the largest admissible virtual control defining the set V by

$$\bar{v} := \max \left\{ v \in \mathbb{R}_{\geq 0} | \mathbf{u}_{ref}(\theta, v) \in U, \forall \theta \in [\bar{\theta}, 0] \right\}. \tag{5.6}$$

At the sampling time $k \in \mathbb{N}_0$, the objective function to be minimized by the MPFC scheme is defined as the summation of running costs over the prediction horizon, $N \in \mathbb{N}_{\geq 2}$, as

$$J_N(\mathbf{z}_k, w) := \sum_{j=0}^{N} \ell(z_w(j, \mathbf{z}_k), \mathbf{w}(j)). \tag{5.7}$$

The MPFC scheme is based on repeatedly solving the OCP

$$\begin{aligned} V_N(\mathbf{z}_k) &= \min_{w \in W^N} J_N(\mathbf{z}_k, w) \\ \text{s.t. } \mathbf{z}(k) &= \mathbf{z}_k, \\ \mathbf{z}(j+1) &= \mathbf{z}(j) + \mathcal{R}_{aug}(\cdot)\mathbf{w}(j) \quad \forall j \in \{0, 1, \ldots, N-1\}, \\ \mathbf{z}(j) &\in Z \quad \forall j \in \{0, 1, \ldots, N\}, \\ \mathbf{w}(j) &\in W \quad \forall j \in \{0, 1, \ldots, N-1\}, \end{aligned} \tag{5.8}$$

where the initial state $\mathbf{z}_0$ is given by

$$\mathbf{z}_0 = (x_{1,0}, x_{2,0}, x_{3,0}, \theta_0)^\top \in Z. \tag{5.9}$$

Following [34], the initial value of the path parameter θ_0 is chosen such the initial reference point on $\mathcal{P}$, i.e. $\mathbf{p}(\theta_0)$, is the closest to $\mathbf{x}_0$, where $\mathbf{x}_0 = (x_{1,0}, x_{2,0}, x_{3,0})^\top$, i.e θ_0 is computed by

$$\theta_0 = \underset{\theta_0 \in [\bar{\theta}, 0]}{\arg\min} \|\mathbf{x}_0 - \mathbf{p}(\theta_0)\|.$$

The solution of OCP (5.8) is the optimal control sequence $w^\star = \mathbf{w}^\star(\cdot, \mathbf{x}_k) \in \mathcal{W}^N(\mathbf{z}_k)$. Moreover, $V_N(\mathbf{z}_k) = J_N(\mathbf{z}_k, w^\star)$ is the optimal value function. The MPFC feedback control law denoted by $\mu_N : Z \to W$ is given by $\mu_N(k, \mathbf{z}_k) = \mathbf{w}^\star(1, \mathbf{z}_k)$. Here, the feedback control applied to the robot is given by

$$\mathbf{u}(k) = \begin{pmatrix} w_1^\star(1, \mathbf{z}_k) & w_2^\star(1, \mathbf{z}_k) & w_3^\star(1, \mathbf{z}_k) \end{pmatrix} \tag{5.10}$$

while the feedback control to the path dynamics (2.50) is applied as $v(k) = w_4^\star(1, \mathbf{z}_k)$. The closed-loop trajectory resulting from applying $\mu(\cdot; \mathbf{z}_k)$ to the augmented system (2.52) is denoted by $z_\mu(\cdot, \mathbf{z}_0)$ is generated by.

$$z_\mu(k+1; \mathbf{z}_0) = f_{aug}\left(z_\mu(k; \mathbf{z}_0), \mu_N(k; z_\mu(k; \mathbf{z}_0))\right),$$

where $z_\mu(0; \mathbf{z}_0) = \mathbf{z}_0$.

Existence of an admissible control sequence $\mathbf{w}^\star$ of OCP (5.8) can be inferred from the continuity of the cost function and the compactness of the sets Z and W via Weierstrass theorem [14]. Moreover, the recursive feasibility of the MPFC closed-loop holds since $0_{\mathbb{R}^4} \in W$ and, thus, $\mathcal{W}^N(\mathbf{z}_k) \neq \emptyset$. However, the asymptotic stability of the closed-loop system does not trivially hold, especially that no terminal constraints or costs are employed in OCP (5.8), see, e.g. [12, 77].

5.3 Stability and Performance Bounds

We recall MPC stability results presented in Theorem 2.2, we show that, if this stability result holds for the MPFC controller, for all initial conditions, the solution of Problem 1 is ensured. We use Theorem 2.2 to show that the closed-loop asymptotic stability of the MPFC scheme for the augmented system (2.52) can be concluded for an initial condition $\mathbf{z}_0 \in Z$. We note that for the path following problem we replace $\mathbf{x}, \mathbf{u}$ and f_d with their augmented system counterparts $\mathbf{z}, \mathbf{w}$ and f_{aug}.

Proposition 3. *We let the prediction horizon N, and the weights Q and R of the running costs (5.4) be selected such that the conditions of Theorem 2.2 are satisfied. Then, the MPFC control input (5.10) solves the path following Problem 1.* □

Proof. The chosen running costs (5.4) implies that, as $\mathbf{z}$ goes to $\mathbf{z}^\star$, the robot state converges to the path and θ converges to 0. Therefore, as the recursive feasibility of OCP (5.8) holds, stabilizing $\mathbf{z}$ at $\mathbf{z}^\star$ in an admissible way implies solving Problem 1. □

In essence, by constructing a growth bound sequence γ_i, $i \in \mathbb{N}$ of the MPFC value function satisfying Condition (2.35), we can find a stabilizing horizon length N ensuring that $\alpha_N > 0$ and thus the conditions necessary for Theorem 2.2 hold. In the following section, we show a method for constructing the bound γ_i for the MPFC scheme of the holonomic mobile robot.

5.4 Growth Bound γ_i

In this section, a bounded sequence γ_i, $i \in \mathbb{N}_{\geq 2}$ satisfying Condition (2.35) is computed. As shown in [96], the bounds γ_i can be obtained using a summable sequence $c(j) \geq 0$, $j \in \mathbb{N}_0$, i.e. $\sum_{j=0}^{\infty} c(j) < \infty$, by

$$\gamma_i = \sum_{j=0}^{i-1} c(j), \qquad i \in \mathbb{N}_{\geq 2},$$

where the sequence $c(j)$ is calculated such that the inequality

$$\frac{\ell(z_{w_{\mathbf{z}_0}}(j; \mathbf{z}_0), \mathbf{w}_{\mathbf{z}_0}(j))}{\ell^\star(\mathbf{z}_0)} \leq c(j) \quad \forall j \in \mathbb{N}_0 \text{ and } \mathbf{z}_0 \in Z \tag{5.11}$$

holds for an admissible sequence of control actions $w_{\mathbf{z}_0} = \mathbf{w}_{\mathbf{z}_0}(j)$, $j \in \mathbb{N}_0$, which fulfills the path-following control Problem 1. Moreover, as the control sequence $\mathbf{w}_{\mathbf{z}_0}(j) = 0_{\mathbb{R}^3}$, $j \in \mathbb{N}_0$ is admissible for all $j \geq 0$, Condition (2.35) holds also for $\gamma_i = i$. In conclusion, the bounds γ_i are calculated via

$$\gamma_i = \min \left\{ i, \sum_{j=0}^{i-1} c(j) \right\}, \qquad i \in \mathbb{N}_{\geq 2}, \tag{5.12}$$

see [96] for more details. The following proposition shows a method for obtaining a possible sequence $c(\cdot)$, which satisfies Inequality (5.11) and used to calculate γ_i from Equation (5.12).

Proposition 4. *Let the weighting matrices Q and R of the running costs (5.4) be chosen such that*

$$q_1 = q_2,\ r_1 = r_2,\ r_1 \leq \lambda q_1/2,\ r_3 \leq \lambda q_1 \quad \textit{and } \hat{r} \leq \hat{\lambda}\hat{q}. \tag{5.13}$$

Then, Inequality (2.35) holds with γ_i given by Equation (5.12) with the sequence $c(j)$ given by

$$c(j) = \begin{cases} \max\left\{1, \left(\frac{k_m - j}{k_m}\right)^2 + \frac{\lambda}{\tau^2 k_m^2}\right\} & j \in \{0, ..., k_m - 1\} \\ \left(\frac{k_m + k_n - j}{k_n}\right)^2 + \frac{\hat{\lambda}}{\tau^2 k_n^2} & j \in \{k_m, ..., k_m + k_n - 1\} \\ 0 & \textit{otherwise} \end{cases} \tag{5.14}$$

for $k_m \in \mathbb{N}_{\geq 2}$ and $k_n \in \mathbb{N}_{\geq 2}$ given by

$$k_m = \max\left\{k_{m,tran}, k_{m,rot}\right\}, \qquad k_n = \frac{|\bar{\theta}|}{\tau \bar{v}}, \tag{5.15}$$

where

$$\begin{aligned} k_{m,tran} &= \left\lceil \frac{d_{max}}{\tau \cdot \min\{\bar{u}_1, \bar{u}_2\}} \right\rceil \quad \textit{where} \\ d_{max} &= \max_{\theta \in [\bar{\theta}, 0], \mathbf{x} \in X} \left\| \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - \begin{pmatrix} p_1(\theta) \\ p_2(\theta) \end{pmatrix} \right\| \quad \textit{and} \\ k_{m,rot} &= \left\lceil \frac{\pi}{\tau \bar{u}_3} \right\rceil. \end{aligned}$$

□

Proof. The growth bound γ_i calculation is based on designing an open-loop control maneuver, fulfilling the path-following control Problem 1, from which the running costs ℓ can be evaluated and the sequence $c(\cdot)$ from Inequality (5.11) can be estimated [67, 96]. Thus, for the initial condition $\mathbf{z}_0 = (x_{0,1}, x_{0,2}, x_{0,3}, \theta_0)^\top \in Z$, we design an open-loop control maneuver that steers the robot to the end point of the reference path $\mathcal{P}$, i.e. $\mathbf{z}^\star$ as follows: first, the robot is steered from its initial condition $\mathbf{x}_0$ to the nearest point on the path $\mathcal{P}$, i.e. $\mathbf{p}(\theta_0)$. This part can be designed as a straight-line trajectory with uniform linear and angular velocities, see Chapter 3. Once the robot is on the path $\mathcal{P}$, the second part of the maneuver is to steer the robot such that it follows exactly the reference path until the path end point

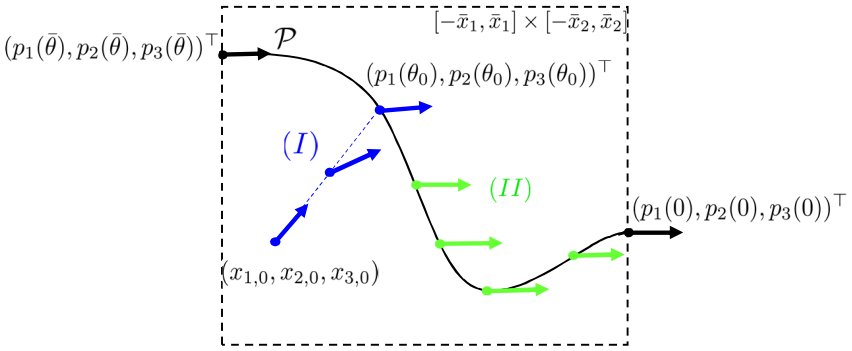

Figure 5.1: Two-steps proposed maneuver for any initial condition $\mathbf{z}_0$.

$\mathbf{z}^\star$. The two-steps maneuver is illustrated in Fig. 5.1. The input sequence $w_{\mathbf{z}_0}$, designed to follow the chosen maneuver yields sub-optimal running costs $\ell(z_{w_{\mathbf{z}_0}}(j; \mathbf{z}_0), \mathbf{w}_{\mathbf{z}_0}(j))$, $j \in \mathbb{N}_0$ for the path following problem. This ensures that Inequality (5.11) holds when the *optimal* control input μ_N is applied. We remark that the minimized cost $\ell^\star(\mathbf{z}_0)$ of Inequality (2.35) is given by

$$\ell^\star(\mathbf{z}_0) = \sum_{i=1}^{3} q_i \left(x_{0,i} - p_i(\theta_0)\right)^2 + \hat{q}\theta_0^2. \tag{5.16}$$

Let us consider the first part of the maneuver; here, this part is achieved via a straight-line trajectory between the points $\mathbf{x}_0$ and the nearest point on the path $\mathbf{p}(\theta_0)$, i.e.

$$\mathbf{x}(j) = \mathbf{x}_0 + \frac{j}{k_m}\left(\mathbf{p}(\theta_0) - \mathbf{x}_0\right), \quad j \in \{0, \ldots, k_m\},$$

where $k_m \in \mathbb{N}_{\geq 2}$ is the required number of steps of the maneuver given by Equation (5.15). For this part of the maneuver, the virtual state θ is kept stationary, i.e. $\theta(j) = \theta_0$ for all $j \in \{0, ..., k_m\}$ and $v = 0$ for all $j \in \{0, ..., k_m - 1\}$; this ensures that the reference control given by Equation (5.5) is $\mathbf{u}_{ref}(\theta, v) = 0$. Therefore, the control input required to follow this trajectory is

$$\mathbf{w}_{\mathbf{z}_0}(j) = \begin{pmatrix} \mathbf{u}_{\mathbf{x}_0}(j) & 0 \end{pmatrix}^\top, \quad j \in \{0, ..., k_m - 1\},$$

where

$$\mathbf{u}_{\mathbf{x}_0}(j) = \mathcal{R}_e(\tau, x_3(j), u_3(j))^{-1} \begin{pmatrix} \mathbf{e}_0 \\ k_m \end{pmatrix}, \tag{5.17}$$

with $\mathbf{e}_0$ defined as $\mathbf{e}_0 = \mathbf{p}(\theta_0) - \mathbf{x}_0$. Here, the matrix $\mathcal{R}_e(\tau, x_3(j), u_3(j))^{-1}$ is given by

$$\mathcal{R}_e(\tau, x_3, u_3)^{-1} = \frac{u_3}{4} \begin{pmatrix} \frac{\sin(x_3+\tau u_3)-\sin(x_3)}{\sin^2(0.5\tau u_3)} & \frac{\cos(x_3+\tau u_3)-\cos(x_3)}{\sin^2(0.5\tau u_3)} & 0 \\ \frac{\cos(x_3)-\cos(x_3+\tau u_3)}{\sin^2(0.5\tau u_3)} & \frac{\sin(x_3+\tau u_3)-\sin(x_3)}{\sin^2(0.5\tau u_3)} & 0 \\ 0 & 0 & \frac{4}{\tau u_3} \end{pmatrix},$$

where the subscript $\mathbf{x}_0$ is dropped from all the elements of $\mathbf{u}_{\mathbf{x}_0}$ to simplify the presentation; we keep this simplification for the rest of the proof. Moreover, k_m is chosen sufficiently large such that $\mathbf{u}_{\mathbf{x}_0}(j) \in U$, $j \in \{1, \ldots, k_m - 1\}$ is ensured for all $\mathbf{x}_0 \in X$.

