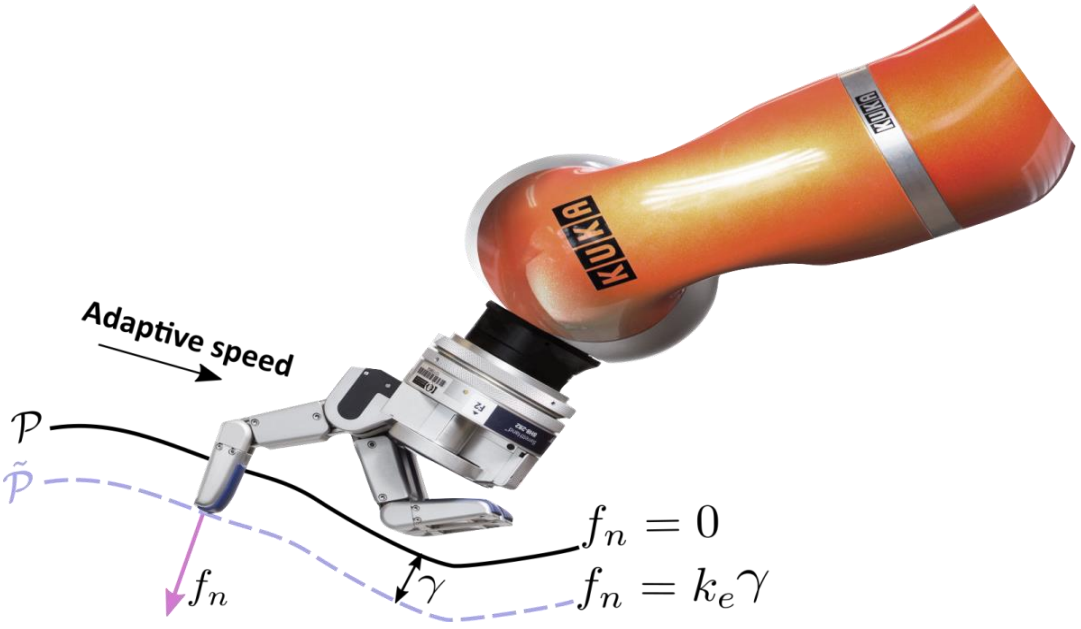
OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG

CONTRIBUTIONS IN SYSTEMS THEORY AND AUTOMATIC CONTROL

Khalid J. Kazim

Towards a Unified Approach for Path-following and Force-feedback Using Nonlinear Model Predictive Control



Towards a Unified Approach for Path-following and Force-feedback Using Nonlinear Model Predictive Control

Dissertation
zur Erlangung des akademischen Grades
Doktoringenieur (Dr.-Ing.)

von
Khalid J. Kazim
geboren am 5. Juni 1981 im Bagdad

genehmigt durch die Fakultät für Elektrotechnik und Informationstechnik der
Otto-von-Guericke-Universität Magdeburg

Gutachter:
Prof. Dr.-Ing. Rolf Findeisen
Prof. Dr. rer. nat. Frank Ortmeier
Prof. Dr.-Ing. Sandra Hirche

eingereicht am 22. Juni 2016
Promotionskolloquium am 14. Februar 2017

Acknowledgements

This thesis is the result of my PhD research work conducted in the Systems and Control Group of Prof. Dr.-Ing. Rolf Findeisen at the Institute for Automation Engineering, Faculty of Electrical Engineering and Information Technology at Otto-von-Guericke-Universität Magdeburg, Germany.

Firstly, I would like to express my gratitude to my supervisor Prof. Dr.-Ing. Rolf Findeisen for the support of my PhD study, for his motivation, and immense knowledge. His advises helped me all the time during research and writing of this thesis. Furthermore, I would like to thank Prof. Dr.-Ing. Sandra Hirche from Technical University of Munich, Germany, and Prof. Dr. rer. nat. Frank Ortmeier from Otto-von-Guericke-Universität Magdeburg, Germany for their willingness to act as reviewers of this thesis.

The members of the Systems and Control group have been a source of good advice and collaboration. I would especially like to acknowledge Dr. Philipp Rumschinski. I am grateful for the time spent with my colleagues, Janine Matschek, Petar Andonov, Juan Pablo Menendez Zometa, Markus Kögel and Tobias Bäthge.

I am grateful to our group's administrative assistants Ulrike Thürmer and Peggy Stein who kept us organized and were always there to help.

I gratefully acknowledge the funding sources, I was funded by Iraqi-German scholarship program, which is managed jointly by the Ministry of Higher Education and Scientific Research in Iraq and the German Academic Exchange Service (DAAD).

I would like to thank my family; my parents and to my brothers and sisters for supporting me spiritually throughout writing this thesis and my life in general.

February 2017 Khalid J. Kazim

Contents

Abstract

In many robotics tasks, one needs to follow a specified path while applying a controlled force. Typically, these tasks are handled by controlling motion and force separately. Existing approaches that solve the path-following and force-control problems simultaneously do not consider constraints explicitly. In addition, these approaches require often a special structure of the dynamic system.

In this work, we propose a unified approach to handle path-following and force-control subject to input and state constraints. To this end, we extend an existing nonlinear model predictive path-following control scheme to include force control.

In the standard nonlinear model predictive path-following control scheme, a timing law is added to the optimal control problem that decides which point on the path is set as the current reference. To realize additionally a desired force, a so-called admittance dynamics is added to the optimal control problem as a secondary constraint (virtual state). The force error, that drives this dynamic, updates the followed path to achieve the desired force. Since both the timing law and the admittance dynamics are included as constraints in the optimization problem, the nonlinear model predictive controller works as a coordinator between path-following and force-control. Hence, the proposed scheme allows to compromise between both tasks by tuning the corresponding weights in the cost function of the optimal control problem.

The approach is tested, in simulation and furthermore experimentally validated in real-time with a lightweight robot arm. Simulation and experimental validation underpin the applicability of the proposed concept. The presented approach sets the basis for completely new force feedback control strategies, reaching from cooperative control towards man-machine interaction subjected to constraints.

Uncontrolled contact of the robot with the environment has to be avoided, because this can generate undesired forces, which may lead to instability. Therefore, we propose an approach to determine the set of initial conditions for which an uncontrolled collision occurs no matter which input is applied. To this end, we developed a set-based recursive procedure to compute the inevitable set. The applicability of this set is not limited to robotic manipulator systems, but can be used in mobile swarm robots and unmanned aerial vehicles.

Deutsche Kurzfassung

In vielen Aufgaben in der Robotik z.B. im häuslichen, industriellen und medizinischen, Umfeld soll ein bestimmter Pfad verfolgt werden, während eine kontrollierte Kraft auf die Umgebung ausgeübt wird. Existierende Regelungsstrategien betrachten dabei Bewegung und Kraft getrennt voneinander, was jedoch die Regelgüte begrenzt. Es existieren zwar bereits Ansätze, die die Pfadverfolgungs und Kraftregelungs aufgabe gleichzeitig lösen, jedoch erlauben diese keine explizite Berücksichtigung von Beschränkungen.

Zusätzlich erfordern diese Ansätze eine spezielle Struktur des betrachteten dynamischen Systems, da sie auf Linearisierungstechniken beruhen. Aufgrund von Sicherheitsanforderungen, ist es jedoch notwendig die auferlegten Beschränkungen unbedingt einzuhalten. In dieser Arbeit wird ein Ansatz zur kombinierten Kraft- und Pfadverfolgungsregelung unter Beschränkungen vorgeschlagen. Zu diesem Zweck ergänzen wir einen nichtlinearen modell-prädiktiven Pfadverfolgungsansatz um eine Kraftformulierung unter Beschränkungen. Hierbei wird die gewünschte Admittanz −die Dynamik der Kraftregelung− dem Optimalsteuerungsproblem als zusätzlicher virtueller Zustand hinzugefügt. Die Admittanz wird hierbei dynamisch über einen Kraftfehler bestimmt, um eine (virtuelle) Referenzbahn zu erzeugen. Somit wird der Pfad dem der Roboter folgen soll, durch eine virtuelle Referenzbahn vorgegeben. Im Fall präzisem Vorwissens kann die Kraftsteuerung entlang des aktualisierten Pfads exakt erreicht werden.

Im vorgeschlagenen Ansatz übernimmt die nichtlineare modell-prädiktive Regelung die Koordination zwischen Pfadverfolgung und Kraftregelung, da sowohl die zeitliche Taktung der Pfadverfolgung als auch die Admittanz-Dynamik als Beschränkungen im Optimierungsproblem berücksichtigt werden. Die Formulierung als nichtlineare modell-prädiktive Regelung erlaubt einen Kompromiss zwischen der Positionsregelung und der Erreichung der Kontaktkraft, da die Gewichtungen zwischen Kraft und Pfad in der Kostenfunktion des Optimalsteuerungsproblems eingestellt werden können.

Der vorgeschlagene Ansatz wird zunächst in Simulationen am Modell eines Leichtbauroboters getestet. In einem zweiten Schritt werden die Ergebnisse mittels Echtzeit-Experimenten an einem realen Roboter validiert. Die simulierten und experimentellen Ergebnisse zeigen die echtzeitfähige Anwendbarkeit des vorgeschlagenen Konzeptes.

Ungeregelter Kontakt des Roboters mit der Umgebung muss vermieden werden, da dies zu Instabilitäten führen kann. Deshalb schlagen wir einen Ansatz vor, um die Anfangsbedingungen zu bestimmen, für die eine Kollision nicht vermieden werden

kann egal welches Eingangssignal gewählt wird. Zu diesem Zweck wird eine mengenbasierte Methode entwickelt, die diese Anfangsbedingungen bestimen. Dieser Ansatz ist nicht auf Roboter beschränkt, sondern kann für mobile Schwärme und unbemannten Flugzeuge verwendet werden.

List of abbreviations and symbols

Abbreviations and Acronyms

B&B	Branch and Bound Algorithm
LP	Linear Program
MPC	Model Predictive Control
MPFC	Model Predictive Path-following Control
NMPC	Nonlinear Model Predictive Control
MILP	Mixed-integer Linear Program
OCP	Optimal Control Problem
ODE	Ordinary Differential Equation
QP	Quadratic Program

Symbols

t	Time variable
t_i	Recalculation time instant
θ	Scalar path parameter
n_x	Dimension of the real valued state vector
n_y	Dimension of the real valued output vector
n_u	Dimension of the real valued input vector
x	State vector $x \in \mathbb{R}^{n_x}$
y	Output vector $y \in \mathbb{R}^{n_y}$
u	Input vector $u \in \mathbb{R}^{n_u}$
x_0	Vector of the real valued initial state
$\hat{x}$	Vector of the real valued virtual state
f	Vector field for the system dynamics
$x(\tau, u(\cdot)\|x(t_0))$	The state trajectory starting from x_0 at time t_0 driven by an input signal $u(\cdot) : [t_0, t_1] \to U$ and $\tau \in [t_0, t_1]$
$\mathcal{X}$	Set defining the state constraints $\mathcal{X} \subseteq \mathbb{R}^{n_x}$
$\mathcal{U}$	Set defining the input constraints $\mathcal{U} \subseteq \mathbb{R}^{n_u}$
$\hat{\mathcal{X}}$	Set defining the virtual state constraints $\hat{\mathcal{X}} \subseteq \mathbb{R}^{\hat{r}+1}$ where is a well-defined vector relative degree
$\mathbf{I}^{n\times n}$	Identity matrix with dimension $\mathbb{R}^{n\times n}$
$\mathbf{0}^{n\times m}$	Zero matrix with dimension $\mathbb{R}^{n\times m}$
C^k	Set of k-times continuously differentiable functions

$intA$	Interior of a set A
∂A	Boundary of a set A
$\mathcal{K}$	Set of class $\mathcal{K}$ functions, see Appendix A
$\mathcal{L}$	Set of class $\mathcal{L}$ functions, see Appendix A
$\mathcal{KL}$	Set of class $\mathcal{KL}$ functions, see Appendix A
$L_f h$	Lie derivative of h along f, i.e. the directional derivative of h along f
$Q \geq 0$	Positive semi-definite matrix Q, see Appendix A
$R > 0$	Positive semi-definite matrix Q, see Appendix A
$\|x\|$	L^2-norm of a vector $x \in \mathbb{R}^n$
$\|x\|_Q^2$	Notation for $x^T Q x$, $Q \geq 0$
$\bar{x}$	Predicted states vector
$\bar{y}$	Predicted output vector
$\bar{u}$	Predicted inputs vector
$J(\cdot)$	Cost-function of an optimal control problem
$V(x(t_i))$	Optimal value of the cost-function for a measured state x at time t_i
$u^*(\tau, x(t_i))$	Optimal input sequence, with $\tau \in [t_i, t_{ii}]$, $t_{ii} > t_i$ starting at initial state $x(t_i)$
$\mathcal{L}_i$	The set of all points on link i

1 Introduction

1.1 Overview of Path-following with Force Feedback

To perform many of their assigned tasks, *industrial* and *domestic* robots must follow a specific path while applying a force on or along this path. For example, during machining processes, such as sawing, deburring, grinding, soldering, gluing and milling; robots are normally required to follow a defined path and maintaining a desired force. Another example is a humanoid hand, where the fingers of the robot hand should grasp an object and move it along a specified path, (see Figure 1.1), and e.g. for writing (see Figure 1.2). Similar tasks also appear in cooperative robotics tasks, for instance, when two robot arms cooperatively should move an object along a specific path while grabbing it with a controlled force.

Typically, manipulation tasks during interaction with an environment are described in terms of position and force. In general, methods to control constrained manipulators can be classified into two main categories: hybrid controls and impedance controls (for more details see Chapter 3). Hybrid position/force control allows a robot to apply forces to a constrained direction while moving along a specified path in free direction. Such that, the task space is divided into orthogonal subspaces, each of which is either assigned as a position or a force control subspace [30]. Impedance control methods provide a dynamic relation between the robot's end-effector and the environment, i.e. the desired impedance. Hence, the main objective in impedance control is to control in each direction on the task space the dynamic relation between motion error and contact force. Impedance controllers have an intrinsic robustness to environment modeling mismatch [23, 78].

(a) (b)

Figure 1.1: Examples of KUKA LWR IV manipulating an object.

Figure 1.2: KUKA LWR IV writing. [38]

These outlined methods can be used to control along each direction either position or force or a compromise between them. Alternatively, [46] proposed a simultaneous approach controls independently transversal and tangential forces to the path. This is achieved by decomposing the system into linear transversal and tangential subsystems using feedback linearization. As stated in 2014 by Flixeder et al. [46]: "To the authors knowledge, no attempt has been made so far to independently control both, the force transversal and tangential to the machining path". While [46] controls both forces along the path, constraints can only be considered indirectly.

We present a unified approach to the path-following and force-feedback control problems simultaneously using model predictive control. In our approach, constraints on states and inputs are considered. The solution is preformed within an optimization setting. To present the approach, we describe the components used. We begin by a summary of the advantages of Nonlinear Model Predictive Control, Path-following and Force-feedback Control. Then, we describe the proposed approach, which combines these three control strategies to solve path-following and force-feedback simultaneously.

Nonlinear Model Predictive Control

Nonlinear Model Predictive Control (NMPC) [20, 51, 79, 84, 86] is an advanced control method that has been used majorly for process control applications, i.e. oil refineries, reactors, and chemical plants. In NMPC, optimization techniques are used to solve an optimal control problems within a specified period of time, the so-called prediction window or horizon. In this strategy, the current time horizon is optimized while taking the future system evolution into account. This can be done by using the system model to simulate - i.e. predict - the future system behavior. According to these predictions the input is optimized, so as to minimize a control objective while still meeting the constraints. The initially obtained optimal input is applied, and the

procedure of optimization/prediction is repeated at each sampling instant using the new measurements or estimations as an initial condition (for more detail see Chapter 2).

MPC has been used in the field of robotics, e.g. in robot locomotion [49, 63, 81, 117, 124], in robot manipulation [73], or controlling a parallel manipulator [32].

The main advantages of NMPC compared to other control strategies are: It allows to handle highly nonlinear and stiff dynamics, treats explicitly states and inputs constraints and its ease-to-use for multivariable processes [58, 79, 84]. Furthermore, it allows to online improving performance based-on cost function criterion, and achieves robustness with respect to changes in system parameters [1, 58, 97].

Path-following

In path-following problem, the system output is supposed to follow a geometric path without a prespecified time reference. The time-evolution along the path is considered as an additional degree of freedom [5, 34–37, 55, 74, 89, 111, 127]. Path-following problems appear in a wide range of applications e.g. process engineering [50], robotics [26, 109, 113], and autonomous aircrafts [25], ships [52, 87], and vehicles [4].

Path-following has, compare to reference tracking several advantages. As known, good tracking can be achieved when there is no unstable zero-dynamics (i.e. non-minimum phase systems). If a system is non-minimum phase, then the tracking error is no longer achievable, since part of the energy is required for stabilization [3]. This limitation is structural. It can be only overcome when the structure of the system is changed or the tracking problem is reformulated [3]. One possible reformulation of the tracking problem is to decompose it into two sub-problems: first, a geometric path-following problem, second, following the path with satisfaction of time, speed, or acceleration assignment along the path [3, 111]. The main idea, is to use a path-parameter −which is used to parametrize the geometric path− as a new additional virtual control input to stabilize the non-minimum phase dynamics, while the system control variables drive the system on the path. However, the dominant methods for solving path-following problems are geometric and Lyapunov based nonlinear feedback control [11, 88, 111]. Basically, These methods do not allow to consider the constraints on states and inputs in a structured way [34]. Alternatively, predictive control can be used to solve the path-following problem while taking stability and constraints into account [34, 36, 127] (for more details see Chapter 2).

Force Feedback Control

Force feedback control aims to control the force applied to an (possibly soft) object. In the force feedback control two major approaches exist: *Passive Compliance* and *Active Compliance* control [110]. Passive compliance control is induced inherently by the structural mechanical compliance of the robot manipulator, i.e. finite stiffness of the

links, grippers, joints, and actuators. Normally, in passive compliance, the measurement of force is not needed [10, 110]. In active compliance, the interaction between the end-effector and the environment is controlled by changing the joint stiffness using a user-defined control law.

Active-interaction control can be classified into two groups: indirect-force control and direct-force control [53, 108, 110, 128]. In indirect-force control the control is achieved implicitly by motion control, without an explicit force feedback loop. Whereas, in the direct force control, the contact force with respect to the desired value is controlled using a force feedback control loop (for more detail see Chapter 3). Indirect force control approach is typically realized by so-called impedance/admittance control. The aim of the impedance/admittance control is to design a desired dynamic interaction between the end-effector and the environment [24, 57, 70, 107]. Any active compliance approach receives both motion and force errors, to provide the proper input to the robot joint actuators.

Impedance/admittance control is realized by a virtual mass−spring−damper dynamic system with adjustable parameters [57, 107, 110]. The active (indirect) compliance is an impedance if the control reacts to the position error by generating forces, while it is called an admittance if the control complies to contact forces by producing a deviation from the desired position. We adopt admittance force control, as our approach updates the desired position based on the measured force. Admittance control is well suited for applications where position accuracy is relevant and the environment is not highly stiff [6]. For low stiffness environment, admittance control can be only used if the damping coefficient is significantly increased, However, to avoid unstable behavior −due to low bandwidth− during interaction with stiff environment, the velocity (of the robot) needs to be excessively reduced [6].

1.2 Path-following and Force-feedback Based-on Predictive Control

We propose a unified approach to handle both path-following and force-feedback problems simultaneously. To achieve this, we reformulate the *Nonlinear Model Predictive Path-following Control* scheme introduced in [34, 36, 127] by including a desired admittance dynamics as an additional constraint (virtual state). The admittance is driven by the force error to generate a (virtual) reference trajectory, which updates the path to be followed by the robot. Considering a perfect following of the path, force control is then achieved by following the updated path.

The inaccuracy in the modeling of both the robot dynamic model and the environment are making trajectory-tracking of a desired position or force challenging or even impossible. Therefore, considering path-following instead of trajectory-tracking (see

Chapter 2, Section 2.1.4), provides the freedom to additionally change the timing along the desired path. This freedom in time facilitates the tasks of real-time adaptive motion control during interaction while applying a desired force. The proposed approach is discussed in detail in Chapter 4.

1.3 Contributions

The main contributions of this thesis are:

1. We designed a unified approach to handle path-following and force control problems simultaneously considering constraints on state and input.

2. Both force regulation and path-following problems are solved within one optimization problem.

3. The proposed approach can be used during free motion and constrained motion, i.e. interaction with the environment, as well as, for known and unknown environments as explained in Chapter 4 & 5.

4. By using a MPC strategy, the future evolutions of a parametrized-path, the force trajectory, and the dynamic system are taking into account during the optimization problem. By observing the proposed optimization problem, the performance index penalizes the deviations of the system output from the path and the parametrization variable from its final value. Increasing the weight in the performance index on the deviation from the path, one can prioritize convergence to the path, i.e. the evolution of parametrization variable slows down when the deviation from the path is big. This property ensures fast convergence to the desired force as shown in the simulation and experimental results.

5. The receding horizon feature of the predictive scheme provides fast and stable convergence, since, the dynamic model is used to predict the future evolution and the force effect.

6. The proposed approach can be used to control the force along a normal, tangential, and bi-normal vector to the path, by imposing a desired admittance dynamic along a corresponding direction.

7. The proposed approach can be expanded to handle time delays in measurements as validated in the simulation and experimental studies.

8. Both tangential and transversal forces can be controlled while moving the robot arm manually along a specified path by a human operator e.g. master and slave robotics systems.

9. We furthemore propose an approach to determine the set of initial conditions the so-called *−inevitable set−* for which an uncontrolled collision occurs no matter which input is applied. By knowing this set a priori, one can avoid it. To this end, we developed a new set-based recursive procedure to compute the inevitable set. This set is not limited to robotic manipulators, it can be used in cooperation of swarm of mobile robots and swarm of unmanned aerial vehicles (UAV).

1.4 Thesis Outline

Chapter 2, presents the predictive control basics used in our approach. Basically, we briefly review the principle and the mathematical description of NMPC. Then we discuss how the controller parameters effects both the stability and performance. Additionally, we comment on problem of stability for NMPC. Afterwards, we present a brief overview of the predictive path-following problem in state- and output-space. Then, we present their solutions in the NMPC framework, as well as, the conditions of path followability are discussed for unconstrained and constrained dynamics system. Finally, we outline the effects of state-estimation and time delays on the stability of NMPC.

In Chapter 3, we give a brief overview on force feedback control. We outline active and passive compliance-based control methods, in particular, *impedance* and *admittance* force control. We comment on their advantages and limitations, as well as, the issue of the environment identification. In addition, the way of choosing the admittance parameters based-on duality principle is explained. Then, we survey briefly optimization-based force control approaches, with emphasis on model predictive control schemes.

In Chapter 4, we present the main contribution of this thesis. First, we review the existing approach for combined path-following and force-feedback control using a feedback linearization method. Then, we explain our approach, which exploits the model predictive control scheme to solve path-following and force feedback simultaneously within one optimization problem.

In Chapter 5, we verify the proposed approach in simulation and experimental validations. First, we present the model of the robot, which is adopted in simulation and experimental work. Then we perform simulation results for predictive path-following. Third, we test the applicability of the proposed approach (i.e. path-following and force feedback problem) using different desired paths, contact-forces and different degrees of environment stiffness in simulations. Finally, we conduct the experimental validation.

In Chapter 6, We solve the obstacle avoidance problem and compute the set of inevitable initial states. Additionally, we illustrate the applicability of these methods with examples. Then both problems are included in the MPC framework. In the last chapter, concluding remarks with future perspective are provided.

2 Review of Nonlinear Model Predictive Path Following Control

In this chapter the basic concept of Nonlinear Model Predictive Control (NMPC) is reviewed. Basically, we outline its key advantages to control nonlinear systems subjected to constraints on state and input, and show the differences between sampled-data open-loop NMPC − which is adopted in the next chapters− and instantaneous NMPC. This chapter does not provide a comprehensive review of NMPC; it rather provides the basis for the following chapters. For more complete reviews we refer to [7, 21, 29, 41, 42, 54, 80, 92, 96]. We first present a mathematical setup of sampled-data open-loop NMPC, and provide some remarks on the stability of NMPC. Furthermore, the predictive path-following problem is introduced. Lastly, we discuss two challenges in NMPC, namely state estimation and the handling of measurement delays.

2.1 Principle of Model Predictive Control

Model predictive control (MPC) belongs to the family of model-based control [20, 51, 79, 84, 86]. In MPC, open-loop optimal control problems are solved repeatedly minimizing an objective function while considering constraints on inputs and states. One speaks of nonlinear model predictive control if the model and/or the constraints in the optimization problem are nonlinear. The model in the NMPC controller describes the controlled plant and is used to produce a predicted trajectory starting from a measured (estimated) initial state with the optimized input. In principle, the optimization problem should be solved over infinite horizon. Solving infinite horizon optimization problem is however time consuming if possible at all. Therefore, many approaches have been proposed to use a finite instead of an infinite horizon NMPC.
Choosing a finite horizon leads to sacrificing global optima and stability. Furthermore, there will be a mismatch between the real trajectory and predicted one. In addition, in practice, there exist external disturbances and model plant mismatch leading to differences between true system state and predicted state. Thus, to overcome this deviation and to eliminate the disturbances it is required to incorporate feedback. This is achieved in NMPC by applying the calculated optimal open-loop input just until the next recalculation time. Then the complete process prediction and optimization is repeated along the same prediction horizon N. For this reason NMPC is also known as *Moving Receding Horizon Control*, (see Figure 2.1). One can summarize model

predictive control procedure in the following algorithm:

1. Obtain the current state of the dynamic system.
2. Calculate an admissible optimal-input by solving an open-loop optimal control problem over the prediction horizon using the system model and the current state for prediction.
3. Applying the obtained optimal-input until the next recalculation time.
4. Goto to 1.

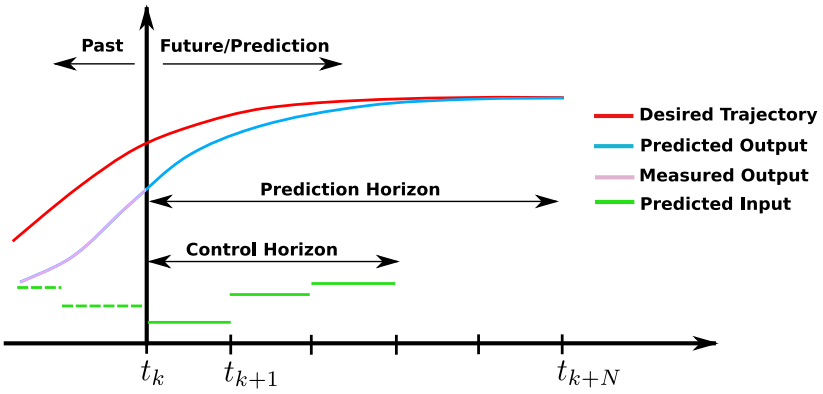


Figure 2.1: Concept of model predictive control.

2.1.1 Mathematical Setup

In this section, we provide a formal mathematical description of NMPC similar to [41, 48]. Consider a nonlinear system of the form

$$\dot{x}(t) = f(x(t), u(t)), \quad x(t_0) = x_0 \in \mathcal{X}_0, \tag{2.1a}$$
$$x \in \mathcal{X} \subseteq \mathbb{R}^{n_x}, \tag{2.1b}$$
$$u \in \mathcal{U} \subseteq \mathbb{R}^{n_u}. \tag{2.1c}$$

Where $x(t) \in \mathcal{X} \subseteq \mathbb{R}^{n_x}$ is the system state, $u(t) \in \mathcal{U} \subset \mathbb{R}^{n_u}$ is the input applied to the system. (2.1b), and (2.1c) are of constraints on state and input. The initial condition is denoted by x_0, the state trajectory starting from x_0 at time t_0 is defined by $x(\tau, u(\cdot)|x(t_0))$, where $u(\cdot) : [t_0, t_1] \to \mathcal{U}$ and $\tau \in [t_0, t_1]$. it is assumed the maps $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$ is sufficiently often continuously differentiable, satisfies $f(0, 0) = 0$, and is locally Lipschitz in x. Note that $\mathcal{X}, \mathcal{X}_0, \mathcal{U}$ are assumed such that $\mathcal{X}$ is simply connected, $\mathcal{X}_0 \subseteq \mathcal{X}$, $\mathcal{U}$ is compact, and $(0, 0) \in \mathcal{X} \times \mathcal{U}$.

To counteract disturbances feedback in NMPC, the above an open-loop optimal control problem is solved repeatedly. The mathematical formulation for this is as follows

$$\min_{\bar{u}(.)\in\mathcal{PC}(\mathcal{U})} J(\bar{x}(\cdot),\bar{u}(\cdot)) \tag{2.2a}$$

$$\text{s.t.} \quad \dot{\bar{x}}(\tau) = f(\bar{x}(\tau),\bar{u}(\tau)), \quad \bar{x}(t) = x(t) \tag{2.2b}$$

$$\bar{x}(\tau) \in \mathcal{X}, \quad \tau \in [t, t+N], \tag{2.2c}$$

$$\bar{u}(\tau) \in \mathcal{U}, \quad \tau \in [t, t+N], \tag{2.2d}$$

$$\bar{x}(t+N) \in \Omega, \tag{2.2e}$$

where the cost functional J has to be minimized over the prediction horizon N

$$J(\bar{x}(\cdot),\bar{u}(\cdot)) = \int_t^{t+N} L(\bar{x}(\tau),\bar{u}(\tau))d\tau + E(\bar{x}(t+N)). \tag{2.2f}$$

Here L, E are the stage cost function and stability terminal penalty term respectively, which can be formed to achieve the desired performance. The symbol $\bar{\cdot}$ in (2.2) refers to the predicted variables (internal controller variables). This is necessary as one needs to distinguish between the real system variables and the controller variables, because there will be a mismatch between them even in nominal case due to the finite optimization horizon. As common in NMPC the stage cost function $L : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}_0^+$ is assumed to be continuous, fulfills $L(0,0) = 0$, and is lower bounded by a class $\mathcal{K}$-function[1] α_L, i.e. $\alpha_L(x) \leq L(x,u)$. The terms in L are added depending on the desired performance has to be achieved, e.g. economical −minimize cost or maximize profit− and safety considerations. Often, a quadratic form adopted for the cost function, i.e. $L(x,u) = x^TQx + u^TRu$, with weighting matrices $Q > 0$ (positive definite) and $R \geq 0$ (semi-positive definite), (see Appendix A).
The optimal cost of (2.2) as a function of the state is called *value function*:

$$V(x(t)) = J(x(\cdot;\bar{u}^*(\cdot;x(t)) \mid x(t)), \bar{u}^*(\cdot;x(t))). \tag{2.3}$$

2.1.2 Sampled-data Realization

In sampled-data open-loop NMPC (for a sake of simplicity, we use NMPC interchangeably with sampled-data open-loop NMPC), the optimal control problem (2.2) is solved at fixed recalculation instants. The resulting optimal input is applied open-loop to the system. We refer to the recalculation instants by t_k. The time in-between recalculation instants can vary for practical reasons, i.e. the state information is often determined externally and might vary. The recalculation instants t_k are defined by a partition π of the time axis [41].

[1] See Appendix A for the definition of $\mathcal{K}$-function

Definition 2.1 (Partition)
Consider that $\pi = (t_k)$ is a series, with $k \in \mathbb{N}$ of (finite) positive real numbers such that $t_0 = 0$, $t_k < t_{k+1}$ and $t_k \to \infty$ for $k \to \infty$. Furthermore, $\bar{\pi} = sup_{k\in\mathbb{N}}(t_{k+1} - t_k)$ denotes the longest recalculation time of π and $\underline{\pi} = inf_{k\in\mathbb{N}}(t_{k+1} - t_k)$ denotes the shortest recalculation time of π.

In the following, we define the *sampling time* as follows:

Definition 2.2 (Sampling Time)
The sampling time $t_k \in \pi$ is given by

$$\delta_k = t_{k+1} - t_k. \tag{2.4}$$

Such that, state measurements are calculated at times $t_k = t_0 + k\delta$ where t_0 is the starting time and $k = 0, 1, 2, \ldots$.

The solution of (2.2) initialized at $x(t_k)$ is denoted by $\bar{u}^*(\cdot; x(t_k)) : [t_k, t_k + N] \to \mathbb{R}^{n_u}$. The applied (sub-)optimal open-loop input until the next sampling instant t_{k+1} is defined as:

$$u(t; x(t_k)) = \bar{u}^*(t; x(t_k)), \quad t \in [t_k,\, t_{k+1})\,. \tag{2.5}$$

If the open-loop optimal control problem is solved at all time instants is *instantaneous* NMPC. While, it is called *sampled-data open-loop* NMPC, if the open-loop optimal control problem is solved only at discrete feedback time instants, then the resulting optimal control signal is applied open-loop in between. The applied input and nominal closed-loop for both schemes shown in Table 2.1.

Table 2.1: Comparison between Instantaneous and Sampled-data open-loop NMPC

NMPC Scheme	The applied input	Nominal closed-loop
Instantaneous NMPC	$u(t) = \bar{u}^*(\tau; x(t))$	$\dot{x} = f(x(t), \bar{u}^*(t; x(t)))$
Sampled-data open-loop NMPC	$u(t) = \bar{u}^*(\tau; x(t_k))$	$\dot{x} = f(x(t), \bar{u}^*(t; x(t_k)))$

Optimization Parameters

Although the optimal control problem (2.2) is solved on-line at any recalculation instant $t_k \in \pi$, the problem is parametrized by a set of offline defined parameters. They have an effect on stability and performance of the closed-loop system. These parameters are:

- the cost functional $J(\cdot)$ parametrized by
  - the integral cost function $L(x(t), u(t)) : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}_0^+$.

  - the terminal cost $E(x(\cdot)) : \mathbb{R}^{n_x} \to \mathbb{R}_0^+$.

- the terminal region $\Omega \subseteq \mathcal{X}$.
- the prediction horizon $N \in \mathbb{R}^+$.
- the sampling time δ.