From Equation (5.17), we obtain the control $u_3(j)$ as

$$u_3(j) = \frac{e_{0,3}}{\tau k_m}, \qquad \forall j \in \{0, ..., k_m - 1\} \tag{5.18}$$

Using the formulation of the running costs in Equation (5.4), and the first two assumptions in (5.13), we obtain

$$\ell(z_{w_{\mathbf{z}_0}}(j; \mathbf{z}_0), \mathbf{w}_{\mathbf{z}_0}(j)) = \hat{q}\theta_0^2 + \left(\frac{k_m - j}{k_m}\right)^2 \sum_{i=1}^{3} q_i e_{0,i}^2 + r_1 \left(u_1(j)^2 + u_2(j)^2\right) + r_3 \frac{e_{0,3}^2}{\tau^2 k_m^2},$$

for $j \in \{0, \ldots, k_m - 1\}$. Following a similar procedure to Chapter 3 [Proposition 4], the term $r_1 \left(u_1(j)^2 + u_2(j)^2\right)$ can be calculated by squaring then adding the first two equations in (5.17); thus, we have

$$r_1 \left(u_1(j)^2 + u_2(j)^2\right) = \frac{r_1 u_3(j)^2}{4 \sin^2(0.5\tau u_3(j))} \left(\frac{e_{0,1}^2}{k_m^2} + \frac{e_{0,2}^2}{k_m^2}\right).$$

Taylor series expansion of $\sin^2(0.5\tau u_3(j))$ leads to

$$\sin^2(0.5\tau u_3(j)) \geq \frac{\tau^2 u_3(j)^2}{4} \left(1 - \frac{\tau^2 u_3(j)^2}{12}\right) > 0,$$

which is valid for all $u_3(j)^2 := \left(e_{0,3}/\tau k_m\right)^2 \leq \left(\pi/\tau k_m\right)^2$. As a result, we have the estimate

$$r_1\left(u_1(j)^2 + u_2(j)^2\right) \leq \frac{12 \cdot r_1}{12 - \left(\frac{e_{0,3}^2}{k_m^2}\right)} \left(\frac{e_{0,1}^2}{\tau^2 k_m^2} + \frac{e_{0,2}^2}{\tau^2 k_m^2}\right) \leq \frac{12 \cdot r_1}{12 - \left(\frac{\pi^2}{k_m^2}\right)} \left(\frac{e_{0,1}^2}{\tau^2 k_m^2} + \frac{e_{0,2}^2}{\tau^2 k_m^2}\right). \tag{5.19}$$

The term $12/\left(12 - \frac{\pi^2}{k_m^2}\right)$ has an upper bound of ≈ 1.3 for $k_m \geq 2$. Choosing the weight r_1 to satisfy Assumption (5.13), leads to

$$r_1 \left(u_1(j)^2 + u_2(j)^2\right) \leq \lambda \cdot q_1 \left(\frac{e_{0,1}^2}{\tau^2 k_m^2} + \frac{e_{0,2}^2}{\tau^2 k_m^2}\right). \tag{5.20}$$

We remark that the term $u_3(j)^2/4\sin^2(0.5\tau u_3(j))$ reduces to $1/\tau^2$ for $u_3(j)=0$ as

$$\lim_{u_3(j)\to 0}\frac{u_3(j)^2}{4\sin^2(0.5\tau u_3(j))}=\frac{1}{\tau^2};$$

thus, the estimate (5.20) holds also for $u_3(j)=0$. Now, choosing r_3 satisfying Assumption (5.13), the running costs $\ell(z_{w_{\mathbf{z}_0}}(j;\mathbf{z}_0),\mathbf{w}_{\mathbf{z}_0}(j))$, for $j\in\{0,\dots,k_m-1\}$, can be estimated by

$$\ell(z_{w_{\mathbf{z}_0}}(j;\mathbf{z}_0),\mathbf{w}_{\mathbf{z}_0}(j))\leq \hat{q}\theta_0^2+\left[\left(\frac{k_m-j}{k_m}\right)^2+\frac{\lambda}{\tau^2k_m^2}\right]\sum_{i=1}^{3}q_ie_{0,i}^2.$$

As $\ell^\star(\mathbf{z}_0)$ of Equation (5.16) can be written as $\ell^\star(\mathbf{z}_0)=\sum_{i=1}^{3}q_ie_{0,i}^2+\hat{q}\theta_0^2$, Inequality (5.11) is ensured with $c(j)$ given by Equation (5.14) for $j\in\{0,\dots,k_m-1\}$.

Now, we consider the second part of the maneuver, where the robot moves exactly along the path until it reaches its end point $\mathbf{z}^\star$ at time step k_m+k_n for $k_n\in\mathbb{N}_{\geq 2}$ given by Equation (5.15). This is achieved via applying the control

$$\mathbf{w}_{\mathbf{z}_0}(j)=\begin{pmatrix}\mathbf{u}_{ref}(\theta(j),v(j)) & v(j)\end{pmatrix}^\top, \tag{5.21}$$

for $j\in\{k_m,\dots,k_m+k_n-1\}$, where $v(j)=\frac{-\theta_0}{\tau k_n}$, i.e. the path parameter θ is steered from $\theta(k_m)=\theta_0$ to $\theta(k_m+k_n)=0$ while the robot follows the reference path exactly via the control $\mathbf{u}_{ref}(\theta(\cdot),v(\cdot))$. Here, k_n is chosen large enough such that $v(\cdot)\in V$ is ensured. Moreover, applying $\mathbf{u}_{ref}(\theta(\cdot),v(\cdot))$, calculated via Equation (5.5), to the robot ensures that it follows the path $\mathcal{P}$ exactly. Therefore, the running costs given be Equation (5.4) for this part of the maneuver becomes

$$\ell(z_{w_{\mathbf{z}_0}}(j;\mathbf{z}_0),\mathbf{w}_{\mathbf{z}_0}(j))=\hat{q}\left(\frac{\theta_0(k_m+k_n-j)}{k_n}\right)^2+\hat{r}\left(\frac{\theta_0}{\tau k_n}\right)^2,$$

for $j\in\{k_m,\dots,k_m+k_n-1\}$. Now, using the assumption on $\hat{r}$ in (5.13) and $\ell^\star(\mathbf{z}_0)$ given by Equation (5.16) lead to $c(j)$ given by Equation (5.14) for $j\in\{k_m,\dots,k_m+k_n-1\}$.

For $j\geq k_m+k_n$ we have $c(j)=0$. Therefore, γ_i given by (5.12) is bounded for the sequence $c(j)$ given by Equation (5.14). We remark that the sequence $c(j)$ is independent from the initial condition $\mathbf{z}_0$ and depends only on the maneuver times k_m and k_n, the sampling time τ, and the weight ratios λ and $\hat{\lambda}$. □

The calculated bounds γ_i, $i\in\mathbb{N}_{\geq 2}$ given by Equation (5.12) can be used via Theorem 2.2 to find a stabilizing horizon length $N\in\mathbb{N}_{\geq 2}$ such the closed-loop path-following control problem 1 is asymptotically stable under the MPFC scheme presented in Section 5.2.1. This will be shown in details in the following section.

5.5 Numerical Results

In this section, first, the bounds γ_i, $i \in \mathbb{N}_{\geq 2}$ are used to determine a prediction horizon length $N \in \mathbb{N}_{\geq 2}$ such that the closed-loop MPFC is asymptotically stable. The resulting prediction horizon is then used for closed-loop simulation while asymptotic stability conditions are verified.

5.5.1 Computation of a Stabilizing Prediction Horizon

Recalling Theorem 2.2, the key condition for asymptotic stability of the path following problem under MPFC without terminal constraints or costs is for the relaxed Lyapunov Inequality (2.37) to hold; this requires $\alpha_N > 0$. Therefore, a minimal stabilizing prediction horizon $\hat{N}$ can be calculated as

$$\hat{N} = \min \{N \in \mathbb{N}_{\geq 2} : \alpha_N > 0\},$$

where α_N is calculated by means of Equation (2.36) by adopting the bounds γ_i, $i \in \mathbb{N}_{\geq 2}$ given by Equation (5.12). For different values of k_m, k_n, λ and $\hat{\lambda}$, Fig. 5.2 shows the effect of these parameters on the performance index α_N and, thus, on the stabilizing prediction horizon $\hat{N}$. It is observed that altering k_n has no significant effect on α_N. However, larger values of k_m results in longer stabilizing prediction horizons. Changing the values of λ and $\hat{\lambda}$ have a significant impact on α_N. As shown in Fig. 5.2, we see that in both cases a smaller λ or $\hat{\lambda}$ gives a shorter stabilizing horizon $\hat{N}$. This means that by having a higher weight Q on the state error than R on the controls we get a shorter $\hat{N}$. Though we see limited gains from increasing Q once $\lambda < 0.1$ and $\hat{\lambda} < 0.01$. Finally, varying the time step τ has simply the inverse relationship to the results of varying λ or $\hat{\lambda}$ because τ^2 appears in the denominator of the same term in Equation (5.14).

5.5.2 Simulations

The simulations are conducted using MATLAB for a holonomic mobile robot with the exact discrete-time model (2.47). The OCP (5.8) is discretized over the prediction horizon using the multiple shooting method [55]. This produces a NLP which is created symbolically using CasADi toolbox [6]. The NLP is then solved using the interior point method (IPOPT) implemented in [93].

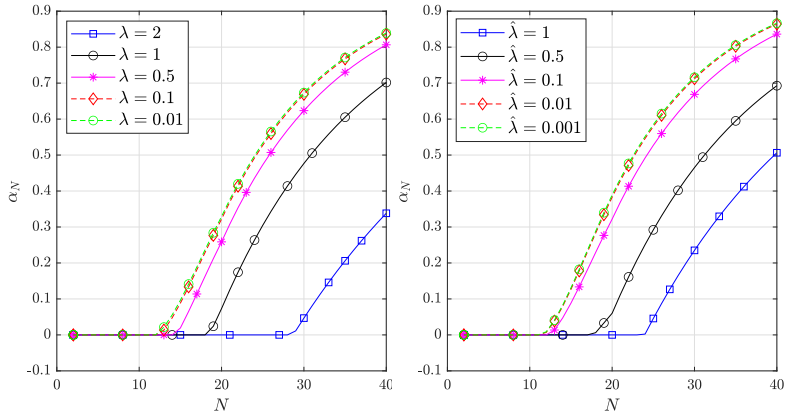

Figure 5.2: Left: Value of α_N with respect to the prediction horizon N, various values of λ are shown, with a fixed $\hat{\lambda} = 0.1$. Right: Value of α_N with respect to the prediction horizon N, various values of $\hat{\lambda}$ are shown, with a fixed $\lambda = 0.1$. The maneuver length is fixed in both cases at $k_m = 10$ and $k_n = 10$.

Path Following of a Sinusoidal Function

The first simulation is for a path $\mathcal{P}$ given by

$$\mathbf{p}(\theta) = [\theta, Asin(\omega\theta), 0]^\top, \qquad \theta \in [-15, 0], \tag{5.22}$$

where the amplitude $A = 10$ and the frequency $\omega = 0.5$. The state and control input constraints are set to

$$X := [-8, 2] \times [-2, 2] \times \mathbb{R}, \text{and } U := [-2, 2]^3,$$

respectively and the time step is set to $\tau = 0.2$. For this set of allowable states and control we can calculate the maneuver lengths, $k_m = 27$ and $k_n = 40$ according to Equation (5.15). By setting $\lambda = 0.1$ and $\hat{\lambda} = 0.1$, we get $\alpha_N > 0$ for all $N \geq 28$.

Additionally, by setting the weights $q_i = 200$ and $r_i = 10$ for $i = \{1, 2, 3\}$ and the weights $\hat{q} = 1$ and $\hat{r} = 0.1$ we ensure that the assumption (5.13) holds for the chosen λ and $\hat{\lambda}$. Fig. 5.3 shows the results of the simulation for the path following algorithm for 4 different initial conditions. It is observed that the robot in all cases converges to the nearest point on the path and remain there until it reaches the end of the prescribed path.

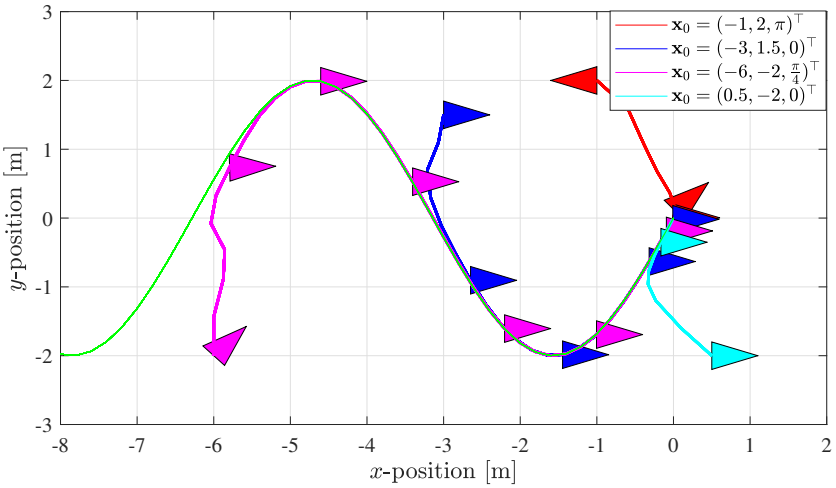

Figure 5.3: Path following simulation of four initial conditions for a sinusoidal path (shown in green). The orientation of the robot along each of the resulting paths is represented by a color-filled triangle.

As seen in Theorem 2.2, the MPFC value function V_N should satisfy the relaxed Lyapunov inequality (2.37) for asymptotic stability; thus, V_N should be strictly monotonically decreasing over the closed-loop trajectory. Fig. 5.4 (left) shows that this is satisfied for all initial conditions considered and the chosen prediction horizon. To verify the performance of our algorithm, we measure the difference between the simulated position of the robot and its desired reference trajectory. In Fig. 5.4 (right) we see that once the robot reaches desired trajectory, it is able to remain on the path within 1×10^{-2} m.

Path following given a set of via points

We further extend the application of the MPFC algorithm to the case where a set of arbitrary via points are to be followed by the mobile robot. This in practice is more useful, as it is more likely that the control objective for the robot be to follow a set of points rather than to follow a function. Using linear interpolation between the given points, we create a piece-wise continuous linear function, which using polynomial regression can be used to develop a continuous differentiable reference path $\mathcal{P}$. Given the following set of n points

$$\tilde{\mathbf{p}}_i = \begin{pmatrix} \tilde{p}_{i,1} & \tilde{p}_{i,2} & \tilde{p}_{i,3} \end{pmatrix}^\top, \qquad \forall i \in \{1, \dots, n\},$$

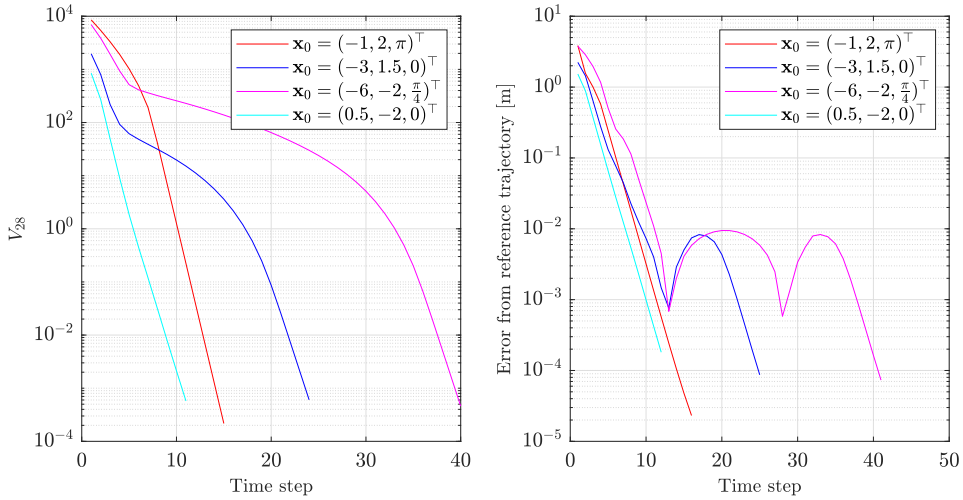

Figure 5.4: Left: Evolution of the value function V_N for each of the trajectories shown in Fig. 5.3, with prediction horizon $N = 28$. Right: Error between the simulated robot path and the reference trajectory for each starting conditions shown in Fig. 5.3.

the piece-wise continuous function for the desired path can be obtained as

$$\hat{p}_j(\theta) = \left(\frac{\tilde{p}_{i+1,j} - \tilde{p}_{i,j}}{\theta_{i+1} - \theta_i} \right) (\theta - \theta_i) + \tilde{p}_{i,j}, \ \forall j = \{1, 2, 3\}, \tag{5.23}$$

where $\theta_i < \theta \leq \theta_{i+1}$. θ will be a negative parametrization of the linear distance between points and is calculated as

$$\theta_i = \theta_{i+1} - \left\| \begin{pmatrix} \tilde{p}_{1,i+1} \\ \tilde{p}_{2,i+1} \end{pmatrix} - \begin{pmatrix} \tilde{p}_{1,i} \\ \tilde{p}_{2,i} \end{pmatrix} \right\| \quad \text{and } \theta_n = 0.$$

Since piece-wise linear functions are non-differentiable, we will use polynomial regression to transform the desired path to a differentiable polynomial. Here to ensure that the emphasis is to follow the desired path rather than pass through all the given points, additional points are added at an interval of $\Delta\theta$ to discretize the linear Function (5.23); essentially, filling in between the given points $\tilde{p}_i$. We recognize that the piecewise function $\hat{\mathbf{p}}(\theta)$ may not be well suited to be represented by a polynomial. Therefore we restrict the polynomial regression to a horizon of the next M points along the discretized path. To ensure that this horizon is long enough to accurately represent the desired trajectory path for the entire prediction horizon of the MPFC controller, we set $\Delta\theta = \bar{v}\tau$ and $M = N$.

The equation over the regression horizon at any time step can be found as

$$p_j(\theta) = c_m\theta^m + \ldots + c_1\theta + c_0, \quad \forall j \in \{1,2,3\} \tag{5.24}$$

where $p_j(\theta)$ is a polynomial of degree m. And the following is satisfied

$$\begin{bmatrix} \hat{p}_j(\theta_1) \\ \hat{p}_j(\theta_2) \\ \vdots \\ \hat{p}_j(\theta_M) \end{bmatrix} = \begin{bmatrix} 1 & \theta_1 & \theta_1^2 & \ldots & \theta_1^m \\ 1 & \theta_2 & \theta_2^2 & \ldots & \theta_2^m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \theta_M & \theta_M^2 & \ldots & \theta_M^m \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_M \end{bmatrix}$$

for $j = 1,2,3$ and $\theta_i = \theta_0 + i \cdot \Delta\theta$ for all $i \in \{1,2,\ldots M\}$, see, e.g. [30] for more details. Here, θ_0 refers to the current virtual state of the system along the path.