Efficient tuning of these parameters influences the performance and constraints satisfaction. To do so, compromise between performance and computation burden are needed. Intuitively the prediction horizon should be as long as possible. This renders solving the optimal control problem in real-time very difficult or infeasible. Furthermore, by properly choosing the weight matrices in the cost function $L(\cdot)$ the control performance can be improved. As common, the stability of the optimal control problem is ensured by adding the terminal cost $E(\cdot)$ and terminal region Ω as explained in the next section. Regarding the sampling time, typically the best choice is as long as possible while achieving all performance demands. In fact, the controller computation-time bounds the minimum value of the sampling time. However, a faster sampling rate is best for disturbance rejection, but it increases the computation-time, we will discuss this issue in detail in Section 2.3.

2.1.3 Stability

We consider the nominal optimal control problem, i.e., there is no model-plant mismatch and no disturbance interfering the controlled system. For this setup, we classify the stability approaches of a sampled-data open-loop NMPC scheme into two categories: (i) Stability of the infinite horizon NMPC, and (ii) Stability of the finite horizon NMPC [47, 61, 84].

Stability of The Infinite Horizon NMPC:

By considering an infinite prediction horizon (i.e. $N = \infty$), obtains in the nominal case the whole information about the system evolution. According to the principle of optimality [17], the calculated and measured trajectories coincide, since there is no model-plant mismatch. The main disadvantage of this approach, it increases computation-time, which is clearly undesirable and is problematic with disturbances and model plant mismatch.

For an infinite horizon NMPC, the cost function of the optimal control problem (2.2) is given by

$$J(\bar{x}(\cdot), \bar{u}(\cdot)) = \int_t^{\infty} L(\bar{x}(\tau), \bar{u}(\tau))\, d\tau, \quad \text{instantaneous NMPC} \tag{2.6a}$$

$$J(\bar{x}(\cdot), \bar{u}(\cdot)) = \int_{t_0}^{t_\infty} L(\bar{x}(\tau), \bar{u}(\tau))\, d\tau, \quad \text{sampled-data open-loop NMPC.} \tag{2.6b}$$

The basic idea of an infinite horizon NMPC stability is to use the value function as a Laypunov-function [83, 84].

Stability of Finite Horizon NMPC Schemes:
For a finite horizon the following approaches are often used to achieve stability of a NMPC controller.

1. **Zero-terminal constraint**:
   In this approach, the stability is achieved by forcing the system states to be zero at the end of the prediction horizon [41, 47, 68, 83], i.e.

   $$x(t_k + N) = 0, \quad \forall t_k \in \pi, \tag{2.7}$$

   during any step of solving the optimal control problem. This approach sets a strong constraint (equality constraint), which leads to numerical difficulties. This could easily reducing the feasible solutions, (i.e. the solution of the optimal control problem satisfying all constraints).

2. **Dual-mode control**:
   Assuming there is a local -linear- controller stabilizing the system inside a set around the origin. The stability in this approach is achieved by forcing the system state to end in a terminal region for which the local stability exists. Once the system state trajectory enters the terminal region, the local linear controller is used to keep the system state inside the terminal region. Convergence to the set is achieved by adding a terminal penalty E, which ensures the decreasing of the value function [85, 105].

3. **Lyapunov-function based NMPC approach**:
   In this approach a terminal cost or terminal region (terminal inequality constraint) are used as a Lyapunov function [64, 116]. If the terminal cost is a global Lyapunov function, then the terminal region constraint is not necessary. For a local Lyapunov control function, the global convergence to the origin can be achieved by increasing the horizon. However, it is not easy to find a control Lyapunov function considering constraints on the states and inputs.

4. **Direct contraction condition**:
   To achieve stability the so-called contraction constraint is added to the open-loop optimal control problem [27, 93]. This constraint explicitly enforces the states to contract at the recalculation instants by adding a constraint of the form

   $$\|\bar{x}(t_{k+1})\| \leq \beta \|x(t_k)\|, \quad \beta \in (0, 1).$$

The main disadvantage of this approach is that recursive-feasibility is not guaranteed, so other assumptions on the system need to be added.

5. **Quasi-Infinite Horizon**:
   In this method the closed-loop stability is guaranteed to both stable and unstable systems subject to input constraints [22]. Basically, the cost functional has two parts, first part is a finite horizon integral square error, and a quadratic terminal cost represents a second part. Additionally, the penalty matrix of the terminal cost is found by solving an appropriate Lyapunov equation. A terminal inequality constraint ensures that the terminal state lies inside a specified terminal region constraint.

Typically the terminal region Ω and terminal penalty E are determine offline such that the cost functional

$$J(\bar{x}(\cdot), \bar{u}(\cdot)) = \int_{t_0}^{t_0+N} L(\bar{x}(\tau), \bar{u}(\tau)) + E(\bar{x}(t+N)), \tag{2.8}$$

sets an upper bound on the infinite horizon cost and therefore ensures the value function decreases as the horizon recedes in time. Theorem A.1 in Appendix A gives sufficient conditions for convergence of the closed-loop state towards the origin.

2.1.4 Control Problems Handled By Nonlinear Model Predictive Controller

NMPC can handle many control problems. We consider three types: set-point stabilization, trajectory tracking, and path-following. We do not comment on so called economic MPC approaches. Next, we give a mathematical description of these problems.

- **Set point stabilization:**
  To define this control problem mathematically, consider we have a dynamic system (2.1) with a reference state

  $$x = x_r, \tag{2.9}$$

  where the state $x_r \in \mathbb{R}^{n_x}$ is a desired reference. In the point stabilization problem, the controller objective is to drive the state of the dynamic system (2.1) from initial state x_0 to the set-point x_r using the feedback controller $\mu_c(x) = x \mapsto u$, such that

  $$\lim_{t\to\infty} \|x(t) - x_r\| = 0. \tag{2.10}$$

  The set-point stabilization problem is well known in control theory, for instance, feedback and feed-forward controllers and appears in a wide range of systems e.g.

(continuous and discrete, linear and nonlinear); and applications e.g. temperature control and many others.

- **Trajectory tracking:**
  In the case that the reference state trajectory is a function of time, i.e. $x_r(t) : [t_0, \infty) \mapsto \mathbb{R}^n$, the control objective is to track online the reference evolution, such that

$$\lim_{t\to\infty} \|x(t) - x_r(t)\| = 0. \tag{2.11}$$

  The velocity of the reference, i.e. $\dot{x}_r(t)$ is determined offline to when to be where on the reference trajectory. According to this setup, this control problem is called a *trajectory-tracking problem.*

- **Path-following:**
  If we consider the reference state trajectory as a function of time, i.e. $x_r(t) : [t_0, \infty) \mapsto \mathbb{R}^n$, but the time evolution of the reference is not preassigned. Then, the reference can be defined as a geometric reference path without given the timing along it a priori. Assuming the geometric path is defined as follows

$$\mathcal{P} = \{x \in \mathbb{R}^{n_x} \mid \theta(t) \in \Theta \mapsto \ x = p(\theta)\}, \tag{2.12}$$

  where $p : \mathbb{R} \to \mathbb{R}^{n_x}$ is a parametrization of $\mathcal{P}$ (2.12), $\theta(t) \in \Theta \subseteq \mathbb{R}$ is called the path-parameter, which is time dependent, but its time evolution is not specified a priori, and Θ is a compact set. In this control problem, the controller objective is to move the system state along the path, such that

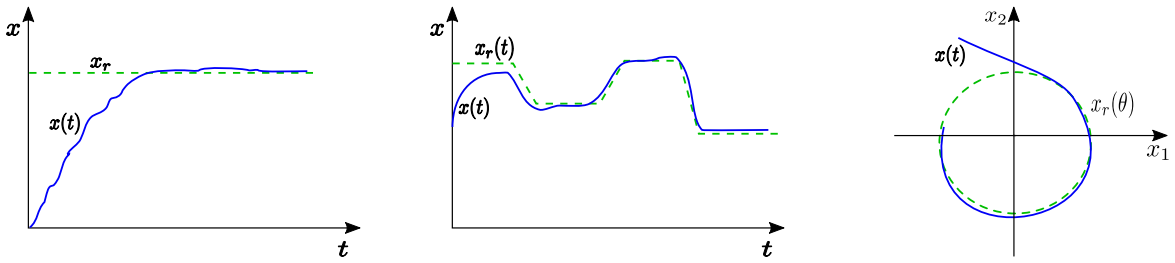
$$\lim_{t\to\infty} \|x(t) - p(\theta(t))\| = 0, \tag{2.13a}$$

$$\dot{\theta}(t) \geq 0. \tag{2.13b}$$

  According to (2.13a), the objective is to move the system state toward the current state of the parametrized-path. While, the constraint on the path-parameter evolution (2.13b) ensures the forward motion along the path.

With a little investigation, one can see that the set-point stabilization and trajectory tracking problems are special cases of the path-following problem. Setting the parametrization variable to be a constant, i.e. $\theta(t) = c$ for all t, the path-following problem turns into the set-point stabilization problem. While, by defining the time evolution of the parametrization variable $\theta(t)$ a priori, the path-following problem turns into the tracking problem. As one can see the freedom to design a timing of the parametrization variable is an advantage of path-following. To illustrate these three control problems, consider that the objective of an unmanned Helicopter is to fly to a

specific point and stay hovering there, this objective represents a set-point stabilization problem. However, if the objective of the unmanned Helicopter is to fly along a specified path and reaching points on the path with preassigned time, then the objective is a trajectory tracking. Whereas, if the objective of the unmanned Helicopter is to fly as precise as possible along a path, this objective is a path-following problem (see Figure 2.2). In the next section, we describe predictive path-following controllers in state-space and output-space.



(a) Set-point tracking (b) Trajectory tracking (c) Path-following

Figure 2.2: Types of control problems.

2.2 Nonlinear Model Predictive Path-following Controller

Here, we give a formal description of the path-following control adopted in our proposed approach. Namely, we describe a tailored NMPC formulation for solving the path-following problem as introduced in [34]. Among other controllers used to tackle the path-following problem [5, 55, 56, 89], predictive path-following [34–37, 127] has the advantage, that it can consider constraints on states and inputs. First, we describe a predictive path-following problem when the reference path is defined in the state-space. Second, we expand the path-following problem to a generic and more applicable problem, when the reference path is defined in the output-space.

2.2.1 Predictive Path-following in the State-space

To illustrate the predictive path-following controller, we assume that both the system states and the followed path live in the same space, i.e. state-space. This assumption makes the problem simpler, where we can compare the system states with the parametrized path coordinates without mapping each of them to a different space, e.g. output-space. Here, we consider the dynamic system (2.1). The followed path is defined for simplicity in the state space such that the final value of the path-parameter is 0,

$$\mathcal{P} = \{x \in \mathbb{R}^{n_x} \mid \theta \in [\theta_{min},\, 0] \longmapsto \; x = p(\theta)\}\,, \tag{2.14}$$

where $\theta_{min} \in (-\infty, 0]$, and $p(0) = 0$; these assumptions are needed to ensure that both the system state and the parametrized path end up at the origin when $t \to \infty$.

Furthermore, the parametrization p is assumed to be sufficiently often continuously differentiable. The path parameter $\theta(t)$, which is time dependent, but its time evolution is not specified a priori. Furthermore, the path evolution is driven by a virtual input $\hat{u}(t)$ such that

$$\dot{\theta}(t) = g(\theta, \hat{u}), \quad \hat{u} \in \hat{\mathcal{U}} \subseteq \mathbb{R}, \tag{2.15}$$

the last virtual dynamic system is called a timing law, which gives an additional degree of freedom to the controller. Here, the timing law is designed for simplicity as a first order dynamic system.

Assumption 2.1 *The reference path $\mathcal{P}$ is within the feasible set of the system states, i.e. $\mathcal{P} \subseteq \mathcal{X}$.*

The Assumption 2.1 is necessary to ensure that there exists at least one $x \in \mathcal{X}$ coinciding every point on the reference path $\mathcal{P}$.

Problem 2.1 (Predictive Path-following Problem)
Given system (2.1)*, and an a priori known geometric path $\mathcal{P}$* (2.14)*, design a controller that drives the system state to fulfill:*

- ***Convergence to path**: The system state $x(t)$ moves toward the path $\mathcal{P}$ such that, $\lim_{t\to\infty} \|x(t) - p(\theta(t))\| = 0$.*
- ***Convergence on path**: The system state keeps going along the path monotonically in increasing direction of $\theta(t)$ s.t. $\dot{\theta}(t) > 0$ holds for almost all $\theta(t) \in [\theta_{min}, 0)$ and $\lim_{t\to\infty} \theta(t) = 0$.*
- ***Feasibility**: The constraints on the states $x(t) \in \mathcal{X}$ and on the inputs $u(t) \in \mathcal{U}$ are satisfied for all time $t \geq t_0$.*

■

The geometric interpretation of the two convergence objectives of the Problem 2.1 is shown in Figure 2.3. To solve Problem 2.1, we use the approach proposed in [34]. There, a Nonlinear Model Predictive Control (NMPC) scheme is introduced to solve the constrained path following problem. Basically, the optimal control problem (OCP) is solved at each instant $t_k = t_0 + k\delta$, with $k \in \mathbb{N}_0$, and a fixed sampling period $\delta > 0$, the cost functional is defined as follows

$$\begin{aligned} J(x(t_k), \bar{e}_x(\cdot), \bar{u}(\cdot), \bar{\theta}(\cdot), \bar{\hat{u}}(\cdot)) \quad &= \\ \int_{t_k}^{t_k+N} L(\bar{e}_x(\tau), \bar{u}(\tau), \bar{\theta}(\tau), \bar{\hat{u}}(\tau))d\tau \quad &+ \quad E(\bar{x}(t_k+N), \bar{\theta}(t_k+N)). \end{aligned} \tag{2.16}$$

As common in NMPC the function $L : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}_0^+$ is assumed to be continuous. The cost function is assumed to

$$\lim_{t\to\infty} L(\cdot) = 0 \to \|e_x\| = 0 \ \wedge \ \theta \to 0,$$

$L(\cdot)$ is lower bounded by a class $\mathcal{K}$ function $\alpha_L(\left\|(e_x, \theta)^T\right\|)$, $N \in (\delta, \infty)$ represents the prediction horizon, and $E : \mathcal{X} \times [\theta_{min}, 0] \to \mathbb{R}_0^+$ is the terminal penalty.

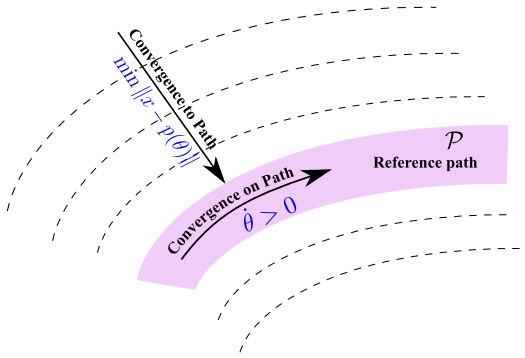


Figure 2.3: Geometric interpretation of Problem 2.1

The open-loop optimal input applied in-between the sampling instants is given by the solution of the following optimal control problem OCP:

$$\min_{(\bar{u}(.),\bar{\hat{u}}(.))\in(U\times\hat{U})} \quad J(x(t_k), \bar{e}_x(\cdot), \bar{u}(\cdot), \bar{\theta}(\cdot), \bar{\hat{u}}(\cdot)) \tag{2.17a}$$

$$\text{s.t.} \quad \dot{\bar{x}}(\tau) = f(\bar{x}(\tau), \bar{u}(\tau)), \quad \bar{x}(t_k) = x(t_k), \tag{2.17b}$$

$$\dot{\bar{\theta}}(\tau) = g(\bar{\theta}(\tau), \bar{\hat{u}}(\tau)), \quad \bar{\theta}(t_k) = \theta(t_k), \tag{2.17c}$$

$$\bar{e}_x(\tau) = \bar{x}(\tau) - p(\bar{\theta}(\tau)), \tag{2.17d}$$

$$\bar{x}(t_k + N) \in \Omega \subseteq \mathcal{X} \subseteq \mathbb{R}^n, \tag{2.17e}$$

$$\bar{x}(\tau) \in \mathcal{X}, \ \bar{u}(\tau) \in \mathcal{U}, \tag{2.17f}$$

$$\bar{\theta}(\tau) \in \hat{\mathcal{X}}, \ \bar{\hat{u}}(\tau) \in \hat{\mathcal{U}}, \tag{2.17g}$$

where $\bar{\cdot}$ symbol denotes predicted variables (internal variables of the controller), the solution of (2.17b) is denoted by $\bar{x}(t, \bar{u}(\cdot)|x(t_k))$, which is controlled by the input $\bar{u}(\cdot) : [t_k, t_k + N] \to \mathcal{U}$ with initial condition $x(t_k)$. The prediction process needs the system model to simulate the future of the system, therefore, the dynamic system (2.1) is used as dynamic constraints (2.17b)- (2.17c). The terminal constraint (2.17e) forces the predicted state $\bar{x}(t_k + N)$ to be in the terminal region Ω at the end of each prediction. The virtual state $\bar{\theta}$ is constrained by (2.17g) where the set $\hat{\mathcal{X}}$ is defined as

$$\hat{\mathcal{X}} := [\theta_{min}, 0]. \tag{2.18}$$

This constraint keeps the path-parameter $\bar{\theta} \in [\theta_{min}, 0]$. To keep the solution of (2.17c) within moderate values, the virtual input $\bar{\hat{u}}$ is constrained by $\hat{\mathcal{U}} = [\hat{u}_{min}, \hat{u}_{max}] \subset \mathbb{R}$ where, $\hat{u}_{min} \leq 0 \leq \hat{u}_{max}$. Solving the differential equation (2.17c) requires an initial condition $\theta(t_k)$ at every sampling instant. If the initial point on the reference path is specified $p(\theta(t_0))$, then the corresponding path-parameter is chosen as an initial condition $\bar{\theta}(t_0)$ at the initial sampling instant. If no initial point is specified a priori, it is possible to choose the path-parameter as the point corresponding to the point on the path, which is closest to the initial state of the system $x(t_k)$ by solving the following minimum distance problem [34]

$$\bar{\theta}(t_k) = \underset{\theta \in [\theta_{min}, 0]}{\operatorname{argmin}} \left\| x(t_k) - p(\theta) \right\|. \tag{2.19}$$

This minimization problem might have more than one optimal solution. In this case we can pick one of them to serve as an initial condition of the timing-law equation. For each next sampling instant, i.e. $k > 0$ the new measured or estimated state $x(t_k)$ acts as initial condition for (2.17b). However, the initial condition of timing law (2.17c) is the last predicted value at time t_k, i.e. $\theta(t_k) = \bar{\theta}(t_k, \bar{\hat{u}}_{k-1}(\cdot) \mid \bar{\theta}(t_{k-1}))$. The solution of the OCP (2.17) is denoted by $\bar{u}^*(\cdot; x(t_k))$ and $\bar{\hat{u}}^*(\cdot; \theta(t_k))$. Where, $\bar{u}^*(\cdot; x(t_k))$ represents the (sub-)optimal open-loop input that is applied to the system until the next sampling instant t_{k+1}:

$$u(t; x(t_k)) = \bar{u}^*(t; x(t_k)), \quad t \in [t_k, t_{k+1}). \tag{2.20}$$

While $\bar{\hat{u}}^*(\cdot; \theta(t_k))$ is used to generate the new initial condition of the virtual system as well as to control the path dynamics, i.e. $t \to p(\theta(t)) \in \mathcal{P}$.