Fig. 5.5 shows a simulation for a set of via points given as the corners of a 4 [m]×4 [m] square with a desired orientation at each point of $p_3(\theta) = 0$. The piece-wise continuous linear function for the path is then calculated according to Equation (5.23) and is defined for $\theta \in [-16, 0]$. The state and control input constraints are set as

$$X := [-3,3] \times [-3,3] \times S, \text{and } U := [-2,2]^3,$$

respectively. A continuous and differentiable reference path function is calculated according to (5.24) for each time step, set to $\tau = 0.1$. For this set of allowable states and control we can calculate the maneuver lengths, $k_m = 34$ and $k_n = 40$ according to Equation (5.15). By setting $\lambda = 0.1$ and $\hat{\lambda} = 0.1$, we get $\alpha_N > 0$ for all $N \geq 35$. Additionally, we set the weights $q_i = 200$ and $r_i = 10$ for $i = \{1,2,3\}$ and the weights $\hat{q} = 10^3$ and $\hat{r} = 0.1$, for which assumption (5.13) holds.

We note here that the error between the position of the robot and the desired path shown in Fig. 5.5 (left) is in part due to approximating the path via the polynomial regression. The evolution of the value function V_N is shown in Fig. 5.5 (right), which confirms that the value function for is monotonically decreasing and, thus, verifies the relaxed Lyapunov inequality (2.37).

5.6 Conclusion

In this chapter the asymptotic stability of model predictive path following control (MPFC) for holonomic mobile robots has been proven without employing any terminal constraints

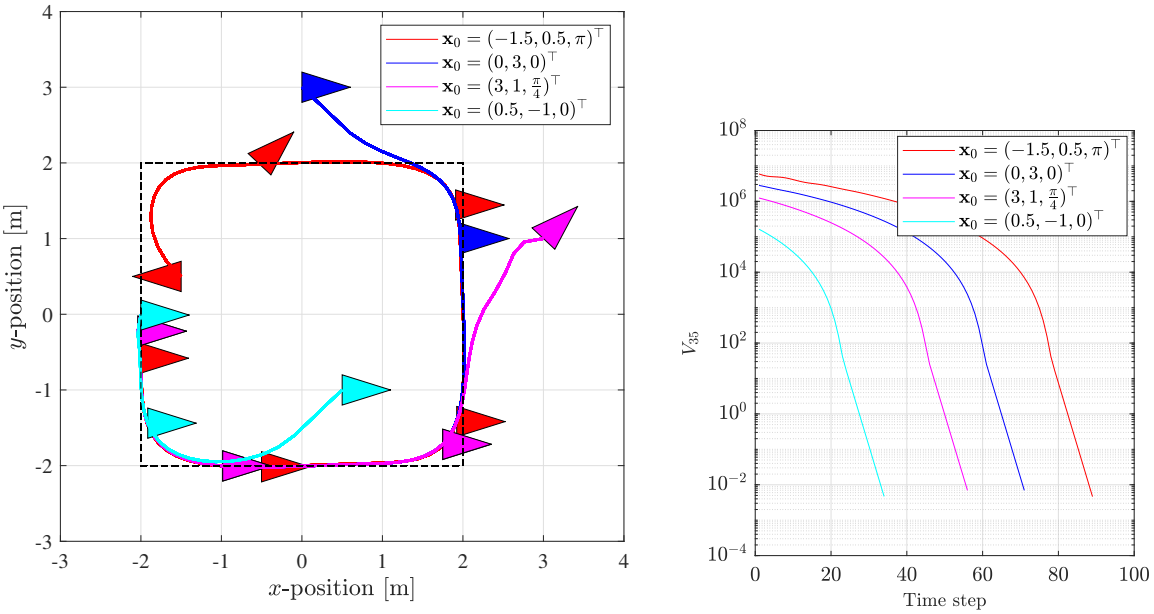

Figure 5.5: Left: Path following simulation with via points at each corner of a square, the desired path is shown with the dashed black line, with the final desired position at $\mathbf{p}(0) = (-5, 0, 0)^\top$. Right: Evolution of the value function for the model predictive path following controller for a set of via points shown on the left.

or costs. The analysis is based on verifying the cost controllability assumption in which a bound on the MPFC value function is derived and then used to find a stabilizing prediction horizon. The value function bound is obtained by constructing a sub-optimal maneuver achieving the control objective along which the running costs are estimated. Results are verified by numerical simulations in which two types of reference paths are considered, i.e. references given by explicit functions and by a set of via points.

Chapter 6

Joint Torque Minimization Model Predictive Control

6.1 Introduction

In this chapter, we develop a novel MPC algorithm with dual objectives, the first being to accurately reach a desired end effector position and the second to position the robot in a way which minimizes the actuator effort throughout the maneuver. We look at the actuator effort as the sum of the square all the joints torques in the robotic arm. Minimizing the torque in the joints, reduces stress on the actuator motors, which helps extend the life of the hardware.

Finding ways to minimize the joint torque in robotic arms through optimization is not a new idea, there have been several applications of this in literature. Some of the early works in joint torque minimization took advantage redundancy resolution to find torque optimal pose for robots. Any redundant robot, will have a set of joint velocities which exist in the null-space [42], this is a subspace of joint velocities which do not produce an end effector velocity. Using the null-space control, joint commands can be found to minimize the torque for a given trajectory without affecting the desired end effector pose. This work is used as the basis for [95] where null space control is used to minimized torque for cases with large external loading. This is done by moving in the direction of the gradient of a torque weight function with respect to joint command inputs projected into the null space. These null space joint commands, are added to the joint commands required to follow a desired trajectory.

An alternate approach presented in [63] uses optimization with the objective of minimizing the sum of the square of the joint torque. The optimization variable used here is a discretized B-spline parametrization of the path. Using the Power of Exponential (POE) to derive the system dynamics, the analytical gradient of the cost function with respect to the control input is calculated and used to solve the optimization problem. This work is extended in [11] where the torque term is applied to an OCP, here motion of the joints is approximated with B-spline polynomials.

A number of researchers have applied MPC to open chain manipulators in the past. Using computed torque control for feedback linearization, [9] implements an MPC scheme to a control a PUMA 560 manipulator. Three methods of implementing nonlinear MPC schemes for a 2-link vertical manipulator are presented in [94]. These methods based on dynamic matrix control, include a adaptive controller, a PID based controller and a simplified nonlinear MPC controller. Another formulation of MPC control for a 2 link arm is presented in [40] using a combination of linearization control and MPC. For our application, we consider dual objective formulation of MPC, which looks to both regulate a system to a desired set point and achieve a secondary objective. Combined regulatory and economic MPC is studied in [61], [44] to regulate chemical processes, while also trying to achieve a secondary objective. By using a weight modifier a greater emphasis can be put on the economic objective when possible, without compromising the regulatory objective. This is the idea we look to apply in Section 6.5. Other advantages of using MPC for a problem such as this one include that it allows constraints on joint actuation and velocity and torque limits. It also leaves the possibility to include task space limits such as obstacles, or other physical constraints in the environment.

The proposed algorithm has two main advantages. First, the POE formulation of the system kinematics combined with the Newton-Euler inverse dynamics allow for an exact nonlinear formulation of the dynamics which less computationally demanding than the Lagrangian forward dynamics. This allows us to run an MPC scheme, without having to take a linear approximation of the highly nonlinear dynamics of a robotic arm. Second, the introduction of a preference function let us formulate an MPC scheme without terminal constraints or costs, since we are able to reach zero running costs at the desired end effector position.

The remainder of this chapter is structured as follows, we first set up the POE formulation for an open chain manipulator in Section 6.2, we do this for a general n link manipulator. A Newton-Euler recursive algorithm to calculate the system dynamics is presented in Section 6.3 based on our model. Using the joint torques calculated from the system dynamics, we set up a finite horizon optimal control problem in Section 6.4 and examine the effects of changing the weights has on our solution. We then take results from

the finite horizon case and apply them to the receding horizon case or MPC in Section 6.5. We develop an alternative model which uses the Lagrangian system dynamics and compare the two approaches in Section 6.6. We conclude this chapter with comments on stability and some closing remarks.

6.2 Open Chain Manipulator Model

The system of interest here is a serial or open chain robot arm. The power of exponential formulation is used here to describe the system kinematics, this allows us to use an efficient recursive Newton-Euler method to obtain the system's inverse dynamics [74]. Let us consider an open chain robot with n joints, where $q, \dot{q}$ and $\ddot{q}$ are the joint positions, velocities and accelerations respectively. The position and orientation of one frame with respect to another can be represented by a transformation matrix in $SE(3)$

$$\mathcal{T}_{i,j} = \begin{bmatrix} \Theta & \mathbf{b} \\ 0 & 1 \end{bmatrix} \tag{6.1}$$

where Θ is a rotation matrix in $SO(3)$ and $\mathbf{b} \in \mathbb{R}^3$ is a relative position between frames i and j. While a complete discussion on and derivation can be found in [74] [71], the product of exponential formula gives a function between joint positions in a serial manipulator and its transformation matrix.

$$\mathcal{T}_{i-1,i} = f(q_i) = e^{[\mathcal{S}_i]q_i} M_i \tag{6.2}$$

Where $M_i \in SE(3)$ is defined by $f(0)$, which is the transformation $\mathcal{T}_{i-1,i}$ when $q_i = 0$, ie. the joint is in its home position. $\mathcal{S}_i$ is the screw axis

$$\mathcal{S} = \begin{bmatrix} \omega \\ \mathbf{v} \end{bmatrix} \in \mathbb{R}^6 \tag{6.3}$$

in the $i-1$ frame corresponding to joint motion at its home position. Here ω and $\mathbf{v}$ represent the normalized angular and linear velocities respectively, where either (i) $\|\omega\| = 1$ or (ii) $\omega = 0$ and $\|\mathbf{v}\| = 1$. If (i) holds then $\mathbf{v} = -\omega \times \mathbf{r}$ for pure rotation where $\mathbf{r}$ is a point on the screw axis [59]. Here $\mathbf{a} \times \mathbf{b}$ denotes the cross product of vectors $\mathbf{a}$ and $\mathbf{b}$. $[\mathcal{S}_i]$ is the $se(3)$ representation of $\mathcal{S}_i$,

$$[\mathcal{S}_i] = \begin{bmatrix} [\omega] & \mathbf{v} \\ 0 & 0 \end{bmatrix} \quad \text{and} \quad [\omega] = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \tag{6.4}$$

By multiplying transformations together we are able to obtain the transformation from the base frame to the end effector frame n, for a robot with n links.

$$\mathcal{T}_{0,n} = f(q_1, ..., q_n) = \mathcal{T}_{0,1}\mathcal{T}_{1,2}\dots\mathcal{T}_{n-1,n} \tag{6.5}$$

6.3 Newton-Euler Recursive Dynamics

The ability to look at the transformation from one link to the next individually lets us use a Newton-Euler recursive algorithm to obtain the kinematics and dynamics of our robot. This algorithm is derived in [74, Section 3]. The algorithm first does a forward recursion where the motion of each joint is propagated starting from the base link. Once we've obtained the velocities and accelerations of each link, the torque through each link can be calculated from backward propagation starting at the end effector, adding in any applied forces and the forces due to gravity. This allows us to avoid solving the forward dynamics. The following variables are defined for the purpose of this algorithm:

- $\mathbf{V}_i \in \mathbb{R}^6$ is the generalized velocity of the link i frame in frame i coordinates.
- $I_i \in \mathbb{R}^6$ is the inertia mass matrix of link i about joint i.
- $\mathbf{W}_i \in \mathbb{R}^6$ is the generalized force (or wrench) transmitted from link $i-1$ to link i. $\mathbf{W}_i = [\mathbf{m}_i^\top \mathbf{f}_i^\top]^\top$, where m_i is a moment vector and f_i is a force vector.
- Γ_i is the actuator torque at joint i.

The Newton-Euler recursive algorithm goes as follows:

- Initialization

$$\mathbf{V}_0 = \dot{\mathbf{V}}_0 = \mathbf{W}_{n+1} = 0 \tag{6.6}$$

- Forward recursion for $i = 1$ to n

$$\mathcal{T}_{i-1,i} = e^{[\mathcal{S}_i]q_i} M_i \tag{6.7}$$

$$\mathbf{V}_i = \mathrm{Ad}_{\mathcal{T}_{i-1,i}^{-1}}(\mathbf{V}_{i-1}) + \mathcal{S}_i \dot{q}_i \tag{6.8}$$

$$\dot{\mathbf{V}}_i = \mathcal{S}_i \ddot{q}_i + \mathrm{Ad}_{\mathcal{T}_{i-1,i}^{-1}}(\dot{\mathbf{V}}_{i-1}) + \mathrm{ad}_{\mathrm{Ad}_{\mathcal{T}_{i-1,i}^{-1}}(\mathbf{V}_{i-1})}(\mathcal{S}_i \dot{q}_i) \tag{6.9}$$

- Backward recursion for $i = n$ to 1

$$\mathbf{W}_i = \mathrm{Ad}^*_{\mathcal{T}_{i,i+1}}(\mathbf{W}_{i+1}) + I_i\dot{\mathbf{V}}_i - \mathrm{ad}^*_{\mathbf{V}_i}(I_i\mathbf{V}_i) \tag{6.10}$$

$$\Gamma_i = \mathcal{S}_i^\top \mathbf{W}_i \tag{6.11}$$

Note that we make use of the adjoint and dual adjoint mappings in this algorithm, see Appendix C for more details on the adjoint mappings.

6.4 Minimum Effort Optimal Control

Now that we have outlined an algorithm which enables efficient calculation of the torque in each joint, we use this information to create a minimal effort optimal controller. For the purpose of this chapter we define effort, E as the integral of the square of the torque $\Gamma = (\Gamma_1, \ldots, \Gamma_n)^\top$, where Γ_i denotes the torque at joint i. This is the definition used in [63], and can be written as

$$E(\Gamma) = \int_0^{t_f} \|\Gamma\|_S^2 dt \tag{6.12}$$

where t_f is the time required to complete the maneuver and $S \in \mathbb{R}^{n\times n}$ is a positive definite weighing matrix. Our system model must now include joints $\mathbf{q}$ as a state where

$$\mathbf{q}^+ = f(\mathbf{q}, \mathbf{u}) = \mathbf{q} + \tau\mathbf{u} \tag{6.13}$$

$$\mathbf{x} = g(\mathbf{q}) \tag{6.14}$$

where for a system with n joints, the joint positions $\mathbf{q} = (q_1, ..., q_n)^\top \in \mathcal{Q} \subset \mathbb{R}^n$ are our system states, the joint velocity, $\mathbf{u} = (u_1, ..., u_n)^\top \in U \subset \mathbb{R}^n$ are the control input to the system and the end effector position and orientation $\mathbf{x} = (x_1, ..., x_6)^\top \in X \subset \mathbb{R}^6$. The state constraint set $\mathcal{Q}$ can be defined as

$$\mathcal{Q} \in [\underline{q}_i, \bar{q}_i], \qquad i \in \{1, 2, ..., n\}.$$

Similarly we set saturation limits on the control inputs such that

$$U \in [\underline{u}_i, \bar{u}_i], \qquad i \in \{1, 2, ..., n\}.$$

where $\underline{u}_i < 0$ and $\bar{u}_i > 0$. Finally we can define the constraint on the output state as

$$X \in [\underline{x}_i, \bar{x}_i], \qquad i \in \{1, 2, ..., 6\}.$$

Before looking at an MPC algorithm, we study the simpler case of a finite horizon non-linear control algorithm. We define the following running costs at any time instant as

$$\ell(\mathbf{q}, \mathbf{u}) := \left\|\mathbf{x}(\mathbf{q}) - \mathbf{x}^\star\right\|_Q^2 + \left\|\mathbf{u}\right\|_R^2 + \left\|\Gamma(\mathbf{q}, \mathbf{u}, \dot{\mathbf{u}})\right\|_S^2 \tag{6.15}$$

where the torque $\Gamma(\mathbf{q}, \mathbf{u}, \dot{\mathbf{u}}) = h(\mathbf{q}, \mathbf{u}, \dot{\mathbf{u}})$ is calculated from the Newton-Euler algorithm in Section 6.2 and Q, R and S are positive definite weighing matrices. Note that adding a penalty term on the joint angular velocity serves two purposes here, it acts as a damping term and drives the system away from singularities. Since singularities produce situations with very large joint velocities, putting higher weight on the joint velocities drives the optimal solution away from these. We now define the objective function for our optimal control problem as the sum of the costs for a control horizon N.

$$J_N(\mathbf{q}_0, \mathbf{u}) := \sum_{k=1}^{N} \ell(\mathbf{q}_u(k, \mathbf{q}_0), \mathbf{u}(k))$$

where $\mathbf{q}_0$ is the initial joint position at $k = 0$ and $\mathbf{q}_u(\cdot, \mathbf{q}_0)$ is the sequence of joint positions for some control input sequence $\mathbf{u}(\cdot) \in U$. Using this objective function, we can now set up the OCP

$$V(\mathbf{q}_0) = \min_{u \in U} J_N(\mathbf{q}_0, \mathbf{u}) \tag{6.16}$$

$$\begin{aligned} \text{s.t.} \quad & \mathbf{q}(0) = \mathbf{q}_0, \\ & \Gamma(k) = h(\mathbf{q}, \mathbf{u}, \dot{\mathbf{u}}) \\ & \mathbf{q}(k+1) = \mathbf{q}(k) + \tau \mathbf{u}(k) \\ & \mathbf{x}(k) = g(\mathbf{q}(k)) \\ & \mathbf{q}(k) \in \mathcal{Q} \\ & \mathbf{u}(k) \in U \\ & \mathbf{x}(k) \in X \end{aligned} \tag{6.17}$$

The constraints for the OCP include enforcing the torque in each joint according to the Newton-Euler recursive algorithm in Section 6.2, the propagation of the control inputs in the joint states and the forward kinematics to determine the end effector position from the joint states. Note that this formulation differs from [63] as we do not use points on a parametrized B-spline as an the optimization variable but instead use the control input $\mathbf{u}$. This provides the optimal control input over the prediction horizon instead of providing an optimal trajectory for the joint pose.