Assumption 2.2 *The first solution of* (2.17) *exists, i.e. there are input signals* $(u(\cdot), \hat{u}(\cdot))$ *such that all constraints are fulfilled.*

The motivation behind Assumption 2.2 is to achieve −with other conditions− the recursive feasibility of NMPC, similar to the one outlined in Section 2.1.3. In the next section, we discuss the predictive following problem in output-space.

2.2.2 Predictive Path-following in The Output-space

So far we described the predictive path-following problem in state-space. Here, we extend predictive path following problem to more application-relevant setup when the reference path is defined in an output-space, which is in general a sub-space of the state space. To do so, it is needed to map the states to the output-space, as well as, the path smoothness is govern by the relative degree of the dynamic system as will be explained in the following. First we extend the dynamic system (2.1) by adding the

output function as follows

$$\dot{x}(t) = f(x(t), u(t)), \quad x(t_0) = x_0, \tag{2.21a}$$
$$y(t) = h(x(t)), \qquad y \in \mathbb{R}^{n_y} \tag{2.21b}$$
$$x \in \mathcal{X} \subset \mathbb{R}^{n_x}, \tag{2.21c}$$
$$u \in \mathcal{U} \subset \mathbb{R}^{n_u}, \tag{2.21d}$$

where $x(t), u(t), x_0, f(\cdot,\cdot), \mathcal{X}, \mathcal{U}$ are defined as before. $y(t) \in \mathbb{R}^{n_y}$ is the output of the system and its trajectory defined by $y(\tau|x(t_0)) = h(x(\tau, u(\cdot)|x(t_0)))$. For sake of simplicity, we assume that the system is square, i.e. $n_u = n_y$.

we furthermore consider that, we have a parametrized geometric curve defined in output-space

$$\mathcal{P} = \{y \in \mathbb{R}^{n_y}\,; \theta \in [\theta_{min}, \theta_{max}] \mid y = p(\theta)\}\,, \tag{2.22}$$

where $p(\theta)$ is a parametrization of $\mathcal{P}$ in the output-space. While the parameter $\theta(t)$ is a function of time, the time evolution $t \to \theta(t)$ is not known a priori. Furthermore, the system input $u : [t_0, \infty) \to \mathcal{U}$ and the time evolution of $\theta(t)$ have to be chosen such that the system output follows the specified path as precise as possible.
In the following, we consider the problem of driving the system output (2.21b) to the reference path (2.22), as well as keeping it moving along the path in the increasing direction.

Problem 2.2 (Predictive Output Path-following Problem)
Given system (2.21)*, and an a priori known geometric path* $\mathcal{P}$ (2.22)*, design a controller that drives the system output* (2.21b) *to fulfill:*

- ***Convergence to path****: The system output* $y = h(x)$ *moves toward the path* $\mathcal{P}$ *such that*

$$\lim_{t\to\infty} \|h(x(t)) - p(\theta(t))\| = 0.$$

- ***Convergence on path****: The system output keeps going along the path monotonically in the increasing direction of* $\theta(t)$*, i.e.* $\dot{\theta}(t) \geq 0$ *holds and* $\lim_{t\to\infty} \theta(t) = \theta_{max}$*.*

- ***Feasibility****: The constraints on states* $x(t) \in \mathcal{X}$ *and on inputs* $u(t) \in \mathcal{U}$ *are satisfied for all time* $t \geq t_0$*.*

■

We choose as the timing law to move along $\mathcal{P}$ a simple integrator chain

$$\theta^{(d+1)}(t) = \hat{u}(t), \tag{2.23}$$

where d is sufficiently large as detailed later. It is supposed that, the timing law input $\hat{u}(t)$ is piece-wise continuous and bounded, i.e. $\hat{u}(\cdot) \in \hat{\mathcal{U}} \subset \mathbb{R}$. Defining

$$\hat{x}(t) := \left[\theta, \dot{\theta}, \cdots, \theta^{(d)}\right]^T$$

we augment (2.21), by the timing law equation as follows

$$\begin{bmatrix} \dot{x} \\ \dot{\hat{x}} \end{bmatrix} = \begin{bmatrix} f(x,u) \\ g(\hat{x},\hat{u}) \end{bmatrix}, \quad \begin{bmatrix} x(t_0) \\ \hat{x}(t_0) \end{bmatrix} = \begin{bmatrix} x_0 \\ \hat{x}_0 \end{bmatrix}, \tag{2.24a}$$

$$\begin{bmatrix} e_y \\ \theta \end{bmatrix} = \begin{bmatrix} h(x) - p(\theta) \\ \hat{x}_1 \end{bmatrix}, \tag{2.24b}$$

where the state space form $\dot{\hat{x}} = g(\hat{x}, \hat{u})$ is added to the system dynamics (2.21a). The output of the augmented system (2.24) consists of two parts. The first part represents the path following error in output-space $e_y = h(x) - p(\theta)$, while the second part stands for the first state in the timing law, i.e. $\theta = \hat{x}_1$. According to this new formulation, the path following controller task is to achieve the convergence of both e_y to 0 and θ to θ_{max}.

To solve Problem 2.2, likewise the path following problem in state-space, i.e. 2.1, we use NMPC. The optimal control problem is solved at each instant $t_k = t_0 + k\delta$, with $k \in \mathbb{N}_0$, and sampling period $\delta > 0$, while the cost functional is defined as follows

$$J(x(t_k), \bar{e}_y(\cdot), \bar{u}(\cdot), \bar{\theta}(\cdot), \bar{\hat{u}}(\cdot)) = \int_{t_k}^{t_k+N} L(\bar{e}_y(\tau), \bar{u}(\tau), \bar{\theta}(\tau), \bar{\hat{u}}(\tau)) d\tau. \tag{2.25}$$

As before the cost function $L : \mathbb{R}^{n_y} \times \mathbb{R} \times \mathcal{U} \times \hat{\mathcal{U}} \to \mathbb{R}_0^+$ is assumed to be continuous, and to fulfill

$$\lim_{t\to\infty} L(\cdot) = 0 \xrightarrow{\lim_{t\to\infty}} \|e_y\| = 0 \ \wedge \ \theta \to \theta_{max}$$

$L(\cdot)$ is lower bounded by a class $\mathcal{K}$ function $\alpha_L\left(\left\|(e_y, \theta)^T\right\|\right)$, and $N \in (\delta, \infty)$ represents the prediction horizon. The open-loop optimal input applied in-between the sampling

instants is the solution of the following OCP [34]:

$$\min_{(\bar{u}(.),\bar{\hat{u}}(.))\in(U\times\hat{U})} J(x(t_k),\bar{e}_y(\cdot),\bar{u}(\cdot),\bar{\theta}(\cdot),\bar{\hat{u}}(\cdot)), \tag{2.26a}$$

$$\text{s.t.} \quad \dot{\bar{x}}(\tau) = f(\bar{x}(\tau),\bar{u}(\tau)), \quad \bar{x}(t_k) = x(t_k), \tag{2.26b}$$

$$\dot{\bar{\hat{x}}}(\tau) = g(\bar{\hat{x}}(\tau),\bar{\hat{u}}(\tau)), \quad \bar{\hat{x}}(t_k) = \hat{x}(t_k), \tag{2.26c}$$

$$\bar{e}_y(\tau) = h(\bar{x}_p(\tau)) - p(\bar{\hat{x}}_1(\tau)), \tag{2.26d}$$

$$\bar{x}(t_k+N) \in \Omega \subseteq \mathcal{X} \subseteq \mathbb{R}^{n_x}, \tag{2.26e}$$

$$\bar{\theta} = \bar{\hat{x}}_1(\tau), \tag{2.26f}$$

$$\bar{x}(\tau) \in \mathcal{X},\ \bar{u}(\tau) \in \mathcal{U}, \tag{2.26g}$$

$$\bar{\hat{x}}(\tau) \in \hat{\mathcal{X}},\ \bar{\hat{u}}(\tau) \in \hat{\mathcal{U}}. \tag{2.26h}$$

The prediction process needs the system model to simulate the future of the system, therefore, the augmented system (2.26) is used as dynamic constraints (2.26b)−(2.26f). The terminal constraint (2.26e) forces the predicted state $\bar{x}(t_k+N)$ to be in the terminal region Ω at the end of each prediction. Here, the solution of (2.26b) is denoted by $\bar{x}(t,\bar{u}(\cdot)|x(t_k))$, which is controlled by the input $\bar{u}(\cdot) : [t_k, t_k+N] \to \mathcal{U}$ with initial condition $x(t_k)$. The virtual states $\hat{x}$ are constrained by (2.26h) where the set $\hat{\mathcal{X}}$ is defined as

$$\hat{\mathcal{X}} := [\theta_{min},\theta_{max}] \times [0,\infty] \times \mathbb{R}^{d-1}. \tag{2.27}$$

This constraint keeps $\bar{\theta} = \bar{\hat{x}}_1 \in [\theta_0, 0]$, and $\dot{\bar{\theta}} \geq 0$. To keep the solution of (2.26c) within moderate values, the virtual input $\bar{\hat{u}}$ is constrained by $\hat{\mathcal{U}} = [\hat{u}_{min},\hat{u}_{max}] \subset \mathbb{R}$ where, $\hat{u}_{min} \leq 0 \leq \hat{u}_{max}$.

As in (2.17), the initial condition of differential equation (2.26c), i.e. $\hat{x}(t_k)$ needs to be provide at every sampling instant. Therefore, if the initial point on the reference path, i.e. $p(\theta(t_0))$ is given a priori, then the corresponding path-parameter is chosen as an initial condition at the initial sampling instant. If there is no initial point on the path specified, one can choose $\bar{\hat{x}}(t_0)$ as the point corresponding to the point on the path, which is closest to the initial state of the system $x(t_k)$ by solving similar to before the following minimum distance problem [34]

$$\bar{\hat{x}}(t_0) = \underset{\hat{x}\in[\theta_{min},\theta_{max}]}{\operatorname{argmin}} \left\| h(x(t_0)) - p(\theta) \right\|. \tag{2.28}$$

This minimization problem might have more than one optimal solution. In this case we can pick one of them to serve as an initial condition of the timing-law equation. For each next sampling instant, i.e. $k > 0$ the new measured or estimated state $x(t_k)$ acts as initial condition for (2.26b). However, the initial condition of timing law (2.26c) is the last predicted value at time t_k, i.e. $\hat{x}(t_k) = \bar{\hat{x}}(t_k, \bar{\hat{u}}_{k-1}(\cdot) \mid \bar{\hat{x}}(t_{k-1}))$.

2.2.3 Exact Followabililty Conditions

Since we use indirect force control in our approach, the accuracy of the force control depends on the position controller, i.e. the predictive path-following controller. Here, we discuss a sufficient condition that ensures that the dynamic system follows the reference path exactly and hence the desired force is applied. First, we define the *followability condition*, then we check whether a given dynamic system follows the reference path, i.e. satisfaction of the followability condition.

Definition 2.3 (Path followability conditions) [34]
The path $\mathcal{P}$ is exactly followable by the system (2.2)*, if there exist admissible inputs $u \in \mathcal{U}$ and $\hat{u} \in \hat{\mathcal{U}}$, such that the continuous trajectories $x(t, t_0, u(\cdot) \mid x_0) \in \mathcal{X}$ and $\theta(t, t_0, \hat{u}(\cdot) \mid \theta_0) \in \Theta$ and their time derivatives lead the path-following error*

$$e = x(t, t_0, u(\cdot) \mid x_0) - p(\theta(t, t_0, \hat{u}(\cdot) \mid \theta_0)),$$

and its time derivative

$$\dot{e}(t) = f(x(t, t_0, u(\cdot) \mid x_0), u(\cdot)) - \frac{\partial p(\theta(t, t_0, \hat{u}(\cdot) \mid \theta_0))}{\partial \theta} \, \dot{\theta}(t, t_0, \hat{u}(\cdot) \mid \theta_0)$$

to zero for all $t \geq t_0$.

The question is, how can one check whether the given dynamic system follows the reference path exactly or not?
The answer of this question depends on the type of the dynamic system and the shape of the reference path. To explain that, firstly, we add the following assumptions and definitions.

Assumption 2.3 (Regular path)
The parametrization $p(\theta(t)) \in \mathbb{R}^{n_y}$ in 2.22 is an embedded submanifold[2], which means that the reference path $\mathcal{P}$ has no self-intersections [56, 89].

Assumption 2.4 (Smooth and square-structure system)
$f(x, u)$ and $h(x)$ are smooth vector fields, as well as the system (2.21) *has a square input-output structure, i.e. $n_u = n_y$.*

Definition 2.4 (Vector relative degree)
A multivariable nonlinear system 2.21 has a vector relative degree $\{r_1, \ldots, r_m\}$ at point x_0 if
(i)

$$\frac{\partial}{\partial u_j} L_f^k h_i(x) = 0$$

[2]See Appendix A for the definition of embedded submanifold

for all $1 \leq j \leq m$*, for all* $k < r_i - 1$*, for all* $1 \leq i \leq m$*, and for all* x *in a neighborhood of* x_0*,*
(ii) the $m \times m$ *matrix*

$$A(x) = \begin{pmatrix} \frac{\partial}{\partial u_1} L_f^{r_1-1} h_1(x) & \dots & \frac{\partial}{\partial u_m} L_f^{r_1-1} h_1(x) \\ \frac{\partial}{\partial u_1} L_f^{r_2-1} h_2(x) & \dots & \frac{\partial}{\partial u_m} L_f^{r_2-1} h_2(x) \\ \dots & \dots & \dots \\ \frac{\partial}{\partial u_1} L_f^{r_m-1} h_m(x) & \dots & \frac{\partial}{\partial u_m} L_f^{r_m-1} h_m(x) \end{pmatrix}$$

is nonsingular at $x = x_0$*.*

Where $L_f h$ is a Lie derivative, which is the directional derivative of h along f. Here we consider a nonlinear system of form $\dot{x} = f(x, u)$ as in [34, 90] instead of an input affine MIMO system $\dot{x} = f(x) + g(x)u$ as in the standard definition [62]. Actually, the relative degree for SISO systems is exactly equal to the number of times one has to differentiate the output at $t = t_0$ in order to have the input explicitly appearing [62].

Assumption 2.5 (Well-defined vector relative degree)
The dynamic system (2.21) *has a vector relative degree* $r = (r_1, r_2, \dots, r_{n_y})$*, such that*

$$\hat{r} = max\left\{r_1, r_2, \dots, r_{n_y}\right\} \quad \rho = \sum_{i=1}^{n_y} r_i \leq n. \tag{2.29}$$

If the system (2.21) has a well-defined vector relative degree, it is possible to find a map $\Phi : x \mapsto z$, which qualifies as a local coordinate transformation in a neighborhood of x_0. If $\rho < n$, it is always possible to find $n - \rho$ more functions $\phi_{\hat{r}+1}(x), \dots, \phi_n(x)$ such that the coordinates transformation matrix Φ has a nonsingular Jacobian matrix at x_0 [62]. If the relative degree of the system (2.21) is equal to the system's order, i.e. $\rho = n$, then the system (2.21) does not have an internal dynamics. Thus, any nonlinear system that has a well-defined (vector) relative degree at some point x_0 can be transformed into a normal form system, which is linear and controllable [62] in a neighborhood of the point $z_0 = \Phi(x_0)$.
As we mentioned that the path-parameter $\theta(t)$ evolution is governed by an ODE system driven by the virtual input $\hat{u}$. Since the path-parameter appears in the output of the augmented dynamic system (2.24), so to achieve the coordinate transformation, it is needed that the output function to be smooth at least $\mathcal{C}^{\hat{r}}$. So we consider that the following assumption holds.

Assumption 2.6 (Path parametrization smoothness)
i) The timing-law is a sufficiently long integrator chain

$$\theta^{\hat{r}+1} = \hat{u} \quad \theta(t_0) = \theta_0 \quad \forall i \in 1, \dots, \hat{r} : \quad \theta^{(i)}(t_0) = 0, \tag{2.30}$$

(ii) The path parametrization (2.22) *is* $p(\theta) \in \mathcal{C}^{\hat{r}}$.
Where $\hat{r}$ *from* (2.29).

According to the Assumption 2.6 it is possible to write the state space representation of the timing law as follows:

$$\dot{\hat{x}} = A\,\hat{x} + B\,\hat{u} := g(\hat{x}, \hat{u}), \quad \hat{x}(t_0) = (\theta_0, 0, \ldots, 0) \in \mathbb{R}^{\hat{r}+1} \tag{2.31a}$$

$$\theta = C\hat{x} = \hat{x}_1. \tag{2.31b}$$

Where

$$C = (1, 0, \ldots, 0) \in \mathbb{R}^{1\times n}, \text{ and}$$

$$A := \left(\begin{array}{c|c} \mathbf{0}^{\hat{r}\times 1} & \mathbf{I}^{\hat{r}\times\hat{r}} \\ \hline 0 & \mathbf{0}^{1\times\hat{r}} \end{array}\right) \in \mathbb{R}^{(\hat{r}+1)\times(\hat{r}+1)}, \quad B := (0, \ldots, 0, 1) \in \mathbb{R}^{(\hat{r}+1)}.$$

Lemma 2.1 (Local transverse normal form existence [34])
Consider the augmented system (2.24) *and Assumptions 2.4–2.6, then the following statements hold:*
i) For all $(x, \hat{x})^T \in \mathcal{X}_0 \times \hat{\mathcal{X}}$ *system* (2.24) *has a well-defined vector relative degree* $\tilde{r} = (r_1, \ldots, r_{n_y}, \hat{r} + 1)$, *where* $r_1, \ldots, r_{n_y}$ *and* $\hat{r}$ *from* (2.29).
ii) For all $(x, \hat{x})^T \in \mathcal{X}, \times \hat{\mathcal{X}}$ *there exists a local diffeomorphism* $(\xi, \eta) = \Phi(x, \hat{x})$ *s.t.* (2.24) *is equivalent to a transverse normal form*

$$\dot{\xi}_i = \left(\begin{array}{c|c} \mathbf{0}^{r_i-1\times 1} & \mathbf{I}^{r_i-1\times r_i-1} \\ \hline 0 & \mathbf{0}^{1\times r_i-1} \end{array}\right) \xi_i + \begin{pmatrix} \mathbf{0}^{r_i-1\times 1} \\ \alpha_i(\xi_1, \ldots, \xi_{n_y}, \eta_1, \eta_2, u) \end{pmatrix}, \tag{2.32a}$$

$$\dot{\eta} = \beta(\xi, \eta, u, \hat{u}), \tag{2.32b}$$

where

$$\xi = \Big(\underbrace{e_1,\, \dot{e}_1, \ldots, e_1^{(r_1-1)}}_{\xi_1}, \underbrace{e_2,\, \dot{e}_2, \ldots, e_2^{(r_2-1)}}_{\xi_2}, \ldots, \underbrace{e_{n_y},\, \dot{e}_{n_y}, \ldots, e_{n_y}^{(r_{n_y}-1)}}_{\xi_{n_y}}\Big) \in \mathbb{R}^{\rho} \tag{2.33}$$

with $\rho = \sum_{i=1}^{n_y} r_i$ *and* $\eta \in \mathbb{R}^{n+\hat{r}+1-\rho}$.

Proof The detailed proof is given in [34]. ∎

According to Lemma 2.1, it is possible to map the coordinates $(x, \hat{x}) \in \mathcal{X}_0 \times \hat{\mathcal{X}}$ into new coordinates (ξ, η), which are valid locally. In this case, the feedback transformation of the system (2.24) maps and decomposes into a transversal ξ and tangential η linear subsystem with respect to the reference path $\mathcal{P}$ (2.22) (see Figure 2.4).

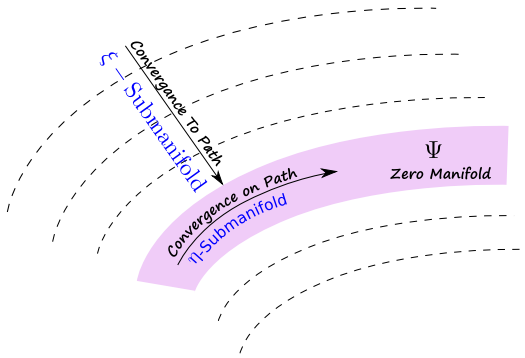


Figure 2.4: The geometric interpretation of new coordinates

To study the local coordinates (2.32) geometrically, we follow [34, 46]. First, we define the projection of the augmented states, i.e. $(x, \hat{x})$ from (2.24) onto the x coordinate as follows

$$\varpi : \ (x, \hat{x})^T \mapsto x. \tag{2.34}$$

Using this projection one can map the augmented system (2.24) to the original system (2.21). Now consider the following subset Γ, which represents all the points in the new local coordinates (ξ, η), which correspond to the local solutions of the system (2.21), such that the output (2.21b) is moving along $\mathcal{P}$ exactly

$$\Gamma = \left\{ (\xi, \eta) \in \mathbb{R}^{n+\hat{r}+1} \mid \xi = 0 \right\}, \tag{2.35}$$

where ξ represents the path error and its derivatives as in (2.33). So Γ is the subset when the path error and its derivatives are zero. Then we can describe the subset Γ in the original output system manifold (2.21b) by using the inverse of transformation map Φ, then projecting the result onto the x coordinates

$$\Psi = \varpi(\Phi^{-1}(\Gamma)). \tag{2.36}$$

Note that this inverse exists since Φ is a diffeomorphism. Moving toward the reference path $\mathcal{P}$ and traveling along it is equivalent to stabilize the submanifold Ψ (see Figure 2.4). Since this submanifold is not invariant [56, 89], it is possible only to stabilize the maximum controlled invariant subset Ψ^* of Ψ. Where Ψ^* means all solutions for which the system output (2.21b) is driven to travel along $\mathcal{P}$ for all times by an admissible control input u. However, Theorem A.2 in Appendix A presents sufficient condition of local unconstrained path followability.

So far we discussed the followability condition satisfaction for unconstrained dynamic systems. However, to deal with constrained dynamics system, we introduce here the term *Static State Feedback Control*. If the value of the feedback input at time t depends only on the current values of the state and perhaps external target input, then the feedback control is called a *Static State Feedback Control*[62]. Basically, if the nonlinear

dynamic system has a (vector) relative degree equal to the dimension of the state space, i.e. $\rho = n$, then it can be transformed into a system linear and controllable using a static feedback control. Also, if the (vector) relative degree equal to the dimension of the state space, then there is no internal dynamics will appear in normal form. This property can be used to ensure that there is no internal dynamics of the augmented system (2.24) in the transverse normal form (2.32) [34]. However, if the normal form includes internal states, then which may have unstable zero dynamics. As shown in [34] and references therein that the static feedback linearization control is a sufficient condition for differentially flat systems. Thus using flatness, sufficient conditions for exact path following of constrained systems can be set, for more details about this approach see [34].

2.3 Challenges

So far we assumed that there is a nominal NMPC setup, which means, full-state feedback and no measurement delay. In real-time applications of the NMPC scheme, some states are likely to be unmeasurable and have measurement delay. So one has to estimate the unmeasured states using observers or filters. In the following, we study briefly state-estimation and measurement delays and their effects on the stability and the performance of NMPC controllers.

2.3.1 State-estimation

As common in NMPC, the full state information is assumed to be available as initial condition for the prediction process, and can be measured. In practice, some states usually can not be measured and only an output is provided for feedback. Thus the application of NMPC based on a state space model needs an estimator to the unmeasurable states form the measured output using state observers. The main disadvantage of using the estimation process in the NMPC scheme is that nominal stability no longer can be guaranteed [45, 100]. One of the possible approaches to achieve closed-loop stability in the presence of observer errors, is to exploit robustness of the MPC [100]. In [45], it is pointed out, despite of the state feedback NMPC controller and the used observer being both stable, that there is no guarantee that the overall closed-loop is stable even with small region of attraction. As for nonlinear systems no separation principle holds.

There are two main approaches −and a combination of both− to recover a non-local stability of the observer-based output feedback NMPC controller [44, 61]. The first approach uses the certainty-equivalence principle. As a result, the stability can be achieved by separation of the observer error from the state feedback. This

separation can be done by time-scaling, i.e. the observer error needs to converge faster than the loop control response time and semi-regional stability can be achieved. The second approach includes the observer error inside the NMPC controller, commonly by bounding the observer error. This approach is equivalent to designing a robust NMPC by bounding and rejecting the disturbance. For a review of the existing observer-based output feedback NMPC approaches, we refer to [2, 45, 100]

To study the stability of the output feedback sampled-data NMPC, we follow the lines [45]. There the setup exploits the fact that sampled-data NMPC controllers have a continuous value function are inherently robust to small disturbances. The main idea is to consider the estimation error as a disturbance acting on the closed-loop. Then by using the inherent robustness of the sampled-data NMPC, the observer error effect can be tolerated.

To distinguish between the real state and the estimated one, we refer to the estimated state by $\tilde{x}(t)$. Here, at each sampling instant only the estimated state $\tilde{x}(t)$ is fed to the controller. Thus the applied optimal-input is also a function of the estimated state instead of the real state as in (2.5), so the new −disturbed− feedback control law is defined as follows:

$$u(t; \tilde{x}(t_i)) = \bar{u}^*(t; \tilde{x}(t_i)), \quad t \in [t_i,\, t_{i+1})\,. \tag{2.37}$$

It is possible that the estimated state $\tilde{x}(t_i)$ is outside the region of attraction of the state feedback. In this case, to keep the feasibility of the solution, we assume as in [45] that the input is fixed and bounded. Furthermore, we assume that the conditions of Theorem A.1 are achieved by the sampled-data NMPC controller. To counter the effect of the estimation error on the stability conditions, we adopt the following assumptions [45].

Assumption 2.7
For the nominal region of attraction $\Re \subseteq \mathcal{X} \subseteq \mathbb{R}^n$ the following holds
(i) Starting at a sampling instant t_i at $x(t_i) \in \Re$, along solution trajectories, the value function satisfies for all positive σ

$$V(x(t_i + \sigma)) - V(x(t_i)) \leq -\int_{t_i}^{t_i+\sigma} L(x(s), u(s; x(s_i)))\, ds. \tag{2.38}$$

(ii) The value function $V(x)$ is assumed to be uniformly continuous.
(iii) For all compact subsets $\mathscr{S} \subset \Re$ there is at least one level set

$$\pounds_c = \{x \in \Re \mid V(x) \leq c\}$$

such that $\mathscr{S} \subset \pounds_c$.

Assumption 2.7(i) ensures the stability of the state feedback sampled-data NMPC according to the Theorem A.1, since the value function is decreasing with time. Actually, the inequality in the Assumption 2.7(i) is analogous to the negative semi-definite time-derivative of a Lyapunov function, in this case the convergence to the origin can be proved using Barbalat's lemma. While the uniform continuity means that there exists a $\mathcal{K}$-function for any compact subset $\mathscr{S} \subset \Re$ such that for any $a, b \in \mathscr{S}$, $\|V(a) - V(b)\| \leq \alpha_{\mathscr{S}}(\|a - b\|)$.

Assumption 2.8 (Observer error convergence)
The maximum value of the estimation error $\epsilon_{max} > 0$ is given, and there exist observer parameters, such that

$$\|x(t_i) - \tilde{x}(t_i)\| \leq \epsilon_{max}, \quad \forall\ t_i \geq k_{obs}\delta \tag{2.39}$$

where $k_{obs} > 0$ can be chosen freely, but it should satisfy (2.39) *after a fixed number of sampling instants.*

There are a group of observers that allow to satisfy Assumption 2.8 e.g. moving horizon observers with contraction constraint and high-gain observers. Since Assumption 2.8 does not allow that the observer error goes to zero, thus neither the asymptotic stability nor rendering the whole region of attraction invariant can be achieved. [45]

The question is, can we render the system closed-loop to be semi-globally practically stable based on the Assumption 2.8? To answer the last question, we review the proposed approach in [45]. Then we show the challenge of bringing the observer-based feedback NMPC to be stable. Firstly, we start the answer by defining the semi-globally practically stability as desired stability property

Definition 2.5 (Semi-globally practically stability)
Assume there exist sets $\psi_a \subset \psi_b \subset \psi_c \subset \Re$, $0 < a < b < c$, with observer parameters and a maximum sampling time δ such that $\forall x(0) \in \psi_b$:

1. $x(t) \in \psi_c, \quad t > 0,$
2. $\exists\ t_a > 0 \quad s.t. \quad x(t) \in \psi_a, \quad \forall\ t > t_a.$

Figure 2.5 simplifies the Definition 2.5. The main idea of the proposed approach is based on the proportional relationship between state-estimation error and the predicted-trajectory error due to difference between estimated and real states. Which means, a small estimation error leads to a small predicted trajectory difference between estimated and real states if both of them stay in the set ψ_c. The effect of the

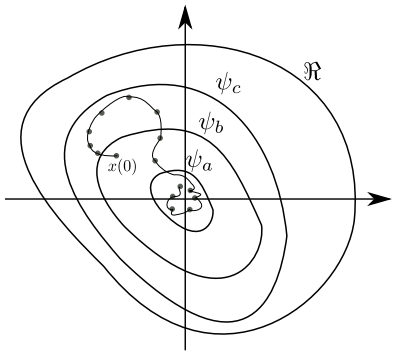


Figure 2.5: Set of initial conditions ψ_b, desired maximum attainable set ψ_c and desired region of convergence ψ_a [45].

estimation error when using the observer can be bounded by

$$\begin{aligned} V(x(t_i)) &- V(x(t_{i+1})) \leq e_{obs}(\|x(t_i) - \tilde{x}(t_i)\|) \\ &- \int_{t_i}^{t_{i+1}} L(x(\tau; u^*(\cdot; \tilde{x}(t_i)), x(t_i)), u^*(\tau; \tilde{x}(t_i)))\, d\tau, \end{aligned} \tag{2.40}$$

where e_{obs} represents the state estimation error interference. Since the integral term in (2.40) is strictly negative, and e_{obs} is proportional to the size of the observer error and the sampling time. Then the contraction of the value function is attainable. According to Assumption 2.8, the upper bound on e_{obs} is achievable after a specified time, i.e. $(k_{obs}\delta)$. To keep the system state within the set ψ_c until reaching this time, we must decrease the sampling time δ or use an adaptive sampling time. While for bounding the integral term in (2.40), we consider the following:

Fact 1. For any $c > a > 0$ with $\psi_c \subset \Re$, $N > \delta > 0$ (N is the prediction horizon) the lower bound $V_{min}(c, a, \delta)$ on the value function exists and is non-trivial $\forall x_0 \in \psi_c/\psi_a$:

$$V_{min}(c, a, \delta) := \min_{x_0 \in \psi_c/\psi_a} \int_0^\delta L(\bar{x}(s; \bar{u}^*(\cdot; x_0), x_0), \bar{u}^*(s; x_0))\, ds < \infty.$$

Note that the function $L(\cdot, \cdot)$ is assumed to be lower bounded by $\mathcal{K}-$function. Now, we finish the answer to the raised question by the following theorem.

Theorem 2.1 [45]
If Assumptions 2.7 and 2.8 hold then arbitrary level sets $\psi_a \subset \psi_b \subset \psi_c \subset \Re$ are given, and there exists a maximum allowable observer error ϵ_{max} and a maximum sampling time δ_{max}, such that for all states $x(0) \in \psi_b$ the state $x(\tau)$ does not leave the set ψ_c and converges within finite time to the set ψ_a.