6.4.1 Effect of Weights on Optimal Solution

To set the weights for our the multi-objective optimization problem defined in (6.16), we must first understand how the weights affect the solution to the OCP. Considering the OCP as a Multi-objective optimization problem (MOOP) and with all of the weights being positive, the solution to the OCP is Pareto optimal [62]. That is to say that the solution we obtain must be strictly better than another solution in at least one objective. Changing the weights of our objective function will provide us with different Pareto points. However, we may not be able access all Pareto points by changing weights, since we have non-linear constraints in both torque and position.

To study the effects of varying the weights we use an example of a three revolute joint (3R) robot in 2D space shown in Fig. 6.1. A system with a redundant degree of freedom is chosen deliberately as it gives the algorithm freedom to settles to a minimum torque static configuration. The three joint arm is modeled as three thin rods of mass $m_1 = m_2 = m_3 = 1$ [kg] and length, $l_1 = l_2 = l_3 = 0.5$ [m]. For the first simulations, the robot has an initial joint position of $\mathbf{q} = (q_1, q_2, q_3)^\top = [0, 0, 0]^\top$ and a desired end effector pose for of $\mathbf{x}^\star = [0.5, 0.5]^\top$. A desired orientation is not specified and there is no applied force to the end effector. The simulation is implemented in MatLab. To solve the OCP, the Casadi toolbox is used with an interior point optimization solver (IPOPT)[6]. The visualization of the simulation is created through the MatLab robotics toolbox.

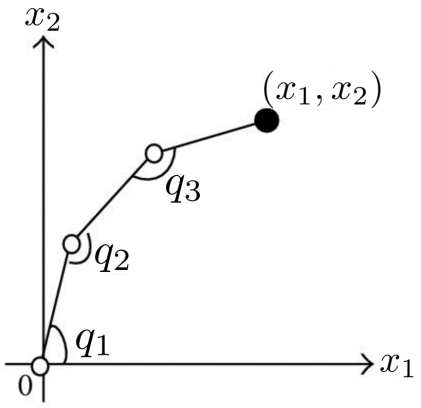

Figure 6.1: Model of a 3R planar robot used for simulation with an actuator at each joint, this image is reprinted from [90].

The results of our first simulation are shown in Table 6.1. Where the values for Q, R and S represent the diagonal entries in the respective weighting matrices, the error is the

steady state position error defined by the Euclidean norm, the effort is the sum of the square of the torque at each time instant as defined in 6.12 and the peak is the peak torque for all the joints over the maneuver. The trajectories for Trial 1 and Trial 6 are shown in Fig. 6.2 (Left) and (Right) respectively.

Table 6.1: Results of simulation trials with $x_d = [0.5, 0.5]^\top$, with terminal constraints added to trial 3.

Trial	Q	R	S	Error [m]	Effort [Nm]^2	Peak [Nm]
1	1000	1	0	0	2001	86.3
2	0	1	1	2.01	51.8	7.98
3	0	1	1	0	182	10.5
4	1	1	1	1.11	175	12
5	100	1	1	0.26	351.1	34.3
6	1000	1	1	0.106	247.2	18.3
7	1.00E+06	1	1	0.00016	2111	88.3

Though we try to minimize joint torque, the primary objective of the controller is still to accurately move the end effector to a desired position. We note that with no weight on the position error (Trial 2), the arm simply falls to a minimum torque position. We also see in Trial 3 that adding a terminal constraint

$$\mathbf{x}(N) = \mathbf{x}^\star \tag{6.18}$$

can force the arm to the desired end effector position. When the terminal constraint is removed, we must add very large weights on the position error to drive the steady state error to zero. We do however note that since the costs are not normalized, the significantly larger weights on the position can be attributed in part to scaling since the torque costs reach values much larger than the position costs.

These results highlight an issue as only one objective is satisfied at a time, the solution where both objectives are met simultaneously is not found, i.e. the weights can be set to either minimize the torques or have no steady state position error.

6.4.2 Preference Function

To address this issue, we use the idea of a preference function to add a time dependence to the weights. This lets the weight be heavier on the torque costs at the beginning of

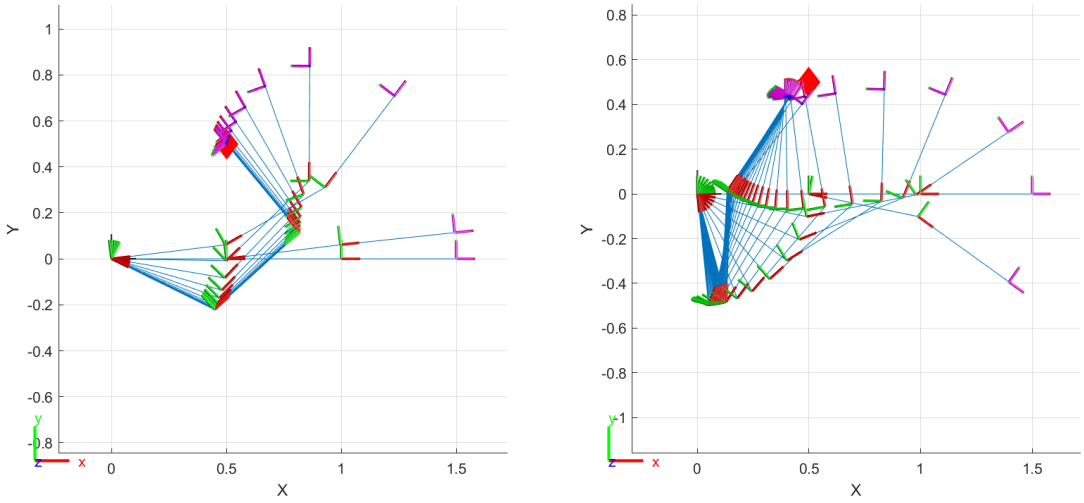

Figure 6.2: Trajectory of the 3R arm stabilizing to $x_d = [0, 1.5]^\top$ using only regulation OCP (Left) and using the dual objective OCP algorithm. (Right) The red diamonds represent the desired end effector pose.

the maneuver, then reduces the weight later in the trajectory when an accurate position is needed. The running costs become

$$\ell(\mathbf{q}, \mathbf{u}, k) := \left\|\mathbf{x}(\mathbf{q}) - \mathbf{x}^\star\right\|_Q^2 + \left\|\mathbf{u}\right\|_R^2 + \phi(k) \left\|\Gamma(\mathbf{q}, \mathbf{u}, \dot{\mathbf{u}})\right\|_S^2 \tag{6.19}$$

where $\phi(k)$ is the preference function. We want the OCP to start with a heavier weight on the torque term and reduce it to zero as time goes on to eliminate any position error. Therefore we choose a sigmoid function

$$\phi(k) = \frac{1}{1 + e^{-\rho(-k+\sigma)}} \tag{6.20}$$

as a preference function. Here ρ and σ are tuning parameters which stretch and shift the function horizontally as shown in Fig. 6.3.

To appropriately tune $\phi(k)$ we must have some understanding of the desired system behaviour. In this case, we want to ensure that ρ is sufficiently large so that the settling time to the desired position isn't too long, but not so large that we get a very sudden change in $\phi(k)$. Through simulation trials a value of $\rho = 0.2$ was shown to be appropriate. σ should be set in a way which reflects how long we anticipate the maneuver to be. If σ

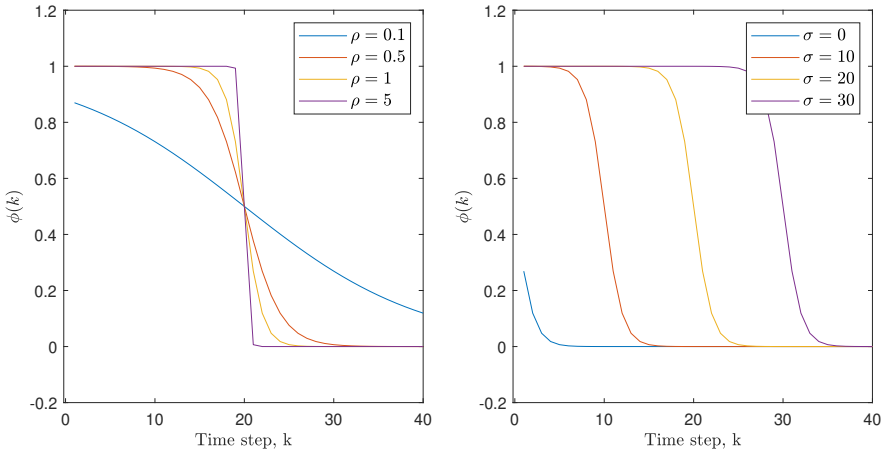

Figure 6.3: Left: Value of ϕ with $\sigma = 20$ and varying ρ. Right: Value of ϕ with $\rho = 1$ and varying σ.

is set too small, the torque will only be minimized for a small part of the trajectory and conversely if it is set too large there will again be longer than desired settling times. We then set σ based on the distance from the desired position

$$\sigma(\mathbf{x}_0, \mathbf{x}^\star) = \nu \|\mathbf{x}_0 - \mathbf{x}^\star\|$$

where ν is a scaling constant. $\nu = 7$ was found to give desirable behaviour in simulation. The last simulation where $\mathbf{x}^\star = [0.5, 0.5]^\top$ is repeated with running costs (6.19). The results are shown in Table 6.2. The improvement in the performance can be seen by comparing trial 2 with trial 6 from Table 6.1 which has the same parameters. We see that the error of 0.106m vanishes, the peak torque and effort are also less. The optimization problem is also less sensitive to weights, the results not changing drastically with changes on the torque weights. The trajectory of the arm using the preference function is compared to the trajectory with weights only on the position error in Fig. 6.4.

Table 6.2: Results of simulation trials with $x_d = [0.5, 0.5]^\top$ and running costs (6.19)

Trial	Q	R	S	Error [m]	Effort [Nm^2]	Peak [Nm]
1	1000	10	0.1	0	406	37.2
2	1000	10	1	0.002	296	19.1
3	1000	10	10	0.013	384	24.5

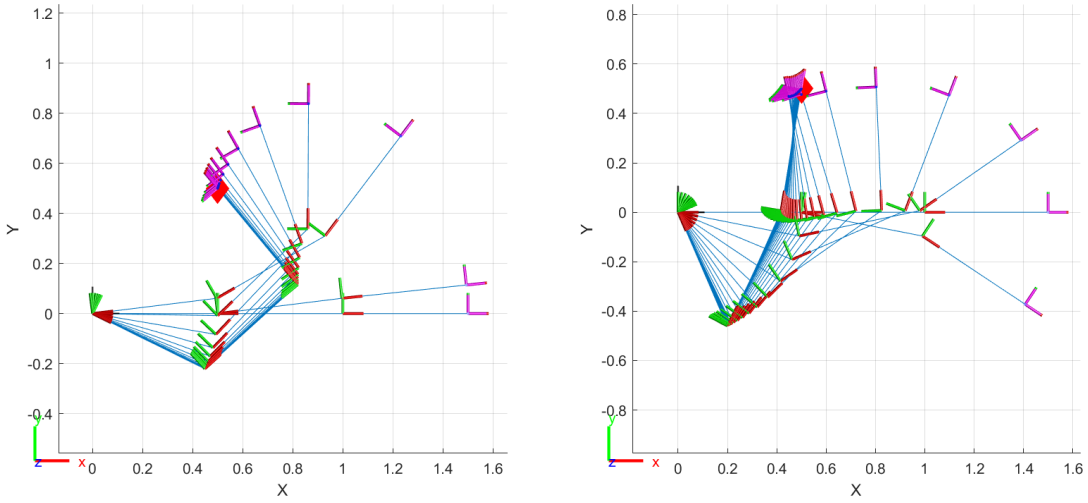

Figure 6.4: Trajectory of the 3R arm stabilizing to $x_d = [0.5, 0.5]^\top$ using only regulation OCP (Left) and using the dual objective OCP algorithm (Right). The red diamonds represent the desired end effector position.

6.5 Torque Minimization MPC formulation

Our next task is to apply what we developed in the finite horizon OCP to the receding horizon optimal control problem of MPC. An algorithm for enforcing convergence on combined regulatory and economic MPC is presented in [61]. The method presented uses a weighted dual-objective function.

$$\ell_\phi(\mathbf{x}, \mathbf{u}) := \phi(\mathbf{x}, \mathbf{u})\ell_{\bar{e}}(\mathbf{x}, \mathbf{u}) + (1 - \phi(\mathbf{x}, \mathbf{u}))\ell_r(\mathbf{x}, \mathbf{u})$$

where $\phi(\mathbf{x}, \mathbf{u})$ is a tunable weight. To relate this to our problem, ℓ_r are the regulatory running costs

$$\begin{aligned} \ell_r(\mathbf{q}, \mathbf{u}) &:= \left\|\mathbf{x}(\mathbf{q}) - \mathbf{x}^\star\right\|_Q^2 + \left\|\mathbf{u}\right\|_R^2, \qquad \text{and} \\ \ell_{\bar{e}}(\mathbf{q}, \mathbf{u}) &:= \ell_e(\mathbf{q}, \mathbf{u}) - \ell_e(\mathbf{q}_s, \mathbf{u}_s) \end{aligned}$$

where ℓ_e are the economic costs

$$\ell_e(\mathbf{q}, \mathbf{u}) = \left\|\Gamma(\mathbf{q}, \mathbf{u}, \dot{\mathbf{u}})\right\|_S^2 \tag{6.21}$$

$\mathbf{q}_s$ and $\mathbf{u}_s$ are the solution to the steady state economic optimization problem

$$\begin{aligned}(\mathbf{q}_s, \mathbf{u}_s) := \arg\min_{(\mathbf{q},\mathbf{u})} &\{\ell_e(\mathbf{q}, \mathbf{u}) | \mathbf{q} = f(\mathbf{q}, \mathbf{u})\} \\ \textbf{s.t.} \quad &\mathbf{q}(k) \in \mathcal{Q} \\ &\mathbf{u}(k) \in U \\ &\mathbf{x}(k) \in X\end{aligned}$$

The main issue with this formulation is that since in general the torque at the desired set point $\Gamma(\mathbf{q}_s) \neq 0$, which implies that $\ell(\mathbf{q}_s, \mathbf{u}_s) \neq 0$ and therefore $\ell_{\bar{e}}(\mathbf{q}, \mathbf{u}) \not\geq 0 \ \forall \ \mathbf{q} \in \mathcal{Q}$ and $\mathbf{u} \in U$ which violates the assumptions required for asymptotic stability in Theorem 2.2.2. Instead we simply use the economic cost ℓ_e with sigmoid function (6.20) to drive the weight on the economic cost to zero as we move along the trajectory, so that there is no penalty on torque by the end of the maneuver. Our weighted dual-objective running costs become

$$\ell_\phi(\mathbf{x}, \mathbf{u}, k) := \ell_r(\mathbf{x}, \mathbf{u}) + \phi(k)\ell_e(\mathbf{x}, \mathbf{u}).$$

The MPC algorithm is set to run until the states converge to some desired position $\mathbf{x}^\star$ within some tolerance $\delta > 0$. $\phi(i)$ will continue to shift more weight onto the position error until either the system converges to $\mathbf{x}^\star$ or until $\phi(k) = 0$. If $\phi(k) = 0$ we are back to simply having a point stabilization problem. For the optimal control problem, we can use (6.16) with the value function

$$J_N(\mathbf{q}_0, \mathbf{u}) := \sum_{k=1}^{N} \ell_\phi(\mathbf{q}_u(k, \mathbf{q}_0), \mathbf{u}(k))$$

The solution to the OCP is a control sequence $\mathbf{u}_N^* \in U_N(\mathbf{x}_0)$. The first control input of the sequence denoted as $\mu_{x_0} = \mathbf{u}_N^*(1, \mathbf{x}_0)$ is applied to the system. $U_N(\mathbf{x}_0)$ denotes all admissible control sequences such that the constraints in (6.16) are satisfied. The OCP is solved and μ_{x_0} is applied repeatedly until $|\mathbf{x} - \mathbf{x}^\star| \leq \delta$.

We study the performance of our new MPC algorithm, with a comparison to the results of regulation MPC by setting $\phi(k) = 0 \quad \forall k \in \mathbb{N}$. For this simulation $Q_{i,i} = 200$ for $i = 1, 2$ and $R_{i,i} = 20$, $S_{i,i} = 1$ for $i = 1, 2, 3$. The system in the simulation is the same as Section 6.4.1. The results of the simulation are shown in Table 6.3. There is a clear improvement in torque and overall effort for the dual-objective controller. It should be noted that the algorithm stopped once $|\mathbf{x} - \mathbf{x}^\star| \leq 0.001$ therefore both algorithms were able to achieve the desired accuracy. The trajectories of the arm for both algorithms are shown in Figs. 6.5.

Table 6.3: Results of MPC Simulation

Trial	Q	R	S	Error [m]	Effort [Nm^2]	Peak [Nm]
Dual	200	20	1	0.001	164.3	14.3
Regulation	200	20	0	0.0003	1929	89.8

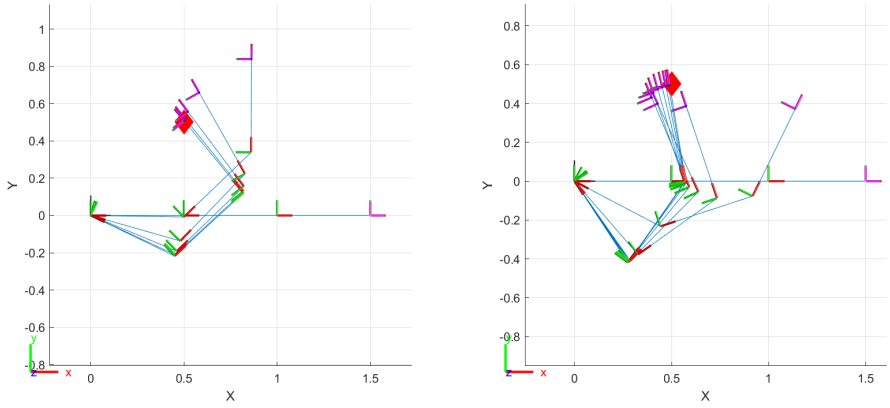

Figure 6.5: Trajectory of the 3R arm stabilizing to $x_d = [0.5, 0.5]^\top$ using only regulation MPC (Left) and using the dual objective MPC algorithm (Right).

It is also important to show that the algorithm can stabilize the system to the desired position from any position in the allowable space. We therefore show in Fig. 6.6 that we can stabilize the system to $\mathbf{x}^\star = [0.5, 0.5]^\top$ from a variety of initial conditions.

6.6 Alternate Dynamic Model Formulation

As an alternate formulation to Newton-Euler formulation used in previous sections, we can also create a dynamic model of the system using the Lagrangian, for an example with a 2-link planar robot see [40]. Here we consider the dynamic model obtained from the Lagrangian of the system

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \Gamma \tag{6.22}$$

where $\mathbf{q}$, $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ are the joint angular position, velocity and acceleration respectively and Γ is the torque applied at the joints. $\mathbf{M}(\mathbf{q})$ is the mass matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ accounts for Coriolis forces and $\mathbf{G}(\mathbf{q})$ accounts for forces due to gravity. Like the Newton Euler formulation, the position of the end effector can be determined from the joint positions, i.e. $\mathbf{x}(\mathbf{q}) = g(\mathbf{q})$.

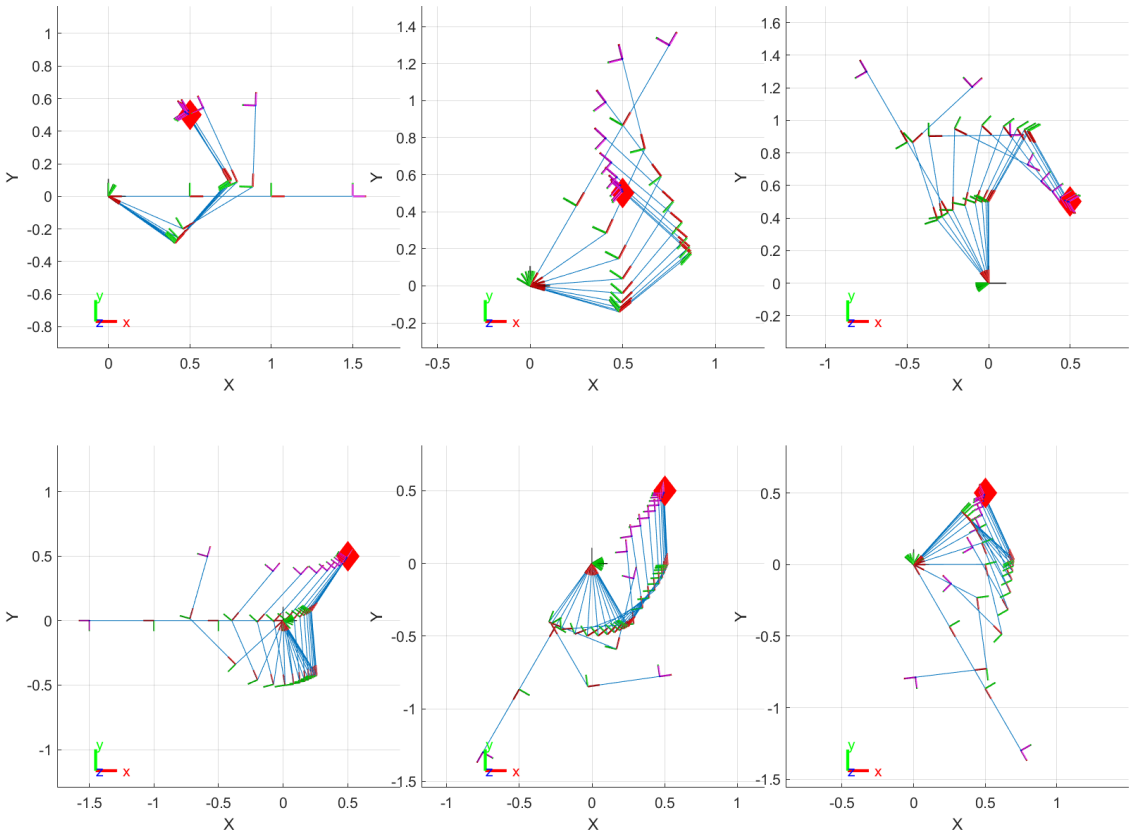

Figure 6.6: Trajectory of the 3R arm stabilizing to $x_d = [0.5, 0.5]^\top$ using the dual objective MPC algorithm from various initial positions. The red diamonds represent the desired end effector pose.

In this formulation, we treat the applied torque as the control input and can solve for joint acceleration $\ddot{\mathbf{q}}$ for a given torque Γ using forward dynamics

$$\ddot{\mathbf{q}} = \mathbf{M}(\mathbf{q})^{-1} \left(\Gamma - \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{G}(\mathbf{q}) \right).$$

Assuming a constant acceleration over a small time interval, we obtain an approximation for the angular velocity and position of the joints at some future time $t^+ = t + \tau$ by integrating over the time step τ we obtain

$$\dot{\mathbf{q}}^+ = \dot{\mathbf{q}} + \ddot{\mathbf{q}}\,\tau$$
$$\mathbf{q}^+ = \mathbf{q} + \dot{\mathbf{q}}\,\tau + \frac{1}{2}\ddot{\mathbf{q}}\,\tau^2$$

Using this dynamic model of the system, we set up an optimal control problem. We

begin by defining the following running costs

$$\ell(\mathbf{q}, \dot{\mathbf{q}}, \Gamma) := \left\|\mathbf{x}(\mathbf{q}) - \mathbf{x}^\star\right\|_Q^2 + \left\|\dot{\mathbf{q}}\right\|_R^2 + \left\|\Gamma\right\|_S^2 \tag{6.23}$$

These are the same running costs as (6.15) reformulated with for the new model. We define the objective function for the optimal control problem as the sum of the costs for a prediction horizon N.

$$\begin{aligned}
\min_{u \in U} \quad & J_N(\mathbf{q}, \dot{\mathbf{q}}, \Gamma) = \sum_{k=1}^{N} \ell(\mathbf{q}, \dot{\mathbf{q}}, \Gamma) && (6.24)\\
\text{s.t.} \quad & \mathbf{q}(0) = \mathbf{q}_0, && (6.25)\\
& \ddot{\mathbf{q}}(k) = \mathbf{M}(\mathbf{q}(k))^{-1} \left(\Gamma(k) - \mathbf{C}(\mathbf{q}(k), \dot{\mathbf{q}}(k)) - \mathbf{G}(\mathbf{q}(k))\right) && (6.26)\\
& \dot{\mathbf{q}}(k+1) = \dot{\mathbf{q}}(k) + \ddot{\mathbf{q}}(k)\, \tau && (6.27)\\
& \mathbf{q}(k+1) = \mathbf{q}(k) + \dot{\mathbf{q}}(k)\, \tau + \frac{1}{2}\ddot{\mathbf{q}}(k)\, \tau^2 && (6.28)\\
& \mathbf{x}(k) = g(\mathbf{q}(k)) \\
& \mathbf{q}(k) \in \mathcal{Q} \\
& \Gamma(k) \in U
\end{aligned}$$

Here the system dynamics are enforced through constraint (6.26) and the system kinematics are enforced through constraints (6.27) and (6.28). In a similar fashion to the Newton Euler MPC formulation, the solution to the OCP (6.24) is applied to the system in a receding fashion. Since the running costs are the same as the previous implementation, we again cannot reach both objectives of minimizing torque and stabilizing the system to a desired position simultaneously. We therefore modify the running costs with a preference function again giving

$$\ell_\phi(\mathbf{q}, \dot{\mathbf{q}}, \Gamma, k) := \left\|\mathbf{x}(\mathbf{q}) - \mathbf{x}^\star\right\|_Q^2 + \left\|\dot{\mathbf{q}}\right\|_R^2 + \phi(k) \left\|\Gamma\right\|_S^2, \tag{6.29}$$

where $\phi(k)$ is defined in Equation (6.20). The trajectories starting from various initial joint positions are shown in Fig. 6.7. We note that this formulation produces different trajectories from those using the Newton-Euler recursive algorithm shown in Fig. 6.6. Though both these models provide a closed form solution to the system dynamics, the Newton-Euler model considers the inverse dynamics where as the Lagrange model presented in this section considers the forward dynamics. Due to the fact that we employ a discrete time model of this system, propagating even a small difference over a prediction horizon could cause the optimization algorithm to return different solutions. Regardless of these differences, we note that the algorithm can still regulate the system to the desired position from any of the initial positions shown.

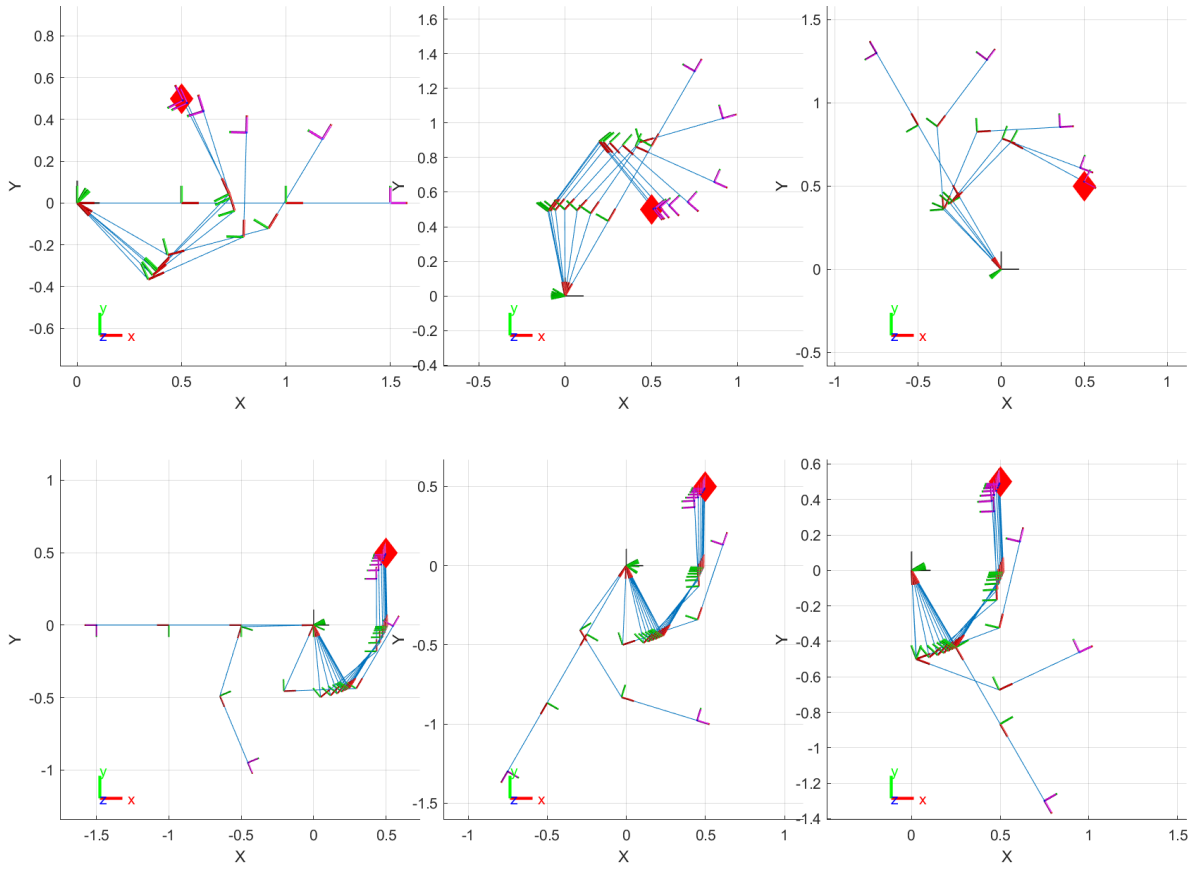

Figure 6.7: Trajectory of the 3R arm stabilizing to $x_d = [0.5, 0.5]^\top$ using the dual objective MPC algorithm from various initial positions. The red diamonds represents the target end effector pose.

We compare the two algorithms using three metrics, the peak torque over all the joints, the total effort required to complete the trajectory as defined in (6.12) (with $S = \mathbf{I}_{3\times3}$ and the time to solve one iteration of the receding horizon OCP. The results from both algorithms are shown in Table 6.4. Some of the key takeaways from this comparison include that due to their different solutions, the dynamic model formulation achieves slightly lower peak torque and effort, but has significantly higher computing costs. The average time to solve the OCP for the Lagrangian model is almost twice that of the Newton-Euler recursive model. This difference is expected to get even more significant as the number of degrees of freedom in the system increases.

Table 6.4: Comparison of two dual-MPC algorithms for the trajectories shown in Figs. 6.6 and 6.7.

	Dynamic Model			Newton Euler Model		
Pose [rad]	Peak [Nm]	Effort [Nm^2]	Time [s]	Peak [Nm]	Effort [Nm^2]	Time [s]
$[0, 0, 0]$	20.8	646	0.081	23.9	389	0.043
$[\pi/3, 0, 0]$	12.8	291	0.097	16.4	710	0.072
$[2\pi/3, 0, 0]$	16.9	198	0.049	12	159	0.040
$[\pi, 0, 0]$	15.3	408	0.137	20.8	293	0.042
$[4\pi/3, 0, 0]$	14	348	0.088	16.5	491	0.044
$[5\pi/3, 0, 0]$	18.4	486	0.088	13.8	707	0.038

6.7 Stability

No formal proof of stability is presented in this chapter, however, since the minimum effort controller developed here classifies as an MPC scheme without terminal costs or constraints. We can look at the stability of this system through the lens of the stability review in Section 2.2.2. The main result of this Section state that for a system satisfying assumptions 5 and 6 the value function will act as a Lyapunov function as described in inequality 2.34. Without proving that Theorem 2.1 holds, we can verify the result of a monotonically decreasing value function for the initial states shown in Figs. 6.6 and 6.7. We see that this is indeed the case for our algorithm using either of the discussed model in Fig. 6.8.

For a more complete analysis of stability, conditions for which Theorem 2.1 holds would need to be developed, in a similar way to Chapters 4 and 5. This will be the subject of future works, with the intent of establishing a minimum prediction horizon length and weights for which stability is guaranteed.