Proof :

As outlined in [45], the proof of Theorem 2.1 consists of three parts. Here we do not review a complete proof, instead we show only the first part of the proof, since it is important to show the tradeoff between the sampling time and computation time

of the controller to achieve stability of the observer-based output feedback NMPC controller, as we will see this issue in later chapters.
In the first part of the proof, we prove that the system state stays in the maximum admissible set ψ_c until convergence time $k_{obs}\delta$ is reached. We want to achieve that $(x(\tau) \in \psi_c \forall x(0) \in \psi_b, \tau \in [0, k_{obs}\delta))$, where $\psi_b \subset \psi_c$. We assume there are sets such that,

$$(\psi_b \subset \psi_{c_1} \subset \psi_{c_2} \subset \psi_c), \text{ with } c_1 := b + (c - b/2) \text{ and } c_2 := c_1 + (c - c_1/2).$$

Accordingly, there exists a time t_{b-c_1} such that

$$x(\tau) \in \psi_{c_1}, \forall\ 0 \leq \tau \leq t_{b-c_1}.$$

This time can be guaranteed by considering that

$$x(\tau) \in \psi_c, \|x(\tau) - x(0)\| \leq \int_0^\tau \|f(x(s), u(s))\|\ ds \leq k_{\psi_c}\tau,$$

here k_{ψ_c} is a constant related to the Lipschitz constant of f and the limits on u. t_{b-c_1} is considered as the smallest time to touch the boundary of set ψ_{c_1} for any point starting at ψ_b driven by any admissible input value $u \in \mathcal{U}$. Using the same arguments, there exists a time t_{c_2-c} such that

$$\forall x(t_i) \in \psi_{c_2}, x(\tau)\psi_c,\ \forall \tau \in [t_i, t_i + t_{c_2-c}).$$

Then the maximum sampling time δ_{max} can be chosen such that:

$$\delta_{max} \leq \min\left\{t_{b-c_1}/k_{obs},\ t_{c_2-c}\right\}. \tag{2.41}$$

It is assumed that the sampling time δ to be less or equal to the maximum sampling time δ_{max}. The remaining two parts of the proof are similar to [45] and can be found in Appendix D ∎

For stability according to this result, there is an upper bound on the maximum sampling time (2.41). While the lower bound on the sampling time is the computation time of the controller and observer. Which means that, before the next recalculation instant starts, the controller has to provide an input to the system. As shown in the Figure 2.6, we denote the controller computation time by T_c, the observer calculation time T_{obs}, and the system evolution time between each two recalculation instants T_{sys}, which is identical to the sampling time δ.

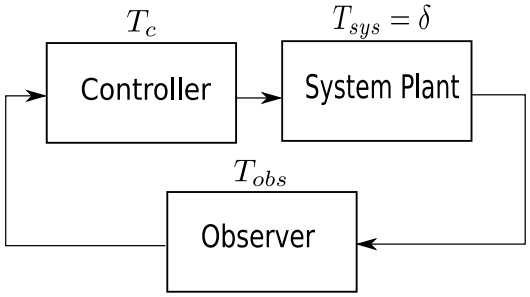


Figure 2.6: Closed-loop system

Then we conclude that the sampling time is bounded as follows:

$$(T_c + T_{obs}) < \delta \leq \delta_{max}. \tag{2.42}$$

As discussed in Section 2.1.2 a long predication horizon is better for achieving stability, but it increases the computation time of the NMPC controller. Therefore reducing the computation time of the NMPC controller is necessary to make the lower bound on the sampling time smaller. This ensures the convergence of the observer-based output feedback NMPC controller. In the next section, we discuss the problem measurement delays and its effect on the stability and performance of the NMPC controller.

2.3.2 Measurement and Computational Delays

In this section we briefly discuss the measurement and computational delays and their effect on the stability and the performance of NMPC controller. There are many sources of delay e.g. using state observer, sensors, real-time scheduling [41]. Actually, small time delays can be considered as a disturbance, which can be rejected by the inherent robustness of NMPC controllers. However, it is not a trivial task to specify the degree of robustness a priori or even existence of inherent robustness to deal with delays [41, 43]. It is important to specify delays, since unknown delays might lead to bad performance or instability of the closed-loop. For the sake of simplicity, we assume a constant sampling time δ, and a constant time delay denoted by δ^d. Basically, the effect of measurement and computational delays is the same, which lead to input trajectory that different to the current state. Therefore, this discrepancy might introduce instability of the closed-loop system.
It is assumed that the maximum time delay for both computations and measurements to be strictly less than the sampling time $\delta^d \leq \delta$. To deal with delays, first we give the mathematical representation. In measurement delays the measured state is provided at time t_i but it is actually measured at time $t_i - \delta^d$, i.e. $x_{t_i} = x(t_i - \delta^d)$. While in computational delays the open-loop feedback trajectory for time instant t_i is only available at time $t_i + \delta^d$. To overcome the effect of delays in both cases, one can use the feedforward simulation of the dynamic system (assuming no disturbance is acting

on the system) over the delay time to estimate the system state, which fits with the current state. Then the estimated state is provided to the controller for computing a corrected or shifted input trajectory to compensate the delay effect [41]. For measurement delays, we assume that the open-loop optimal input from previous time instants is stored, i.e. $u(\cdot; x(t_{i-1}), t_{i-1})$, such that the estimated state can be produced using forward simulation of the dynamic system as follows:

$$\dot{\tilde{x}} = f(\tilde{x}(\tau), u(\tau; x(t_{i-1}), t_{i-1})), \quad \tilde{x}(t_i - \delta^d) = x(t_i - \delta^d), \quad \tau \in \left[t_i - \delta^d, t_i\right].$$

For computational delays, one requires a shift of the applied input by time delay δ^d. This can be achieved by shifting the initial states forward using open-loop simulation of dynamic system using previous optimal open-loop input $u(\cdot; x(t_{i-1} + \delta^d), t_{i-1} + \delta^d)$, such that

$$\dot{\tilde{x}} = f(\tilde{x}(\tau), u(\tau; x(t_{i-1} + \delta^d), t_{i-1} + \delta^d)), \quad \tilde{x}(t_i) = x(t_i), \quad \tau \in \left[t_i, t_i + \delta^d\right].$$

When controlling fast dynamic systems with measurements delay less than the sampling time, NMPC controller can cope with this delay depending on inherent robustness of the MPC. So far we assumed that the time delay is constant. However, in [75], NMPC controller subject to time-varying measurement delays is studies. In this approach, the stability of the system subject to time varying measurement delays can be recovered by modifying the constraints of the Lyapunov-based model predictive controller.

Summary

In this chapter, we reviewed briefly the principle and the mathematical description of NMPC controllers. Then we discussed the controller parameters effect on both the stability and performance. Additionally, we reviewed the existing methods of stability. Afterwards, we presented a brief overview of predictive path-following problem in state and output spaces, as well as, the conditions for path followability for unconstrained and constrained dynamics system. Finally, we outlined the effects of state-estimation and delays on the stability of NMPC. So far, in this chapter, the position control part of our unified path-following force-feedback controller is presented. In the next chapter, we discuss existing force-feedback control schemes.

3 Force Feedback Basics

In this chapter, we give a brief overview on force feedback control. We outline active and passive compliance-based control methods. We focus on active compliance methods, in particular, *impedance* and *admittance* force control, with notes about their advantages and limitations. In addition, we discuss the environment identification −any object/surface with which the robot makes contact−, and explain the way of choosing admittance parameters based-on the duality principle. We furthermore review optimization-based force feedback control approaches.

3.1 Introduction

Many manipulation tasks of robotic manipulators require an interaction with the environment. Examples are milling, deburring, welding/soldering, twisting, grinding, pounding, polishing, and cutting. Using a motion control strategy alone, i.e. neglecting the force, is insufficient or even leads to an unstable control system, unless the interaction captured by a model accurately [15]. Both the robot manipulator model and the environment model need to be rather exact. In practice this is seldom the case, basically, a manipulator model might be known with an acceptable accuracy, but an accurate model of the environment is not in general easy to obtain. The interaction -compliant behavior- can be classified into two categories: *Passive Compliance* and *Active Compliance* [110], leading to passive and active compliance control.

Passive compliance is induced by the structural mechanical compliance of the robot manipulator. Here compliance is defined by the finite stiffness of the links, grippers, joints, actuators, and the manipulated objects. In classical passive compliance control/design, the measurement of force is not required [10, 110]. In active compliance, the interaction between the end-effector and the environment is controlled by changing the "joint stiffness" by an user-defined control law such that the interaction between a manipulator's end-effector and the environment is considered virtually as a general mass−spring−damper system driven by the contact force. Hence, a force measurement is required in the active compliance approach. The measured force is fed back to the controller and compared with the desired force to update the trajectory.
In the field of robotics, there exist two major approaches for active compliance control: indirect-force control and direct-force control [53, 108, 110, 128]. Indirect force control is achieved implicitly by motion control, without an explicit force feedback loop. In

contrast, direct force control employs a force feedback loop. To this end, the indirect force control approach is realized by using so-called impedance (or admittance) control. The aim of impedance/admittance control is to design a desired dynamic interaction between the end-effector position and the environment, which is controlling the position and the force, whether the contact exists or not [24, 57, 70, 107]. Namely, any active compliance approach receives both motion and force errors, to provide the proper input to the robot.

Impedance/admittance control is often realized in form of a mass−spring−damper dynamic system with adjustable parameters [57, 107, 110]. The active (indirect) compliance is an impedance if the control reacts to the position error by generating forces. It is called an admittance if the controller complies to contact forces by producing a deviation from the desired position. We use *admittance force control*, as our approach updates the desired position based on the measured force.

One can classify force control methods according to the desired region of interest in the system response, i.e. static (steady-state response) and dynamic (transient response) model-based force control methods [24, 39]. Static model-based control methods are easier to implement, as they often only include the gravity term and do not rely on the dynamic models. Impedance static model-based control is called *stiffness force control*, while the admittance static model-based control is denoted by *compliance force control* [39]. The dynamic model-based force control methods include in-direct-force control (i.e impedance and admittance), and direct-force control (i.e. hybrid position/force control [53]). Dynamic model-based control methods require a dynamic model of the system -robot- which leads to additional complexity in design and implementation.

Remark 3.1 (Actuator Bandwidth)
Actuator bandwidth specifies the frequency range at which the commanded forces can be tracked accurately by the robot. A perfect actuator for achieving force control is one that can be considered as an ideal force source, which means that it generates the commanded force exactly. However, this is infeasible practically, since the moving load produces an additional force to the actuator output [39].

3.2 Impedance-based Force-Feedback Control

We now outline a mathematical setup for impedance control. As mentioned, the aim of impedance control is to establish a dynamic relation between the end-effector and the environment. The objective of impedance control is to control in each direction of the task space the dynamic relation between motion error and contact force. First,

we will describe the model of the manipulator dynamics in the task space coordinates. Then we add a desired impedance at the end-effector of the manipulator by exploiting the computed-torque control method.

3.2.1 Task space dynamic model

In general, a dynamic model of n-DOF robotic system in joint-space can be written as [102]:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) + \tau_F(\dot{q}) = \tau_e - J^T(q)f_e. \tag{3.1}$$

Here $q \in \mathbb{R}^n_x$ is the vector of joint angular positions, and its time derivatives $\dot{q}$ and $\ddot{q}$ represent the angular velocity and angular acceleration respectively. The vector $\tau_e \in \mathbb{R}^n_x$ refers to the applied torques on the corresponding joints. The inertia matrix $M(q) \in \mathbb{R}^{n_x \times n_x}$ is a symmetric positive definite matrix; $C(q,\dot{q}) \in \mathbb{R}^{n_x \times n_x}$ are the Coriolis and centrifugal effects; $g(q) \in \mathbb{R}^{n_x}$ represents the gravitational torque; and $\tau_F(\dot{q}) \in \mathbb{R}^{n_x}$ is the friction torque in the joints, while J^T is the transposed Jacobian matrix, which maps the Cartesian external force f_e into the joint-space.

Remark 3.2 (Geometric and Analytical Jacobians)
The manipulator Jacobian matrix $J(q) \in \mathbb{R}^{6\times n}$ maps the joint displacement dq to the corresponding end-effector displacement dy [102]:

$$dy = J(q)dq. \tag{3.2}$$

Commonly, the Jacobian matrix follows from the geometry by computing the contribution of each joint velocity to the linear and angular velocities of the end-effector. Therefore, the Jacobian can also be termed as the geometric Jacobian $J_g(q)$ of the manipulator. If the end-effector position and orientation are represented by the manipulator parameters in the task space using a forward kinematics map $T_{ca} : \mathbb{R}^n_x \to \mathbb{R}^{n_y}$, the joint positions of the robot in Cartesian space are given by

$$y = T_{ca}(q). \tag{3.3}$$

Thus is possible to compute the Jacobian matrix by direct differentiation of the forward kinematic equation, i.e.

$$\dot{y} = J_a(q)\dot{q}, \tag{3.4}$$

where the matrix $J_a(q) = \frac{\partial T}{\partial q}$ is the analytical Jacobian [28]. The relationship between the geometric and the analytical Jacobians is found as follows

$$J_g(q) = T_a(\phi_e)J_a(q), \tag{3.5}$$

where $T_a(\phi_e)$ is a transformation matrix, which depends on the set of parameters representing the end-effector orientation [28].
One can see that the two Jacobians are in general not the same, however, the two Jacobians are identical for the linear part. Regarding their implementation, the geometric Jacobian is used when physical quantities are important, while the analytical Jacobian is chosen when the task space quantities are of interest. It is possible to transform one Jacobian into the other, if and only if the transformation matrix $T_a(\phi_e)$ is non-singular [28].

We use the analytical Jacobian $J_a(q)$, thus the model (3.1) can be rewritten as follows:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) + \tau_F(\dot{q}) = \tau_e - J_a^T(q) f_e. \tag{3.6}$$

The interaction between the manipulator and the environment takes place in the task-space. Therefore, we express the model of manipulator dynamics in task-space coordinates. To this end, first, we need to take the second time derivative of (3.4) to obtain the map between the Cartesian acceleration and joint acceleration:

$$\ddot{y} = J_a(q)\ddot{q} + \dot{J}_a(q)\dot{q}. \tag{3.7}$$

Substituting (3.7) into (3.6) gives the dynamic model in the task-space coordinates:

$$M_y(q)\ddot{y} + C_y(q,\dot{q})\dot{y} + g_y(q) + \tau_{Fy}(\dot{q}) = J_a^{-T}(q)\tau_e - f_e \tag{3.8}$$

where

$$\begin{aligned} M_y(q) &= J_a(q)^{-T} M(q) J_a^{-1}, \\ C_y(q,\dot{q}) &= J_a(q)^{-T} C(q,\dot{q}) J_a^{-1} - M_y(q)\dot{J}_a(q) J_a^{-1}, \\ g_y(q) &= J_a^{-T}(q) g(q), \\ \tau_{Fy}(\dot{q}) &= J_a^{-T}(q)\tau_F(\dot{q}). \end{aligned}$$

As one can see the arguments of the nonlinear terms in (3.8) remain in $(q, \dot{q})$. However, it is possible to transform them into the coordinates $(y,\ \dot{y})$ using the inverse kinematics transformation. Keeping these arguments dependent on the joint variables is often better from computational perspective [28].

The end-effector's contact force vector with the environment is defined as $f_e = [f_n^T\ f_t^T]^T$, where f_n stands for the normal contact force, while f_t represents the tangential contact force due to a friction contact with the environment. When a contact force arises during interaction, then the convergence to the desired trajectory is no longer ensured. Basically, there will be a steady-state offset which is a trade-off between the force and position error convergence.

In general, perfect modeling of the contact between the end-effector and the environment is difficult due to uncertainties in the robot dynamics and the imprecise knowledge of the environment stiffness. For simplicity the environment deformation model is assumed to be given by a linear spring with symmetric stiffness matrix $K_e \in \mathbb{R}^{n_y \times n_y}$. Let $y_e \in \mathbb{R}^{n_y}$ be the position of the undeformed environment. According to these assumptions, when the end-effector is in contact with the environment, a generalized normal force

$$f_n = K_e(h(q) - y_e); \quad y(t) = T_{ca}(q) \geq y_e, \tag{3.9}$$

is exerted at the end-effector, where $(h(q) - y_e)$ represents the penetration depth. Note, however, that we are expressing all output and force variables relative to a fixed reference frame. The tangential contact force f_t is modeled, as proposed in [13, 126]:

$$f_t = \mu \, |f_n| \, \mathrm{sgn}(\dot{y}), \tag{3.10}$$

where μ is the dry friction coefficient between the end-effector and the environment, and $\dot{y}$ the unconstrained velocity state. The aim of impedance control is to establish a desired dynamic interaction between the manipulator end-effector and the environment. Next, we describe the acceleration-resolved approach, the computed-torque control method, which is used to decouple and linearize the nonlinear dynamics of the manipulator at the acceleration level.

3.2.2 Computed-torque control

To control the interaction between the robot and the environment at the contact point, first we decouple and linearize the closed-loop dynamics in the task-space such that

$$\ddot{y} = u_c, \tag{3.11}$$

where u_c is an admissible control delivered from a controller. To this end, the complete dynamic model of the manipulator is used to cancel the effect of Coriolis and centrifugal force, gravity, friction, and the manipulator inertia tensor. Normally, the estimation of robotic arm parameters are used to implement computed-torque control. By assuming that $\left\{\tilde{M}_y(q), \tilde{C}_y(q,\dot{q}), \tilde{g}_y(q), \tilde{\tau}_{Fy}(\dot{q})\right\}$ are the estimated values of the robot dynamics, then the feedback linearization control law is given by [102]

$$\tau_e = J_a^T(q)[\tilde{M}_y(q)u_c + \tilde{C}_y(q,\dot{q}) + \tilde{g}_y(q) + \tilde{\tau}_{Fy}(\dot{q}) + f_e]. \tag{3.12}$$

By applying (3.12) to (3.6) we obtain

$$M_y(q)\ddot{y} \; = \; \tilde{M}_y(q)\, u_c \; + \; \tilde{r}, \tag{3.13}$$

where $\tilde{r}$ is the residual due to the mismatch between real and estimated dynamic parameters, which is defined as

$$\tilde{r} = [C_y(q, \dot{q}) - \tilde{C}_y(q, \dot{q})]\dot{y} + [g_y(q) - \tilde{g}_y(q)] + [\tau_{Fy}(\dot{q}) - \tilde{\tau}_{Fy}(\dot{q})]. \tag{3.14}$$

If the robotic arm parameters are estimated correctly, then (3.13) becomes a linear second-order equation, independent of the robotic arm parameters. The nonlinear terms in (3.13) are completely compensated.

As common, the desired impedance is modeled as a second order mass-spring-damper system [108, 122]:

$$M_d[\ddot{y}(t) - \ddot{y}_d(t)] + B_d[\dot{y}(t) - \dot{y}_d(t)] + K_d[y(t) - y_d(t)] = E_f(t), \tag{3.15}$$

where $y(t) \in \mathbb{R}^{n_y}$ is the current desired Cartesian position of the end-effector based on the impedance dynamic, and $y_d(t) \in \mathbb{R}^{n_y}$ represents the desired position trajectory, M_d, B_d, and K_d are diagonal $\mathbb{R}^{n_y \times n_y}$ positive definite matrices of desired mass, damping, and stiffness gains respectively, defined by the user, and $E_f(t) = F^d - f_e$ where F^d is a desired force vector.
To achieve a desired mechanical impedance, the control input can be chosen follows [102, 121],

$$u_c = \ddot{y}_d(t) + M_d^{-1}\left[B_d(\dot{y}_d(t) - \dot{y}(t)) + K_d(y_d(t) - y(t)) + E_f(t)\right]. \tag{3.16}$$

If we substitute (3.16) in (3.11), then, the closed-loop system becomes

$$M_d\,\ddot{e}_d(t) + B_d\,\dot{e}_d(t) + K_d\,e_d(t) = -E_f(t), \tag{3.17}$$

where $e_d = (y_d(t) - y(t))$.

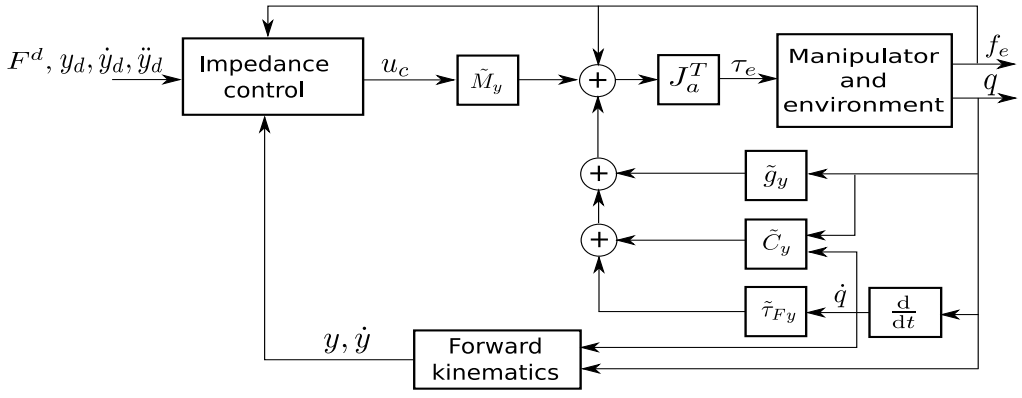


Figure 3.1: Computed-torque based impedance control.

In Figure 3.1 a schematic diagram of the computed-torque control based impedance is sketched. Basically, the impedance control delivers the input (acceleration) accord-

ing to (3.16) based on the current information of the position and orientation, in addition to the force and moment measurements. Then, the computed-torque control law computes the torques for the joint actuators according to (3.12). In the case that there is no interaction, this control approach guarantees that the end-effector asymptotically converges to the desired position. In the presence of interaction with the environment, a virtual dynamic compliance is imposed on the end-effector, i.e. the desired impedance (3.15) as well as, a force/torque sensor is needed to measure the contact force and moment.

3.2.3 Implementation challenges of impedance-based control

As shown in the previous section, for implementing the control scheme, it is required that both the model of the environment and the robot dynamics are accurately known. However, choosing good impedance parameters achieving the desired performance is not in general an easy task, the closed-loop dynamics differs in case an interaction exists. Control objective during interaction is to achieve the best compliance. While during free motion, the objective is to track the desired trajectory and reject the disturbance. Also, the coupling between the manipulator dynamics and the environment dynamic model exists during the interaction [39, 121]. To investigate these problems, we follow [121]. For simplicity, we assume that the environment is modeled as a linear spring:

Definition 3.1 (Mechanical Springs) [121]
Assuming there are two elastically coupled rigid bodies B_1 and B_2 attached to the reference frames Σ_1 and Σ_2, respectively. Consider that the frames Σ_1 and Σ_2 coincide at the equilibrium, then one can describe the compliant behavior in the vicinity of the equilibrium by

$$f_1^2 \;=\; K\,\delta y_{21}^2 \;=\; \begin{pmatrix} K_t & K_c \\ K_c^T & K_r \end{pmatrix} \delta y_{21}^2, \tag{3.18}$$

where f_1^2 is the elastic wrench applied to the body B_1 expressed in the frame Σ_2, when there is an infinitesimal movement δy_{21}^2 of the frame Σ_2 with respect to the frame Σ_1. The elastic wrench and the infinitesimal movement in (3.18) *can also be defined interchangeably in the frame Σ_1, since Σ_1 and Σ_2 are identical at the equilibrium point. In this case, $f_1^1 = f_1^2$ and $\delta y_{21}^1 = \delta y_{21}^2$ as well as the elastic wrench applied to body B_2, $f_2^2 = K_t\,\delta y_{12}^2 = -f_1^1$ implies that $\delta y_{21}^1 = -\delta y_{12}^2$. This property of* (3.18) *is called a port symmetry. Here, $K \in \mathbb{R}^{6\times 6}$ is the symmetric positive semidefinite stiffness matrix. The symmetric matrices $(K_t, K_r) \in \mathbb{R}^{3\times 3}$ are called the translational stiffness and rotational stiffness respectively. While, $K_c \in \mathbb{R}^{3\times 3}$ is called the coupling stiffness. If the matrix K_c is symmetric, then there exists a maximum decoupling between the translation and rotation. Accordingly the point of coinciding origins of the frames Σ_1*

and Σ_2 is denoted by the center of stiffness.

In the following, we study the interaction between the manipulator's end-effector and the environment using Definition 3.1. The frame attached to the manipulator's end-effector is denoted by Σ_m and the frame attached to the origin of the environment by Σ_e. Based on (3.18), the wrench applied by the end-effector on the environment corresponding to the infinitesimal movement δy^e_{me} can be given by

$$f^m_e \;=\; K\,\delta y^m_{me}. \tag{3.19}$$

It is worth to mention that the model (3.19) holds only during an interaction, while in free motion of the end-effector the contact wrench is equal to zero.

The effects of disturbances on the robot manipulator and the uncertainties may be considered as additive disturbances on the right-hand side of the dynamic model (3.1). This represents a disturbance wrench $\mathcal{D}$ applied on the end-effector and appears as well on the right-hand side of (3.11). Then, by applying the control law (3.12), as a result the closed-loop impedance will be as follows:

$$M_d\,\ddot{e}^m_d(t) + B_d\,\dot{e}^m_d(t) + K_d\,e^m_d(t) = f^m_e(t) + M_d\,\mathcal{D}^m, \tag{3.20}$$

here, for simplicity, the desired force is assumed to be zero, i.e. $F^d = 0$. Substituting (3.19) into (3.20) we obtain

$$M_d\,\ddot{e}^m_d(t) + B_d\,\dot{e}^m_d(t) + (K_d + K)\,\delta y^m_{dm}(t) = K\delta y^m_{de} + M_d\,\mathcal{D}^m, \tag{3.21}$$

where $\delta\, y^m_{dm} = \delta\, y^m_{de} - \delta\, y^m_{me}$. The model (3.21) is valid when interaction exists ($K \neq 0$) or not ($K = 0$). The transient behavior of the error $e_d(t)$ can be chosen by tuning the matrices M_d, B_d, and K_d. Assuming that all the matrices are diagonal, one can decouple (3.21) for the six dynamic sub-systems corresponding to the infinitesimal twist displacement. Accordingly, the transient of each dynamic sub-system can be specified by choosing the natural frequency and the damping ratio using the following equations

$$\omega_n = \sqrt{\frac{k_d + k}{m}}, \quad \zeta = \frac{b_d}{2\sqrt{m(k_d + k)}}, \tag{3.22}$$

where k_d, b_d, k, m are the diagonal elements of the matrices K_d, B_d, K, M_d respectively. Due to the contribution of k in the last equations, even if the gains are designed to give adequate natural frequency and damping ratio during the interaction, these values are no longer the same when the end-effector moves in free space, i.e. for $k = 0$.

The steady-state position error and contact force can be defined as follows

$$\delta y_{dm}^{ss} = \frac{k}{k_d + k}\delta y_{de} + \frac{m}{k_d + k}\mathcal{D}$$
$$f^{ss} = \frac{k_d\, k}{k_d + k}\delta y_{de} - \frac{m\, k}{k_d + k}\mathcal{D}.$$

The last relations show that by setting the active stiffness k_d low the contact force gets also small but the position steady-state error becomes large. Furthermore, the disturbance term increases both the position error and contact force for low active stiffness. However, the effect of external disturbance still exists on the steady-state error position even when there is no interaction, i.e. $k = 0$. In the next section, we introduce an approach to overcome the last drawbacks in impedance control.

3.3 Admittance-based Force-Feedback Control

In this section, we review how to overcome the interference of force control with motion control. One can separate them by using the impedance control as a generator of a desired position based on the measured force and -*virtual*- reference position, which depends on the desired contact force, environment position, and environment stiffness, as will be shown later. The desired position is then delivered to a motion controller. Here, the measured force is used to compute the desired position. This scheme is called *admittance* control (see Figure 3.2). In the literature, admittance control is also called *indirect impedance* [125]/ *position-based impedance* [115]. As these names indicate, the principle of admittance control is to modify the desired position of the end-effector, such that the desired robot-environment interaction force is achieved.

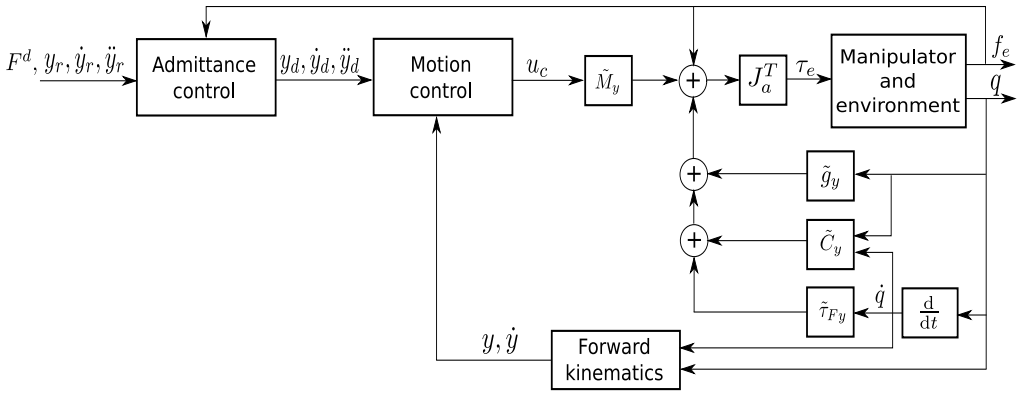


Figure 3.2: Computed-torque control based admittance.

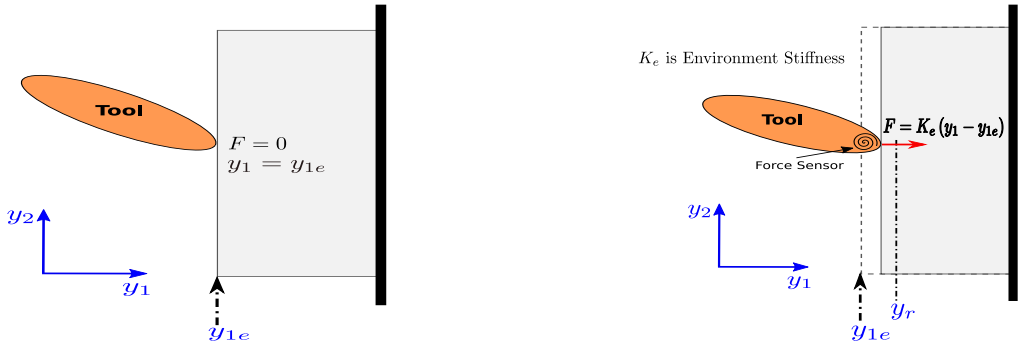
To this end, we define the admittance control law as follows

$$M_d\left[\ddot{y}_d(t) - \ddot{y}_r(t)\right] + B_d\left[\dot{y}_d(t) - \dot{y}_r(t)\right] + K_d\left[y_d(t) - y_r(t)\right] = -f_e(t), \tag{3.23}$$

where $y_d(t) \in \mathbb{R}^{n_y}$ is the current desired Cartesian position of the end-effector based on the admittance dynamic, and $y_r(t) \in \mathbb{R}^{n_y}$ represents the reference position trajectory -*virtual trajectory*- that depends on the desired contact force, environment position, and environment stiffness.
Then, the current desired position is delivered to the motion controller, based on the computed-torque control, the end-effector drives to the desired position. The stability of the overall system, can be ensured if the bandwidth of the motion controller is higher than the bandwidth of the admittance controller [121]. Figure 3.2 shows that, if there is no interaction, then the virtual reference is equal to the desired position. Accordingly, the disturbance rejection depends entirely on the motion controller.

3.4 Environment Modeling

As outlined indirect force control schemes require a model of the environment. This model must be accurate to be able to govern the interaction between the robot and the environment. Normally, a linear model so-called Kevin-Volgt model is chosen, which is similar to a linear spring (see (3.9) and Figure 3.3). Figure 3.3 shows the concept, such that a manipulator touches the environment and exerting normal force, i.e. in y_1-direction while moving towards the desired position y_r, which is corresponding to the desired force.



(a) Before interaction (b) After interaction

Figure 3.3: Environment modeling

For stiff and soft environment, other models exist, e.g. the Hunt-Crossley [31, 39] model. This model describes the nonlinear behavior of the environment. It can be computed on-line, since it does not require expensive computations. This model is given by,

$$f_e(t) = \alpha_1\, y^n(t) + \alpha_2\, y^n(t)\, \dot{y}(t), \qquad y \geq 0, \tag{3.24}$$

where n is a real number related to the geometry of the contact surface. However, to achieve the force control objective when using the impedance/admittance controller, it is necessary to have a perfect model of the environment. There are many approaches to estimate the parameters of the environment in real-time. For example, in [77, 106] adaptive impedance controllers are used to compensate the uncertainties in both the robot model and environment parameters, as well as, imperfect force sensor measurements. In [106] the impedance/admittance reference trajectory is driven by estimation of both the environment stiffness and undeformed position. Then, this trajectory is used to track the desired contact-force. In [33, 66] methods based-on neural networks are used, e.g. in [66] a neural network is used to tackle uncertainties in the modeling, and the force sensor noise is compensated. In [112] a Kalman filter is used to estimate the environment parameters in real-time. We adopt in the following the Kevin-Volgt model.

3.5 Choosing Parameters Based-on The Duality Principle

In this section, we describe the duality principle, which is used to select a desired manipulator impedance in order to achieve a trade-off between position and force tracking [30]. Basis of the duality principle is that the manipulator desired impedance/admittance has to be designed as the dual to the environment [9]. This principle is used intuitively by humans, e.g. when someone throws a heavy object (stiff) to another person, the catcher will be non-stiff, i.e. move his hands along the object course to avoid huge impacts. To minimize the position and contact force errors, the manipulator desired impedance should theoretically be designed equal to the environment admittance [57]. We denote the environment impedance as $Z_e(\omega)$, which is characterized as:

$$
\begin{aligned}
&|Z_e(0)| = 0, \quad \text{for inertial,} \\
&|Z_e(0)| = \alpha_z, \quad 0 < \alpha_z < \infty \quad \text{for resistive,} \\
&|Z_e(0)| = \infty, \quad \text{for capacitive.}
\end{aligned}
$$

At steady-state $\omega = 0$, an inertial environment impedance behaves as a force source, while a capacitive impedance is considered as a position source. Therefore, both inertial and capacitive environments are dual to each other. According to the dual principle, the position control is realized when the environment impedance is inertial, i.e. $Z_e(0) = 0$, and the manipulator/desired impedance is capacitive, i.e. $Z_m(0) = \infty$. The force control can be achieved when the environment impedance is capacitive, i.e. $Z_e(0) = \infty$, and the manipulator/desired impedance is inertial, i.e. $Z_m(0) = 0$. A resistive environment is position controlled when $Z_m(0) = \infty$, and considered as force controlled $Z_m(0) = 0$. Based-on the duality principle, the tracking of position or force

trajectory in Cartesian space can be done by choosing a desired impedance based on the measured environment impedance.
So far, we discussed the duality principle for passive impedances. To extend the duality principle to variable environment impedances, [30] used optimization techniques to minimize the sum of position and force tracking errors, for compromising between position and force tracking. This approach is discussed in the following section.