6.8 Conclusion

In this chapter we develop the proof of concept for a dual-objective MPC algorithm for torque minimization in an open chain robotic arm. Using the power of exponential formula, to apply a fast recursive algorithm for the system dynamics, we calculate the torque in each joint of the arm. By studying the effects of the cost function weights for a simpler finite horizon optimal control problem, we are able to see that by setting the weights to a single value, we are unable to reach a solution which both minimizes torque and position

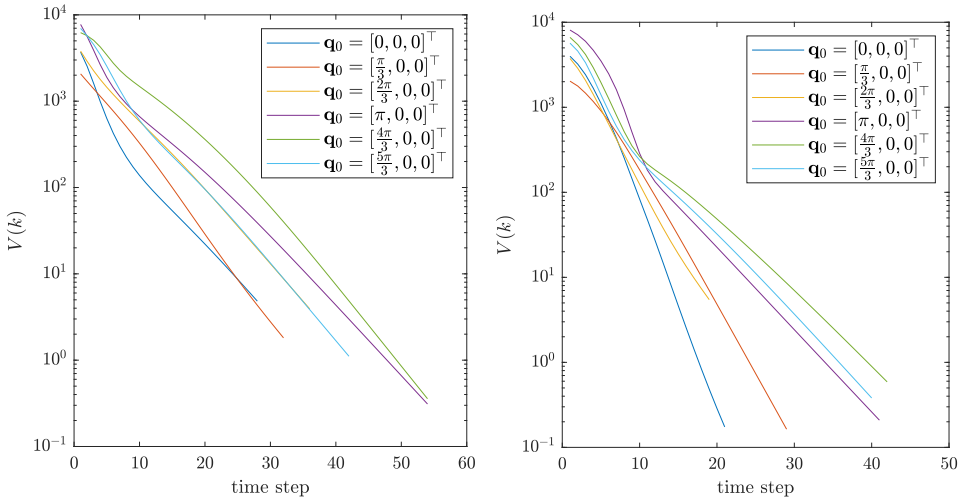

Figure 6.8: Left: Time evolution of the value function for Newton Euler formulation of dynamics starting from various positions, Right: for the Dynamic Model formulation starting from various positions.

error. To overcome this we introduce a time dependent preference function, which at the start of the maneuver puts emphasis on minimizing torque and decreases this emphasis as time goes on. Finally we extend this to MPC, which behaves in a similar way to the finite horizon problem. This formulation provides us with a significant reduction in torque and overall effort for the serial chain manipulator compared to point stabilization MPC. In future works, this can be extended to more complex systems, such as a 7DOF robotic manipulator or mobile manipulator operating in 3D space. Due to the nature of the POE formula and the recursive algorithm for dynamics this can be done with relative ease. We can also extend this method to cases where the robot is supporting some external load at the end effector, as this is also easily incorporated in the Newton-Euler recursive formulation. Finally, future works can also include developing a stability proof got this algorithm.

Chapter 7

Conclusions and Future Works

7.1 Conclusions

In this thesis, we developed four nonlinear model predictive controllers without terminal constraints or costs. The first, a point stabilizing controller, is used as a basis for the next two, an acceleration constrained point stabilizing controller and a path following controller. The asymptotic stability of these three controllers is rigorously proven by using sub-optimal open loop maneuver in order to build a bound on the closed-loop running costs of the MPC algorithms.

For the point stabilizing controller in Chapter 3 we are able to prove that the stability holds for the shortest possible prediction horizon of N=2. We additionally verify our results through numerical simulation and experiment.

For the acceleration constrained point stabilizing algorithm, we extend current work in the theory of barriers to a simplified model of a holonomic mobile robot. This allows us to develop an admissible set in order to guarantee recursive feasibility, which is required to guarantee asymptotic stability for a sufficiently long prediction horizon. These results are verified through numeric simulation.

The path following controller we develop is designed to take advantage to take of the three degrees of freedom of the holonomic mobile robot and can drive it to very accurately follow paths defined by arbitrary functions or sets of via points. The asymptotic stability of this algorithm is proven in a similar way for a sufficiently long prediction horizon. These results are also verified through numeric simulation.

Finally the fourth controller we propose is for the minimum effort control of a robotic arm. This controller employs the power of exponent formula with an efficient recursive Newton-Euler formulation of the inverse dynamics, this allows us to use the exact nonlinear dynamics for MPC. This is combined with a time dependent preference function, which at the start of the maneuver penalizes the joint torque but as time goes on reduces this weight in order to ensure that the desired end effector position is reached. This allows us to run MPC without terminal constraints or costs.

7.2 Future Works

The work on stability in this thesis, using Theorems 2.2 and 2.3 to prove asymptotic stability with growth bound functions, can be extended to other MPC controllers. These could include other controllers for mobile robots such as an obstacle avoidance controller. It may also be used for other systems and control objectives such as the point stabilization for a robotic arm or mobile manipulator.

The work done for acceleration constraint systems can be extended to other second order differential systems. In robotics this could include any system where we want to constrain the force or acceleration. For a holonomic robot, we can directly extend our case to one where the orientation of the robot is not fixed.

The path following controller can be extended to an acceleration constrained system using the method developed in Chapter 4 and to include obstacle avoidance.

The work on the minimum effort controller presented in Chapter 6 can also be extended. First a proof of asymptotic stability can be developed, potentially using the same methods as the other chapters in this thesis. The controller can also be used for higher dimensional systems, such as a 7 degree of freedom robotic arm.

Finally, other than the point stabilization controller, none of these controllers have been verified experimentally. Using a similar experimental method to the one presented in Section 3.5.2 the other controllers in this thesis could also be validated.

References

[1] Markus Achtelik. vicon_bridge, 2010. Code available at: https://github.com/ethz-asl/vicon_bridge.

[2] A Pedro Aguiar, Joao P Hespanha, and Petar V Kokotovic. Path-following for non-minimum phase systems removes performance limitations. *IEEE Transactions on Automatic Control*, 50(2):234–239, 2005.

[3] A. Pedro Aguiar, João P. Hespanha, and Petar V. Kokotović. Performance limitations in reference tracking and path following for nonlinear systems. *Automatica*, 44(3):598 – 610, 2008.

[4] S. A. Al-Hiddabi and N. H. McClamroch. Tracking and maneuver regulation control for nonlinear nonminimum phase systems: application to flight control. *IEEE Transactions on Control Systems Technology*, 10(6):780–792, 2002.

[5] Joel Andersson. *A General-Purpose Software Framework for Dynamic Optimization*. PhD thesis, Arenberg Doctoral School, KU Leuven, Dept. Elect. Eng., Belgium, October 2013.

[6] Joel A E Andersson, Joris Gillis, Greg Horn, James B Rawlings, and Moritz Diehl. CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1):1–36, 2019.

[7] Humberto X. Araújo, André G.S. Conceição, Gustavo H.C. Oliveira, and Jônatas Pitanga. Model predictive control based on LMIs applied to an omni-directional mobile robot. *IFAC Proceedings Volumes*, 44(1):8171 – 8176, 2011. 18th IFAC World Congress.

[8] Giovanni Buizza Avanzini, Andrea Maria Zanchettin, and Paolo Rocco. Constraint-Based Model Predictive Control for Holonomic Mobile Manipulators. In *IEEE/RSJ*

International Conference on Intelligent Robots and Systems (IROS), pages 1473–1479. IEEE, 2015.

[9] Victor M Becerra, Steven Cook, and Jiamei Deng. Predictive computed-torque control of a puma 560 manipulator robot. *IFAC Proceedings Volumes*, 38(1):229–234, 2005.

[10] Robert R Bitmead, Michel Gevers, and Vincent Wertz. *Adaptive optimal control: the thinking man's GPC*. Prentice Hall New York, 1990.

[11] James E Bobrow, Frank C Park, and Athanasios Sideris. Recent advances on the algorithmic optimization of robot motion. In *Fast Motions in Biomechanics and Robotics*, pages 21–41. Springer, 2006.

[12] A. Boccia, L. Grüne, and K. Worthmann. Stability and feasibility of state constrained MPC without stabilizing terminal constraints. *Syst. Control Lett.*, 72(8):14–21, 2014.

[13] Hans Georg Bock and Karl-Josef Plitt. A multiple shooting algorithm for direct solution of optimal control problems. *IFAC Proceedings Volumes*, 17(2):1603–1608, 1984.

[14] Jan Brinkhuis and Vladimir Tikhomirov. *Optimization: Insights and Applications: Insights and Applications*. Princeton University Press, 2011.

[15] Xavier Broqu, Daniel Sidobre, and Khoi Nguyen. From Motion Planning to Trajectory Control with Bounded Jerk for Service Manipulator Robots. In *2010 IEEE International Conference on Robotics and Automation*, pages 4505–4510, 2010.

[16] Eduardo F Camacho and Carlos Bordons Alba. *Model predictive control*. Springer Science & Business Media, 2013.

[17] CC Chen and L Shaw. On receding horizon feedback control. *Automatica*, 18(3):349–352, 1982.

[18] Hong Chen and Frank Allgöwer. A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. *Automatica*, 34(10):1205–1217, 1998.

[19] DW Clark, C Mohtadi, PS Tuffs, et al. Generalized predictive control—part i: The basic algorithm. *Automatica*, 23(2):137–148, 1987.

[20] Andre Scolari Conceicao, A. Paulo Moreira, and Paulo J. Costa. Trajectory tracking for omni-directional mobile robots based on restrictions of the motor's velocities. *IFAC Proceedings Volumes*, 39(15):121 – 125, 2006. 8th IFAC Symposium on Robot Control.

[21] J.-M. Coron, L. Grüne, and K. Worthmann. Model predictive control, cost controllability, and homogeneity. *arXiv1906.05112*, 2019.

[22] John J. Craig. *Introduction to robotics : mechanics and control*. Pearson/Prentice Hall, Upper Saddle River, N.J, 3rd ed. edition, 2005.

[23] Charles R Cutler and Brian L Ramaker. Dynamic matrix control?? a computer control algorithm. In *joint automatic control conference*, page 72, 1980.

[24] José A De Doná and Jean Lévine. On barriers in state and input constrained nonlinear systems. *SIAM Journal on Control and Optimization*, 51(4):3208–3234, 2013.

[25] G De Nicolao, L Magni, and Riccardo Scattolini. On the robustness of receding-horizon control with terminal constraints. *IEEE Transactions on Automatic Control*, 41(3):451–453, 1996.

[26] RAJ De Vries and HB Verbruggen. Multivariable process and prediction models in predictive control—a unified approach. *International journal of adaptive control and signal processing*, 8(3):261–278, 1994.

[27] Moritz Diehl, Hans Georg Bock, Holger Diedam, and P-B Wieber. Fast direct multiple shooting algorithms for optimal robot control. In *Fast motions in biomechanics and robotics*, pages 65–93. Springer, 2006.

[28] P. Encarnação, A. Pascoal, and M. Arcak. Path following for autonomous marine craft. *IFAC Proceedings Volumes*, 33(21):117 – 122, 2000. 5th IFAC Conference on Manoeuvring and Control of Marine Craft (MCMC 2000), Aalborg, Denmark, 23-25 August 2000.

[29] Willem Esterhuizen, Karl Worthmann, and Stefan Streif. Recursive feasibility of continuous-time model predictive control without stabilising constraints. *arXiv preprint arXiv:2003.07598*, 2020.

[30] Jianqing Fan and Irene Gijbels. *Local polynomial modelling and its applications: monographs on statistics and applied probability 66*, volume 66. CRC Press, 1996.

[31] T. Faulwasser. *Optimization-based Solutions to Constrained Trajectory-tracking and Path-following Problems*. Shaker, Aachen, Germany, 2013.

[32] T. Faulwasser and R. Findeisen. Nonlinear model predictive control for constrained output path following. *IEEE Transactions on Automatic Control*, 61(4):1026–1039, 2016.

[33] T. Faulwasser, B. Kern, and R. Findeisen. Model predictive path-following for constrained nonlinear systems. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 8642–8647, 2009.

[34] T. Faulwasser, T. Weber, P. Zometa, and R. Findeisen. Implementation of nonlinear model predictive path-following control for an industrial robot. *IEEE Transactions on Control Systems Technology*, 25(4):1505–1511, July 2017.

[35] Timm Faulwasser, Janine Matschek, Pablo Zometa, and Rolf Findeisen. Predictive path-following control: Concept and implementation for an industrial robot. In *2013 IEEE International Conference on Control Applications (CCA)*, pages 128–133. IEEE, 2013.

[36] Sébastien Gros, Rien Quirynen, and Moritz Diehl. Aircraft control based on fast nonlinear mpc & multiple-shooting. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 1142–1147. IEEE, 2012.

[37] Lars Grüne. Analysis and design of unconstrained nonlinear mpc schemes for finite and infinite dimensional systems. *SIAM Journal on Control and Optimization*, 48(2):1206–1228, 2009.

[38] Lars Grüne and Jürgen Pannek. *Nonlinear Model Predictive Control*, pages 45–69. Springer International Publishing, 2017.

[39] Lars Grüne, Jürgen Pannek, Martin Seehafer, and Karl Worthmann. Analysis of unconstrained nonlinear mpc schemes with time varying control horizon. *SIAM Journal on Control and Optimization*, 48(8):4938–4962, 2010.

[40] Elhadi Guechi, Samir Bouzoualegh, Zennir Youcef, and Saso Blazic. Mpc control and lq optimal control of a two-link robot arm: A comparative study. *Machines*, 6:37, 08 2018.

[41] John Hauser and Rick Hindman. Maneuver regulation from trajectory tracking: Feedback linearizable systems*. *IFAC Proceedings Volumes*, 28(14):595 – 600, 1995. 3rd IFAC Symposium on Nonlinear Control Systems Design 1995, Tahoe City, CA, USA, 25-28 June 1995.

[42] John Hollerbach and Ki Suh. Redundancy resolution of manipulators through torque optimization. *IEEE Journal on Robotics and Automation*, 3(4):308–316, 1987.

[43] Hsu-Chih Huang and Ching-Chih Tsai. Adaptive trajectory tracking and stabilization for omnidirectional mobile robot with dynamic effect and uncertainties. *IFAC Proceedings Volumes*, 41(2):5383 – 5388, 2008. 17th IFAC World Congress.

[44] Lars Imsland and Johannes P Maree. Performance and stability for combined economic and regulatory control in mpc. *IFAC Proceedings Volumes*, 47(3):639–645, 2014.

[45] Tamás Kalmár-Nagy, Raffaello D'Andrea, and Pritam Ganguly. Near-optimal dynamic trajectory generation and control of an omnidirectional vehicle. *Robotics and Autonomous Systems*, 46(1):47 – 64, 2004.

[46] K. Kanjanawanishkul and A. Zell. Path following for an omnidirectional mobile robot based on model predictive control. In *2009 IEEE International Conference on Robotics and Automation*, pages 3341–3346, May 2009.

[47] Kiattisin Kanjanawanishkul. Mpc-based path following control of an omnidirectional mobile robot with consideration of robot constraints. *Advances in Electrical and Electronic Engineering*, 13(1):54–63, 2015.

[48] Kiattisin Kanjanawanishkul, Xiang Li, and Andreas Zell. Nonlinear model predictive control of omnidirectional mobile robot formations. *Intelligent Autonomous Systems 10, IAS 2008*, pages 41–48, 2008.

[49] Kiattisin Kanjanawanishkul and Andreas Zell. Path following for an omnidirectional mobile robot based on model predictive control. In *2009 IEEE International Conference on Robotics and Automation*, pages 3341–3346. IEEE, 2009.

[50] SS a Keerthi and Elmer G Gilbert. Optimal infinite-horizon feedback laws for a general class of constrained discrete-time systems: Stability and moving-horizon approximations. *Journal of optimization theory and applications*, 57(2):265–293, 1988.

[51] Hassan K Khalil and Jessy W Grizzle. *Nonlinear systems*, volume 3. Prentice hall Upper Saddle River, NJ, 2002.

[52] Michel Kinnaert. Adaptive generalized predictive controller for mimo systems. *International Journal of Control*, 50(1):161–172, 1989.

[53] Felipe Kuhne, Walter Fetter Lages, and JM Gomes Da Silva. Point stabilization of mobile robots with nonlinear model predictive control. In *IEEE International Conference Mechatronics and Automation, 2005*, volume 3, pages 1163–1168. IEEE, 2005.

[54] Denise Lam, Chris Manzie, and Malcolm Good. Application of model predictive contouring control to an x-y table. *IFAC Proceedings Volumes*, 44(1):10325 – 10330, 2011. 18th IFAC World Congress.

[55] Daniel B. Leineweber, Irene Bauer, Hans Georg Bock, and Johannes P. Schläder. An efficient multiple shooting based reduced sqp strategy for large-scale dynamic process optimization. part 1: theoretical aspects. *Computers and Chemical Engineering*, 27(2):157 – 166, 2003.

[56] J. Lévine. *Analysis and control of nonlinear systems: a flatness-based approach*. Math. Eng. Springer, Berlin, 2009.

[57] Xiang Li and Andreas Zell. *Motion Control of an Omnidirectional Mobile Robot*, pages 181–193. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[58] B. Lincoln and A. Rantzer. Relaxing dynamic programming. *IEEE Transactions on Automatic Control*, 51(8):1249–1260, 2006.

[59] Kevin M Lynch and Frank C. Park. *Modern Robotics: Mechanics, Planning, and Control*. Cambridge Univeristy Press, 2017.

[60] Lalo Magni, Giuseppe De Nicolao, Lorenza Magnani, and Riccardo Scattolini. A stabilizing model-based predictive control algorithm for nonlinear systems. *Automatica*, 37(9):1351–1362, 2001.

[61] Johannes Philippus Maree and Lars Imsland. Combined economic and regulatory predictive control. *Automatica*, 69:342–347, 2016.

[62] R Timothy Marler and Jasbir S Arora. The weighted sum method for multi-objective optimization: new insights. *Structural and multidisciplinary optimization*, 41(6):853–862, 2010.

[63] Bryan J Martin and James E Bobrow. Minimum-effort motions for open-chain manipulators with task-dependent end-effector constraints. *The international journal of robotics research*, 18(2):213–224, 1999.

[64] David Q Mayne and Hannah Michalska. Receding horizon control of nonlinear systems. *IEEE Transactions on automatic control*, 35(7):814–824, 1990.

[65] David Q Mayne, James B Rawlings, Christopher V Rao, and Pierre OM Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

[66] Mohamed W. Mehrez, George K.I Mann, and Raymond G. Gosine. Formation stabilization of nonholonomic robots using nonlinear model predictive control. In *Electrical and Computer Engineering (CCECE), 2014 IEEE 27th Canadian Conference on*, pages 1–6, May 2014.

[67] Mohamed W Mehrez, Karl Worthmann, Joseph PV Cenerini, Mostafa Osman, William W Melek, and Soo Jeon. Model predictive control without terminal constraints or costs for holonomic mobile robots. *Robotics and Autonomous Systems*, 127:103468, 2020.

[68] Mohamed W Mehrez, Karl Worthmann, George KI Mann, Raymond G Gosine, and Timm Faulwasser. Predictive path following of mobile robots without terminal stabilizing constraints. *IFAC-PapersOnLine*, 50(1):9852–9857, 2017.

[69] Mohamed W. Mehrez, Karl Worthmann, George K.I. Mann, Raymond G. Gosine, and Jürgen Pannek. Experimental speedup and stability validation for multi-step mpc. *IFAC-PapersOnLine*, 50(1):8698 – 8703, 2017. 20th IFAC World Congress.

[70] Hanna Michalska and David Q Mayne. Robust receding horizon control of constrained nonlinear systems. *IEEE transactions on automatic control*, 38(11):1623–1633, 1993.

[71] Richard M Murray, Zexiang Li, S Shankar Sastry, and S Shankara Sastry. *A mathematical introduction to robotic manipulation*. CRC press, 1994.

[72] Christopher Nielsen, Cameron Fulford, and Manfredi Maggiore. Path following using transverse feedback linearization: Application to a maglev positioning system. *Automatica*, 46(3):585–590, 2010.

[73] Christopher Nielsen and Manfredi Maggiore. On local transverse feedback linearization. *SIAM Journal on Control and Optimization*, 47(5):2227–2250, 2008.

[74] Frank C Park, James E Bobrow, and Scott R Ploen. A lie group formulation of robot dynamics. *The International journal of robotics research*, 14(6):609–618, 1995.

[75] S Joe Qin and Thomas A Badgwell. An overview of nonlinear model predictive control applications. In *Nonlinear model predictive control*, pages 369–392. Springer, 2000.

[76] Morgan Quigley, Ken Conley, Brian P. Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, 2009.

[77] T. Raff, S. Huber, Z. K. Nagy, and F. Allgöwer. Nonlinear model predictive control of a four tank system: An experimental stability study. In *Proceedings of the IEEE Conference on Control Applications*, pages 237–242, 2006.

[78] James B Rawlings and Kenneth R Muske. The stability of constrained receding horizon control. *IEEE transactions on automatic control*, 38(10):1512–1516, 1993.

[79] Marcus Reble and Frank Allgöwer. Unconstrained model predictive control and suboptimality estimates for nonlinear continuous-time systems. *Automatica*, 48(8):1812–1817, 2012.

[80] Jacques Richalet, A Rault, JL Testud, and J Papon. Model predictive heuristic control. *Automatica (Journal of IFAC)*, 14(5):413–428, 1978.

[81] Robotnik. SUMMIT- XL STEEL Datasheet, 2018.

[82] Raul Rojas and Alexander Gloye Forster. Holonomic Control of a robot with an omnidirectional drive . *KI - Kunstliche Intelligenz, BottcherIT Verlag*, 2006.

[83] J Anthony Rossiter. *Model-based predictive control: a practical approach*. CRC press, 2003.

[84] Patrick Saint-Pierre. Approximation of the viability kernel. *Applied Mathematics and Optimization*, 29(2):187–209, 1994.

[85] Claude Samson. Path following and time-varying feedback stabilization of a wheeled mobile robot. In *Proceedings of the international conference on advanced robotics and computer vision*, volume 13, pages 1–1, 1992.

[86] Gianluca Serale, Massimo Fiorentini, Alfonso Capozzoli, Daniele Bernardini, and Alberto Bemporad. Model predictive control (mpc) for enhancing building and hvac system energy efficiency: Problem formulation, applications and opportunities. *Energies*, 11(3):631, 2018.

[87] Roland Siegwart, Illah Reza Nourbakhsh, and Davide Scaramuzza. *Introduction to autonomous mobile robots*. MIT press, 2011.

[88] Roger Skjetne, Thor I Fossen, and Petar V Kokotović. Robust output maneuvering for a class of nonlinear systems. *Automatica*, 40(3):373–383, 2004.

[89] Mario Sznaier and Mark J Damborg. Suboptimal control of linear systems with state and control inequality constraints. In *26th IEEE conference on decision and control*, volume 26, pages 761–762. IEEE, 1987.

[90] Ogawa Takehiko and Kanada Hajime. Solution for ill-posed inverse kinematics of robot arm by network inversion. *Journal of Robotics*, 2010, 07 2010.

[91] Ton JJ van den Boom and TCPM Backx. Model predictive control. *DISC Course, Lecture Notes*, 16, 2010.

[92] Sergio Vazquez, Jose I Leon, Leopoldo G Franquelo, Jose Rodriguez, Hector A Young, Abraham Marquez, and Pericle Zanchetta. Model predictive control: A review of its applications in power electronics. *IEEE industrial electronics magazine*, 8(1):16–31, 2014.

[93] Andreas Wächter and Lorenz T Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1):25–57, 2006.

[94] Jacob Wilson, Meaghan Charest, and Rickey Dubay. Non-linear model predictive control schemes with application on a 2 link vertical robot manipulator. *Robotics and Computer-Integrated Manufacturing*, 41:23–30, 2016.

[95] Jon Woolfrey, Wenjie Lu, and Dikai Liu. A control method for joint torque minimization of redundant manipulators handling large external forces. *Journal of Intelligent & Robotic Systems*, 96(1):3–16, 2019.

[96] K. Worthmann, M. W. Mehrez, M. Zanon, G. K. I. Mann, R. G. Gosine, and M. Diehl. Model predictive control of nonholonomic mobile robots without stabilizing constraints and costs. *IEEE Trans. Control Syst. Technol.*, 24(4):1394–1406, July 2016.

[97] Karl Worthmann, Mohamed W Mehrez, Mario Zanon, George KI Mann, Raymond G Gosine, and Moritz Diehl. Regulation of differential drive robots using continuous time mpc without stabilizing constraints or costs. *IFAC-PapersOnLine*, 48(23):129–135, 2015.

APPENDICES

Appendix A

Stability of Nonlinear Systems

Using [51] as a guide, we present a summary of background work for the stability of non-linear systems. Within this chapter of the Appendix we specifically look at the asymptotic stability of a system. Stability is reviewed first for autonomous systems, then for non-linear non-autonomous systems using Lyapunov's theorems which are presented herein.

A.1 Lyapunov Stability for Autonomous Systems

We first consider an autonomous system, (autonomous here in the sense there is no control input)

$$\dot{x} = f(x) \tag{A.1}$$

where $f : X \to \mathbb{R}^n$ is a locally Lipschitz map, where $X \subset \mathbb{R}^n$ is an open subset containing $x = 0$. We can define an equilibrium point in $\mathbb{R}^n$ as follows

Definition A.1.1. *An equilibrium point of* (A.1)*,* $x = 0$ *is*

I stable if, for each $\varepsilon > 0$ *there is* $\varphi = \varphi(\varepsilon) > 0$ *such that*

$$\|x(0)\| < \varphi \implies \|x(t)\| < \varepsilon, \quad \forall t \geq 0$$

II unstable otherwise.

III asymptotically stable, if it is stable and φ *can be chosen such that*

$$\|x(0)\| < \varphi \implies \lim_{t\to\infty} x(t) = 0$$

Using definition A.1.1 of stability, we present Lyapunov's stability theorem.

Theorem A.1. *Consider an equilibrium point* $x = 0$*, for system* (A.1) *on a open and connected set* $X \subset \mathbb{R}^n$ *which contains* $x = 0$*. Let a continuously differential function* $V : X \to \mathbb{R}$ *satisfy*

$$V(0) = 0 \quad \textit{and} \quad V(x) > 0 \quad \forall x \in X \setminus \{0\} \tag{A.2}$$

$$\dot{V}(x) \leq 0, \quad \forall x \in X \tag{A.3}$$

Then, x=0 is a ***stable*** *equilibrium point. Furthermore, if*

$$\dot{V}(x) < 0, \quad \forall x \in X \setminus \{0\} \tag{A.4}$$

then x =0 is ***asymptotically stable****.*

We now want to examine the set of initial states in $x_0 \in X$ for which the equilibrium point $x = 0$ is asymptotically stable.This set of points is known as the basin or region of attraction. Using the Lyapunov function, this basin of attraction can be estimated. Given the a Lyapunov function V which satisfies the conditions presented in Theorem A.1 for asymptotic stability over a set X, if $\Omega_c = \{x \in \mathbb{R}_n | V(x) \leq c\}$ is bounded and contained in X, for some $c > 0$, then every trajectory starting in Ω_c will remain in Ω_c as $t \to \infty$. This in general produces conservative estimate of the region of attraction. Moreover, we present the Barbashin-Krasovskii theorem, which gives conditions where the region of attraction will be the entire space $\mathbb{R}^n$.

Theorem A.2. *Let* $x = 0$ *be an equilibrium point for* (A.1)*. Let* $V : \mathbb{R}^n \to \mathbb{R}$ *be a continuously differentiable function which meets the condition for asymptotic stability in Theorem A.1, additionally if*

$$\|x\| \to \infty \implies V(x) \to \infty$$

then $x = 0$ *is* ***globally asymptotically stable****.*

A.2 Comparison Functions

Looking forward to non-autonomous systems, we see that system $\dot{x} = f(t, x)$, starting at $x(t_0) = x_0$ now depends on both t and t_0. To ensure the asymptotic stability assumptions hold in the initial time t_0, we refine Definition A.1.1 with the use of comparison functions, known as the $\mathcal{K}$ and $\mathcal{KL}$ functions.

Definition A.2.1. *A continuous function $\eta : [0, a)\times \to [0, \infty)$ belongs to class $\mathcal{K}$ if it is strictly increasing and $\eta(0) = 0$. It belongs to class $\mathcal{K}_\infty$ if $a = \infty$ and $\eta(r) \to \infty$ as $r \to \infty$.*

Definition A.2.2. *A continuous function $\beta : [0, a) \times [0, \infty) \to [0, \infty)$ belongs to class $\mathcal{KL}$ if, for each fixed s, the mapping $\beta(r, s)$ belongs to class $\mathcal{K}$ with respect to r, and for each fixed r, the mapping $\beta(r, s)$ is decreasing with respect to s and $\beta(r, s) \to 0$ as $s \to \infty$.*

By presenting the following lemmas, we prepare to use these $\mathcal{K}$ and $\mathcal{KL}$ class functions stability analysis of nonautonomous systems.

Lemma 1. *Consider a continuous positive definite function $V : X \to \mathbb{R}^n$ defined over a domain $X \subset \mathbb{R}^n$ that contains $x = 0$. Let a ball $B_r = \{x \in X | d(x) < r\}$ be defined for some $r > 0$ and some distance function d. Then there exists class $\mathcal{K}$ functions, η_1 and η_2, defined on $[0, r]$ such that*

$$\eta_1(\|x\|) \leq V(x) \leq \eta_2(\|x\|)$$

for all $x \in B_r$. If $X = R^n$, functions η_1 and η_2 are defined over $[0, \infty)$ and the above inequality holds for all $x \in \mathbb{R}^n$. Additionally if $V(x)$ is radially unbounded, then η_1 and η_2 can be chosen to be of class $\mathcal{K}_\infty$.

Lemma 2. *Consider the differential equation*

$$\dot{y} = -\eta(y), \quad y(t_0) = y_0 \tag{A.5}$$

where η is a a locally Lipschitz class $\mathcal{K}$ function defined on $[0, a,)$. For all $0 \leq y_0 < a$, this equation has a unique solution $y(t)$, defined for all $t \geq t_0$. Moreover,

$$y(t) = \sigma(y_0, t - t_0) \tag{A.6}$$

where σ is a class $\mathcal{KL}$ function defined on $[0, a) \times [0, \infty)$.

We expand the definition for uniform stability using $\mathcal{K}$ and $\mathcal{KL}$ class functions and introduce uniform asymptotic stability, in the following Lemma.

Lemma 3. *The equilibrium point $x = 0$ of (A.1) is time invariant and*

I uniformly stable iff there exists a class $\mathcal{K}$ function η and a positive constant c, independent of t_0 such that

$$\|x\| \leq \eta(\|x_0\|), \quad \forall t \geq t_0, \quad \forall \|x(t_0)\| < c \tag{A.7}$$

II uniformly asymptotically stable iff there exists a $\mathcal{KL}$ function β and a constant $c > 0$, independent of t_0, such that

$$\|x(t)\| \leq \beta(\|x(t_0)\|, t - t_0), \quad \forall t \geq t_0, \quad \forall \|x(t_0)\| < c \tag{A.8}$$

III globally uniformly asymptotically stable iff (A.8) holds for any initial state x_0.

A.3 Non-Autonomous Systems

In this section, the results for autonomous system are extended to non-autonomous systems using comparison functions. Here we consider a non-autonomous system

$$\dot{x} = f(t, x) \tag{A.9}$$

where $f : [0, \infty) \times X \to \mathbb{R}^n$ is a piece-wise continuous mapping in t and locally Lipschitz in x on $[0, \infty) \times X$ and $X \subset \mathbb{R}^n$ is an open and connected set which contains the origin $x = 0$. Where the origin is an equilibrium point for (A.9) if

$$f(t, 0) = 0, \quad \forall t \geq 0.$$

Using this non-autonomous system, we can refine Definition A.1.1 to show the dependence of the stability behavior on the initial time t_0. We want to define stability of the origin as a uniform property with respect to the initial time.

Definition A.3.1. *The equilibrium point $x = 0$ of non-autonomous system (A.9) is*

I stable if, for each $\varepsilon > 0$, there exists a $\varphi = \varphi(\varepsilon, t_0)$ for all t_0 such that

$$\|x(t_0)\| < \varphi \implies \|x(t)\| < \varepsilon, \quad \forall t \geq t_0 \tag{A.10}$$

II uniformly stable if, for each $\varepsilon > 0$, there exists a $\varphi = \varphi(\varepsilon) > 0$, independent of t_0, such that (A.10) is satisfied.

III unstable otherwise.

IV asymptotically stable if conditions (A.10) are met and there is a positive $c = c(t_0)$ such that $x(t) \to 0$ as $t \to \infty$, for all $\|x_0\| < c$.

Using these definitions and lemmas, we can now present theorems which deal with the uniform stability and uniform asymptotic stability of these non-autonomous systems.

Theorem A.3. *Given that $x = 0$ is an equilibirum point of (A.9) and is contained within $X \subset \mathbb{R}^n$. Let $V : [0, \infty) \times X \to \mathbb{R}^n$ be a continuously differentiable function such that*

$$W_1(x) \leq V(t, x) \leq W_2(x) \tag{A.11}$$

$$\frac{\partial V}{\partial t} + \frac{\partial V}{\partial x} f(x, t) \leq 0 \tag{A.12}$$

$\forall t \geq 0$ and $\forall x \in X$, and where W_1 and W_2 are continuous positive definite functions. Then $x = 0$ is uniformly stable.

Theorem A.4. *Assuming that conditions on Theorem A.3 hold with inequality (A.3) constrained to*

$$\frac{\partial V}{\partial t} + \frac{\partial V}{\partial x} f(x, t) \leq -W_3(x) \tag{A.13}$$

$\forall t \geq 0$ and $\forall x \in X$, where $W_3(x)$ is a continuous positive definte function on X. Then equilibrium point $x = 0$ is uniformly asymptotically stable. Furthermore if r and c are chosen such that ball $B_r = \{\|x\| \leq r\} \subset X$ and $c < \min_{\|x\|=r} W_1(x)$, then every trajectory starting in $\{x \in B_r | W_2(x) \leq c\}$ satisfies

$$\|x(t)\| \leq \beta(\|x(t_0)\|, t - t_0), \quad \forall t \geq t_0 \tag{A.14}$$

for some $\mathcal{KL}$ class function β. Finally if $X = \mathbb{R}^n$ and $W_1(x)$ is radially unbounded, then $x = 0$ is globally uniformly asymptotically stable.

Though the theorems in this section are presented without proof, these can be found in [51]. These theorems form the basis of Lyapunov stability for nonlinear, non-autonomous systems which are dealt with in this paper. In the next sections we present stability works which build on these theorems to describe the stability behaviour of systems using MPC.

Appendix B

Holonomic Mobile Robot Lower Kinematic Model

In this section of the appendix, we show the derivation of the mecanum wheel holonomic robot lower level kinematics as presented in [59]. For the lower level forward kinematics, the objective is to define an expression which gives the chassis velocities for given wheel speeds. Let us first consider the wheel model shown in Fig. B.1, here we can express the linear velocity of the wheel as

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = v_{drive} \begin{bmatrix} 1 \\ 0 \end{bmatrix} + v_{slide} \begin{bmatrix} -\sin\gamma \\ \cos\gamma \end{bmatrix} \tag{B.1}$$

where $\gamma = \pm 45°$ is the free sliding direction relative to the driven direction. By solving these equations for the driven and sliding speeds, v_{drive} and v_{slide}, we obtain

$$\begin{aligned} v_{drive} &= v_x + v_y \tan\gamma \\ v_{slide} &= v_y / \cos\gamma. \end{aligned}$$

We can obtain an expression for the angular speed of the wheel u_w for a given wheel radius r,

$$u_w = \frac{v_{drive}}{r} = \frac{1}{r}(v_x + v_y \tan\gamma) \tag{B.2}$$

If we consider a wheel in relation to a body fixed frame on the robot $\{b\}$, the wheel i is located at (x_i, y_i) and the relative orientation of the wheel's drive direction to the body's orientation is β. We identify the robot's frame $\{b\}$ relative to a fixed frame $\{s\}$ with coordinates (x, y) and orientation ϕ. This is depicted in Fig. B.2. We express the robots

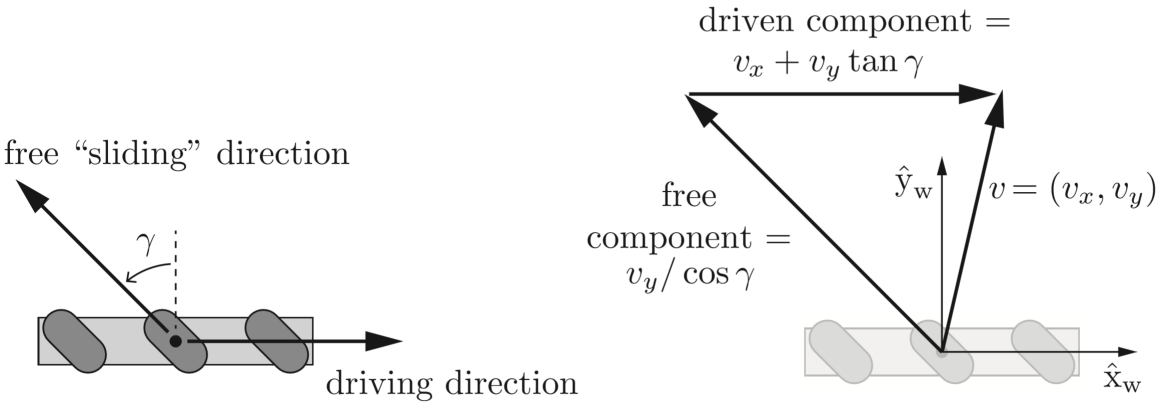

Figure B.1: Left: The driving direction and the free sliding direction allowed by the rollers. For a mecanum wheel we typically have $\gamma = \pm 45°$ Right: A given wheel velocity $v = (v_x, v_y)$ expressed in terms of its driven and free sliding components. These are expressed in the wheel frame. This figure is reprinted from [59].

position in the fixed frame as $q = (\phi, x, y)^\top$ and its corresponding velocity as $\dot{q}$. We can relate the velocity of the robot in the fixed frame with its velocity in the robot frame $\mathcal{V}_b = (\omega_{bz}, v_{bx}, v_{by})^\top$, with the following relationship

$$\dot{q} = \begin{bmatrix} \dot{\phi} \\ \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \omega_{bz} \\ v_{bx} \\ v_{by} \end{bmatrix}. \tag{B.3}$$

We consider this the higher level kinematics for a holonomic robot. Next we consider the relationship between the wheel input speeds and the robots velocity in the $\{b\}$ frame. Here we consider a holonomic robot with a four mecanum wheel configuration as shown in Fig. B.3. For the considered robot the relative orientation of the wheels in $\{b\}$ is $\beta_i = 0$. We can therefore define the relationship between the individual wheel speeds $u_{w,i}$ and the robot velocity $\mathcal{V}_b$ as

$$u_{w,i} = h_i \mathcal{V}_b = \frac{1}{r_i \cos\gamma_i} \begin{bmatrix} x_i \sin\gamma_i + y_i \cos\gamma_i \\ \cos\gamma_i \\ \sin\gamma_i \end{bmatrix}^\top \mathcal{V}_b \tag{B.4}$$

For the considered robot shown in Fig. B.3, we have the parameters shown in Table B. Using these parameters we can determine the relation ship h_i for each of the four wheels.

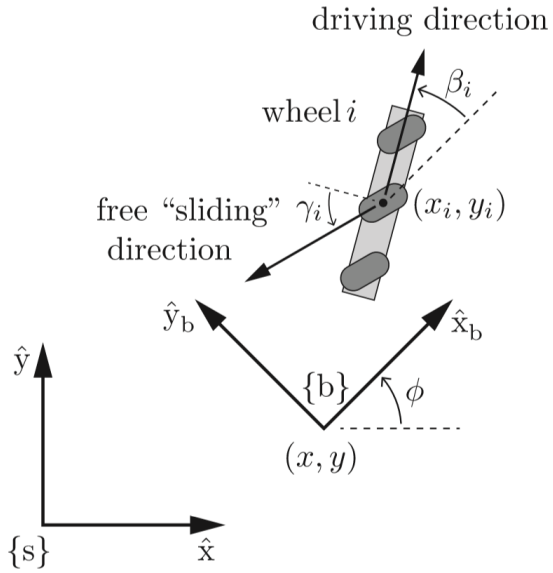

Figure B.2: A fixed frame $\{s\}$, chassis frame $\{b\}$ located at (ϕ, x, y) in $\{s\}$ and wheel i in frame $\{b\}$ located at (x_i, y_i) and driving direction β_i and sliding direction γ_i. This figure is reprinted from [59].

Once we determine the relationship h_i, we can stack these vectors together to build a matrix H, this relates the velocity $\mathcal{V}_b$ to a vector of wheel speeds $u_w = (u_{w,1}, u_{w,2}, u_{w,3}, u_{w,4})^\top$. This produces the lower level kinematic model

$$u_w = \begin{bmatrix} u_{w,1} \\ u_{w,2} \\ u_{w,3} \\ u_{w,4} \end{bmatrix} = H\mathcal{V}_b = \frac{1}{r} \begin{bmatrix} -\ell - w & 1 & -1 \\ \ell + w & 1 & 1 \\ \ell + w & 1 & -1 \\ -\ell - w & 1 & 1 \end{bmatrix} \begin{bmatrix} \omega_{bz} \\ v_{bx} \\ v_{by} \end{bmatrix}. \tag{B.5}$$

Having a lower level kinematic model of the robot is important as we can use it for control at the individual motor level, alternatively if we use the higher level kinematic model (B.3), the lower level kinematics can still be used to ensure that the individual motors do not exceed their rated speeds.

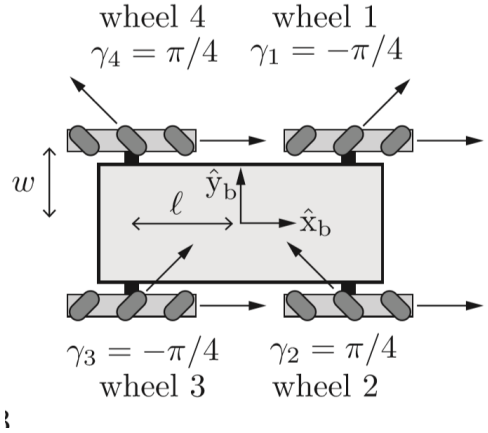

Figure B.3: Kinematic models for a mobile robot with four mecanum wheels. This figure is reprinted from [59].

Table B.1: Parameters for holonomic robot with 4 mecanum wheels.

Wheel (i)	γ_i	x_i	y_i	r_i
1	$-\pi/4$	ℓ	w	r
2	$\pi/4$	ℓ	$-w$	r
3	$-\pi/4$	$-\ell$	$-w$	r
4	$\pi/4$	$-\ell$	w	r

Appendix C

Geometric Background for Robot Dynamics

In this section of the appendix, we present some of the geometric background information on the *Special Euclidean Group* $SE(3)$, its adjoint mappings and how they relate to generalized velocities and torques. This follows the presentation in [74].

C.1 $SE(3)$ and $se(3)$

For the purpose of this thesis, we consider $SE(3)$ as a group of rigid body motion described by

$$\begin{bmatrix} \Theta & \mathbf{b} \\ 0 & 1 \end{bmatrix} \tag{C.1}$$

where $\Theta \in SO(3)$ is a 3×3 rotation matrix in the *Special Orthogonal Group* and $\mathbf{b} \in \mathbb{R}^3$ is a translation. We will employ a simplified notation for $SE(3)$ denoted by the ordered pair $(\Theta, \mathbf{b})$, with the group multiplication $(\Theta_1, \mathbf{b}_1) \cdot (\Theta_2, \mathbf{b}_2) = (\Theta_1\Theta_2, \Theta_1\mathbf{b}_2 + \mathbf{b}1)$. The *Lie algebra* of $SE(3)$, denoted $se(3)$ is in the form

$$\begin{bmatrix} [\omega] & \mathbf{v} \\ 0 & 0 \end{bmatrix} \quad \text{where} \quad [\omega] = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}. \tag{C.2}$$

We note that $[\omega] \in so(3)$ denotes the Lie algebra of $SO(3)$. $[\omega] \in so(3)$ can also be regarded as a vector $\omega \in \mathbb{R}^3$ and is commonly denoted as ω. Elements of $se(3)$ will be denoted as $(\omega, \mathbf{v}) \in \mathbb{R}^6$.

An element of a Lie group has a linear mapping to its Lie algebra through the *adjoint representation*. For a given matrix Lie group, $\mathbf{G}$ with Lie algebra $\mathbf{g}$, for every $\mathbf{X} \in \mathbf{G}$ the adjoint map $\mathrm{Ad}_{\mathbf{X}} : \mathbf{g} \to \mathbf{g}$ is defined by $\mathrm{Ad}_{\mathbf{X}}(\mathbf{x}) = \mathbf{X}\mathbf{x}\mathbf{X}^{-1}$. If $\mathbf{X} = (\Theta, \mathbf{b}) \in SE(3)$ then its adjoint map acting on $\mathbf{x} = (\omega, \mathbf{v}) \in se(3)$ is

$$\mathrm{Ad}_{\mathbf{X}}(\mathbf{x}) = (\Theta\omega, \mathbf{b} \times \Theta\omega + \Theta\mathbf{v}) = \begin{bmatrix} \Theta & 0 \\ [\mathbf{b}]\Theta & \Theta \end{bmatrix} \begin{bmatrix} \omega \\ \mathbf{v} \end{bmatrix}. \tag{C.3}$$

Other useful properties of the adjoint mapping include that $\mathrm{Ad}_{\mathbf{X}}^{-1} = \mathrm{Ad}_{\mathbf{X}^{-1}}$ and $\mathrm{Ad}_{\mathbf{X}}\mathrm{Ad}_{\mathbf{Y}} = \mathrm{Ad}_{\mathbf{XY}}$ for any $\mathbf{X}, \mathbf{Y} \in SE(3)$. We also consider the dual adjoint operator $\mathrm{Ad}_{\mathbf{X}}^{*} : se(3)^{*} \to se(3)^{*}$, where $se(3)^{*}$ denotes the the dual basis on $se(3)$, given by the transpose of $\mathrm{Ad}_{\mathbf{X}}$. For a given $\mathbf{z} = (\mathbf{m}, \mathbf{f}) \in se(3)^{*}$,

$$\mathrm{Ad}_{\mathbf{X}}^{*}(\mathbf{z}) = \begin{bmatrix} \Theta^{\top} & \Theta^{\top}[\mathbf{b}]^{\top} \\ 0 & \Theta^{\top} \end{bmatrix} \begin{bmatrix} \mathbf{m} \\ \mathbf{f} \end{bmatrix}. \tag{C.4}$$

There also exists a linear map between an element of a Lie algebra and its Lie algebra using the Lie bracket. We consider an element $\mathbf{x} \in \mathbf{g}$, the adjoint representation is the map $\mathrm{ad}_{\mathbf{x}} : \mathbf{g} \to \mathbf{g}$ defined by $\mathrm{ad}_{\mathbf{x}}(\mathbf{y}) = [\mathbf{x}, \mathbf{y}]$, where $[\cdot, \cdot]$ denotes the Lie bracket. For a given $\mathbf{x} = (\omega_1, \mathbf{v}_1), \mathbf{y} = (\omega_2, \mathbf{v}_2) \in se(3)$

$$\mathrm{ad}_{\mathbf{x}}(\mathbf{y}) = (\omega_1 \times \omega_2, \omega_1 \times \mathbf{v}_2 - \omega_2 \times \mathbf{v}_1) = \begin{bmatrix} [\omega_1] & 0 \\ [\mathbf{v}_1] & [\omega_1] \end{bmatrix} \begin{bmatrix} \omega_2 \\ \mathbf{v}_2 \end{bmatrix}. \tag{C.5}$$

In a similar way the dual adjoint operator $\mathrm{ad}_{\mathbf{x}}^{*} : se(3)^{*} \to se(3)^{*}$ is defined as a

$$\mathrm{ad}_{\mathbf{x}}^{*}(\mathbf{z}) = \begin{bmatrix} -[\omega_1] & -[\mathbf{v}_1] \\ 0 & -[\omega_1] \end{bmatrix} \begin{bmatrix} \mathbf{m} \\ \mathbf{f} \end{bmatrix}. \tag{C.6}$$

C.2 Generalized Velocities and Forces

We now consider the tangent vector $\dot{\mathbf{X}}(t)$ of a curve $\mathbf{X} = (\Theta(t), \mathbf{b}(t)) \in SE(3)$ which is an element of $se(3)$. Given a described motion $\mathbf{X}(t)$ for a rigid body in reference to an inertial reference frame, the body-fixed velocity representation of $\dot{\mathbf{X}}$ is given by $\mathbf{X}^{-1}\dot{\mathbf{X}} = (\Theta^{-1}\dot{\Theta}, \Theta^{-1}\dot{\mathbf{b}})$. Here the angular and translational velocities of the rigid body relative to its fixed frame are $\Theta^{-1}\dot{\Theta}$ and $\Theta^{-1}\dot{\mathbf{b}}$ respectively. If $\mathbf{X}(t)$ undergoes a transformation $\mathcal{T} \in SE(3)$ such that $\mathbf{X}(t) \mapsto \mathbf{X}(t)\mathcal{T}$, then its transformed body-fixed velocity $\mathbf{V}_b$ is

$$\mathbf{V}_b = \mathcal{T}^{-1}\mathbf{X}^{-1}\dot{\mathbf{X}}\mathcal{T} = \mathrm{Ad}_{\mathcal{T}^{-1}}(\mathbf{X}^{-1}\dot{\mathbf{X}}) \tag{C.7}$$

Conversely, forces and moments can be considered elements of the dual set $se(3)^*$. We view a moment-force pair $(\mathbf{m}, \mathbf{f}) \in se(3)^*$ as consisting of a skew symmetric matrix of moments and a three-vector of forces

$$[\mathbf{m}] = \begin{bmatrix} 0 & -m_3 & m_2 \\ m_3 & 0 & -m_1 \\ -m_2 & m_1 & 0 \end{bmatrix}, \qquad \mathbf{f} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} \tag{C.8}$$

If we want to consider this moment-force pair under a right transformation $\mathcal{T}(\Theta, \mathbf{b}) \in SE(3)$, we can express the transformed pair $(\mathbf{m}', \mathbf{f}')$ as

$$\begin{bmatrix} \mathbf{m}' \\ \mathbf{f}' \end{bmatrix} = \begin{bmatrix} \Theta^\top & \Theta^\top [\mathbf{b}]^\top] \\ 0 & \Theta^\top \end{bmatrix} \begin{bmatrix} \mathbf{m} \\ \mathbf{f} \end{bmatrix} = \mathrm{Ad}^*_{\mathcal{T}}(\mathbf{m}, \mathbf{f}) \tag{C.9}$$

Having identified these relationships we can now use the adjoint maps to transform generalized velocities (twists) and generalized forces (wrenches) between coordinate frames.