3.6 Optimization-based Force-Feedback Control

Optimization-based force control approaches can be classified into two main categories:

- **Optimal impedance in the Joint-space:** In this approach, the coordinated joint impedance trajectories with desired joint velocities are optimized using an optimal policy. Furthermore, transmission stiffness is chosen to ensure a minimum implementation time of the robot task, while achieving an intrinsic safety for any undesired interaction with the environment or a human operator [119].
- **Optimal impedance in the Configuration-space:** Commonly, optimization-based force control research is using an optimal policy, which is minimizing the error of the force/force time-derivative or compromising between force and position error metric [13, 19, 30, 82, 94, 95].

We control the contact force applied on the robot's end-effector, therefore, we focus on the second category, which we further detail in the following.

- **Minimization of force error and (its time derivative)**

  In [94] the admittance controller is obtained as the solution of a dynamic optimization problem. There, the optimization problem is used to minimize the force error and its time derivative. In [13] a fuzzy predictive algorithm computes an optimized virtual trajectory for the impedance controller on-line, where the virtual trajectory is optimized by using the force error as a performance index. The approach proposed in [95] minimizes the interaction force error at the robot end effector, while constraining undesired interaction forces. Since the force time derivative is normally an noisy signal, it is not desirable to use the force time-derivative. Some approaches overcome this drawback depending on the inherent filtering properties of the time integration [94].

- **Optimization of a combined position and force trajectory error metric:**

  The numerical optimization results are used to determine a desired impedance parameters based on the environment and manipulator dynamics [30, 82]. In [82] the parameters of the target impedance are determined analytically and on-line

by solving a quadratic optimal control problem. There, the performance index is formulated as the integral of the position error and the force error.

Choosing a cost function to optimize is challenging. To show the challenge of using a cost function compromising between position and force errors, we follow the lines of [30]. There, the desired impedance parameters are chosen depending on an optimal performance using a geometric view. To do so, the impedance controller is assumed to implement the desired dynamics ideally. The environment is assumed to be highly stiff, which is modeled as a linear spring system (3.9).

To this end, a trade-off between position and force errors can be found by minimizing the sum of squared position and force errors (3.25) with respect to the desired impedance function.

$$\min J \;=\; c\,(y_d - y)^2 \;+\; (f_d - f_e)^2. \tag{3.25}$$

Here, the scalar c is used as a weighting to shift the relative sensitivity between position and force errors. Considering a static condition, the desired impedance is reduced to

$$(f_d - f_e) = -K_d(y - y_d). \tag{3.26}$$

Substituting (3.26) into (3.9), the end-effector position and applied contact force can be written as functions of the desired position, force, environment stiffness, and active stiffness as follows

$$y \;=\; \frac{1}{k_d + k_e}\, f_d \;+\; \frac{k_d}{k_d + k_e}\, y_d, \tag{3.27a}$$

$$f_e \;=\; \frac{k_e}{k_d + k_e}\, f_d \;+\; \frac{k_e k_d}{k_d + k_e}\, y_d. \tag{3.27b}$$

Accordingly, the cost function (3.25) becomes

$$J = c\left(\frac{f_d - k_e y_d}{k_d + k_e}\right)^2 \;+\; \left(\frac{k_d(f_d - k_e y_d)}{k_d + k_e}\right)^2. \tag{3.28}$$

By minimizing this cost function with respect to k_d, i.e. $\frac{\mathrm{d}J}{\mathrm{d}k_d} = 0$, the optimal solution is

$$k_d \;=\; \frac{c}{k_e}. \tag{3.29}$$

The optimal solution confirms the duality principle, where the active stiffness k_d is inverse proportional to the environment stiffness. It is obvious that the duality of both $k_e \approx 0$ and $k_e = \infty$ is fulfilled. For intermediate values of stiffness [30] defines a moderate stiffness $k_{mod} := \sqrt{c}$ to govern the relation between the active and environment stiffness. It means that when the environment is stiffer than the moderate stiffness, i.e. $k_e > k_{mod}$ the active stiffness becomes softer than for the moderate stiffness $k_d < k_{mod}$ and vice versa. While the trivial solution would be $f_d = k_e\, y_d$. For

the trivial solution, the cost function is always equal to zero, since the relationship between desired position y_d and desired force f_d is consistent all the time.

In Figure 3.4 (which is a modified version of figure in [30]) the compromise between position and force in optimal impedance control and the influence of the moderate stiffness is investigated. Here, the loci of the optimal values of (y, f_e) are represented as solid lines for different moderate environments, i.e. $k_{mod1} < k_{mod2} < k_{mod3}$ with k_e ranging from zero to infinity. The desired position and force are set as $y_d = 2\,[m]$ and $f_d = 60,000\,[N]$. To show the relation between the moderate the static environment stiffness, two dot-dashed lines are drawn for two values of $k_e = 8,000$ and $80,000\,[N/m]$. The variable impedance controller behaves like a force controller and a position controller for stiff and soft environments, respectively. As can be seen, the solid lines for moderate stiffness, all of them pass through points $(0, f_d)$, $(y_d, 0)$, and (y_d, f_d). Actually, the last three points lie on the ellipses defined as follows

$$\frac{\left(f_e - \frac{f_d}{2}\right)^2}{\left(\frac{f_d^2 + k_{mod}^2 y_d^2}{4}\right)} + \frac{\left(y - \frac{y_d}{2}\right)^2}{\left(\frac{f_d^2 + k_{mod}^2 y_d^2}{4k_{mod}^2}\right)} = 1 \tag{3.30}$$

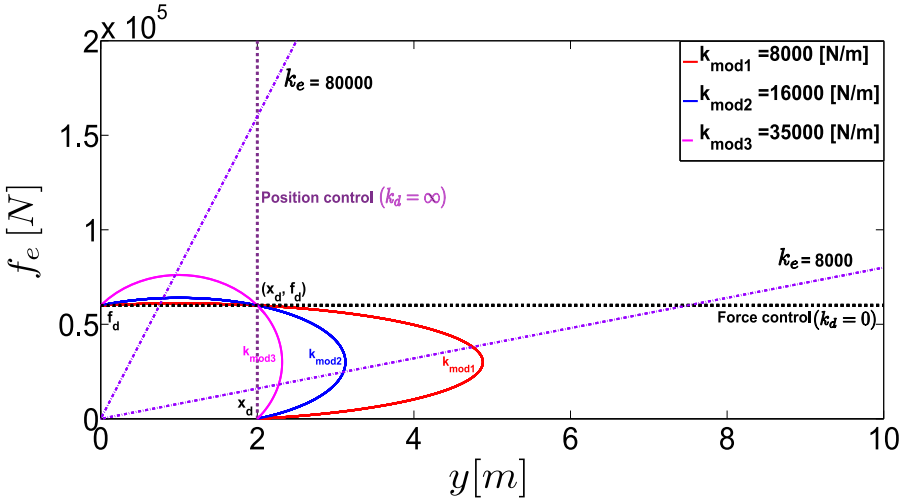


Figure 3.4: Geometric explanation of static-impedance control. [30]

For lower values of k_{mod} the controller is shifted towards force control, since the range of all degrees of the environment stiffness above moderate stiffness is considered as stiff, hence for lower moderate stiffness the stiff range will be wider [30].

To overcome the disadvantages of optimization-based impedance, i.e. noise of (derivative) force signal and trade-off between position and force errors, we use the desired admittance dynamics as a constraint in the optimal control problem instead of including the force error in the cost function (see Chapter 4). This is analogous to

overcome the interference of force feedback and motion control in impedance control as explained in the previous section.

3.6.1 Predictive Force Feedback Control

As shown in Chapter 2, model predictive control minimizes a cost function over a prediction horizon, while using the system dynamics model to predict the future of the output-trajectory. At each sampling instant, the predicted output-trajectory is calculated using the current initial state of the dynamic system and the (sub)-optimal input sequence, which is predicted over a control horizon. In the previous section, we classified the optimization-based force feedback approaches into two classes, similarly, the model predictive force feedback control can be classified into these two classes.
For the first case, i.e. joint impedance level, Model Predictive Impedance Control is used as a general method by [120] for modeling the control system of different human movements. There, the features of this method for modeling different movements is shown, e.g. elbow joint tracking of a periodic or a non-periodic trajectory. While, for the second class, i.e. Cartesian impedance level, in [91] a nonlinear model predictive control scheme based-on neural networks maintains a stable grasp during Hold-state of the prosthetic device. There, Neural networks are used to identify the dynamics model of each finger of the prosthetic hand. Then, this model is used in a predictive control scheme to predict the required grasping force to be exerted by the prosthetic hand. Where, a nonlinear model predictive control scheme is selected based on its ability to tackle challenges in nonlinearities produced by the motor dead bands, gear ratios, friction. In [65] a single-time-step model predictive controller is used to control the interaction between the robot and the environment. This controller depends on a quasi-static mechanical interaction model. This controller drives the robot to the target locations while keeping contact forces low. In this scheme, the robot is equipped with tactile sensors along the whole-arm and compliant joints. The same approach is extended to a multi-time-step MPC formulation in [71].

Normally, the classical approaches of impedance control compute the virtual reference position in the constrained direction to control the force indirectly. Hence, a linear/nonlinear model of the environment is crucial to compute this virtual position. For simplicity, the environment model is considered as a linear model. However, for soft environments which typically have a nonlinear behavior, nonlinear models are required to preserve its characteristics. In [12] an approach to derive the virtual trajectory using model predictive control is proposed.

This approach is designed by combining both the classical impedance controller and a model predictive controller for the force. In this force control strategy, the predictive controller produces the virtual position and velocity references in the

constrained direction, which correspond to a desired contact-force [12]. In this control strategy, the environment and impedance models are considered to be linear [12], the reason is, an analytical solution can be obtained by solving an unconstrained optimization with less computation time. However, for non-rigid environments, commonly, the environment deforms nonlinearly. Therefore, a nonlinear model of the environment needs to be considered. Hence, the constrained optimization control problem solved by the MPC scheme is non-convex. In [13] this problem is solved using a discrete search techniques, i.e. branch-and-bound ($B\&B$) algorithm. There, a tradeoff between the number of discrete actions and the performance due to the discretization by the $B\&B$ algorithm is achieved by using a fuzzy scaling machine. This machine is based on a fuzzy criterion to produce an adaptive set of discrete alternatives. Figure 3.5 shows the concept of predictive impedance control. In this approach, the cost function (3.31) to be minimized by MPC consists of the force error.

$$\min\ J(y_r) = \sum_{i=1}^{N} \left(f_d(k+i) - \hat{f}_e(k+i) \right)^2, \tag{3.31}$$

where f_d represents the desired force, and $\hat{f}_e$ is the predicted contact-force.

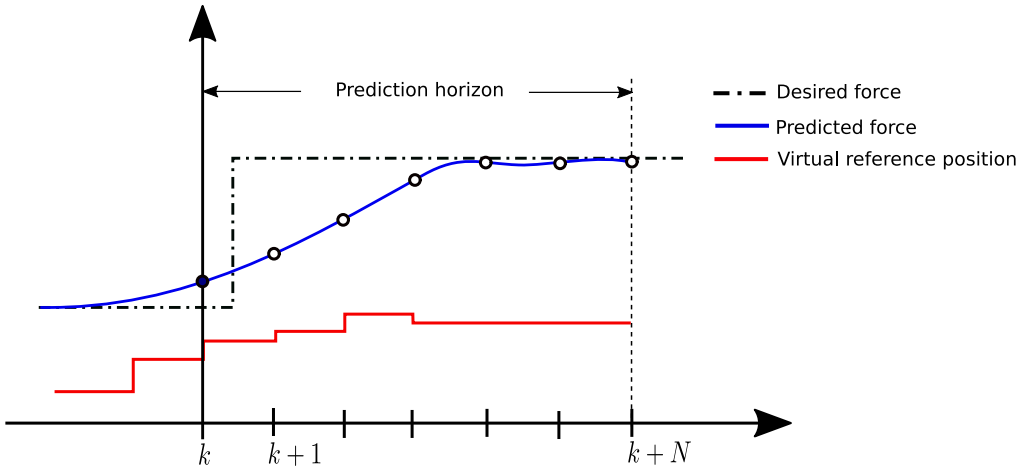


Figure 3.5: Prediction of virtual position using force error.

The MPC controller minimizes the cost function (3.31), to obtain the sequence of predicted virtual refernce positions $(\hat{y}_r(k), \hat{y}_r(k+1), \ldots, \hat{y}_r(k+N-1))$. Then, the new optimal virtual references within the prediction horizon lead to a new current desired trajectory from the impedance control law (3.16).

Summary

In this chapter, we gave a brief overview on force feedback control. We have outlined active and passive compliance-based control methods, in particular, *Impedance* and

Admittance force control. Furthermore, we described the issue of environment identification and modeling. In addition, the way of choosing the admittance parameters based-on the duality principle is explained. Then, we reviewed optimization-based force control approaches, with an emphasis on model predictive control schemes. In the next chapter, we will outline the main contribution of this thesis designing a unified approach to solve the path-following and force control problems simultaneously.

4 A Unified Approach for Path-following and Force-feedback

In this chapter, we outline a unified approach to combine path-following and force control problems using predictive control. First, we review briefly the existing approach to combine path-following and force control based on feedback linearization, introduced in [46]. Then, we design a novel predictive control framework to solve these problems in a unified way. To this end, we define an admittance dynamics to generate a reference trajectory proportional to the desired force. Afterwards, we update a given path using the output of the admittance dynamics. In short, we rewrite the predictive path-following optimization problem by including the admittance dynamics and the updated path.

4.1 Combined Path-following and Force Control Based-on Feedback Linearization

In this section, we describe the approach proposed in [46] to simultaneously, but independently, control the forces transversal and tangential to the path. This is achieved by decomposing a nonlinear dynamic system into transversal and tangential linear subsystems using feedback linearization. This allows to control the contact-force along these directions. Assuming that the Assumptions 2.1, 2.3, 2.6 and the following assumption hold for the path $\mathcal{P}$ (2.22)

Assumption 4.1
There is a smooth function $\mathcal{Y}(\cdot) : \mathbb{R}^{n_y} \to \mathbb{R}$, where the path $\mathcal{P}$ is a zero-manifold of $\mathcal{Y}(\cdot)$.

The dynamic system is defined as follows

$$\dot{x}(t) = \begin{bmatrix} x_v \\ f(x) + g(x)u - f_e \end{bmatrix}, \tag{4.1a}$$

$$y(t) = h(x_p), \tag{4.1b}$$

where the system states are defined as $x = [x_p,\, x_v]^T$, with the position x_p and the velocity x_v. $u(t)$ and $y(t)$ are the input and output of the system, respectively, while f_e represents the external force. Moving towards and following the path $\mathcal{P}$ can be

achieved by stabilizing the sub-manifold $\mathcal{M}_{\mathcal{Y}} := \{y \in \mathbb{R}^n \mid \mathcal{Y} \circ h(x_p) = 0\}$. Since $\mathcal{M}_{\mathcal{Y}}$ is not invariant [56, 89], therefore it is only possible to stabilize the maximum controlled invariant subset $\mathcal{M}^*_{\mathcal{Y}}$ of $\mathcal{M}_{\mathcal{Y}}$. Where, $\mathcal{M}^*_{\mathcal{Y}}$ includes the states for which there exists an admissible input such that the output stays on $\mathcal{P}$ for all times. The set $\mathcal{M}^*_{\mathcal{Y}}$ can be used to achieve the path followability of path following problem, see Definition 2.3 for sufficient conditions of path followability.

Feedback linearization is used to decompose and linearize a nonlinear system by mapping it into new coordinates using a transformation $\mathcal{T} : y \to (\eta, \xi)$, defined in a neighborhood of $\mathcal{P}$. Consequently, the nonlinear system (4.1) is transformed and decomposed into $\xi-$transversal (normal) and $\eta-$tangential subsystems with respect to $\mathcal{P}$.

Let B_r be the set of points around the path $\mathcal{P}$ within a distance r such that

$$B_r = \Big\{(y, p) \mid y \in \mathbb{R}^2 \wedge p(\theta) \in \mathcal{P} \mid \|y - p(\theta)\| \leq r\Big\}. \tag{4.2}$$

Here, we consider the path as a closed curve, i.e.

$$\theta \in [\theta_0,\, \theta_f]\,, \quad p(\theta + \theta_f) = p(\theta).$$

First, the tangential coordinate is defined by using the equation (2.28) to map any point $y \in B_r$ into a corresponding value of path-parameter $\theta^* \in [\theta_0, \theta_f)$, where θ^* is defined as follows

$$\overline{\Theta}(y) := \theta^*(t) = \underset{\theta \in [\theta_0, \theta_f)}{\operatorname{argmin}} \|y(t) - p(\theta(t))\|\,. \tag{4.3}$$

Equation (4.3) can have more than one solution, however, we can pick one of them. Here, $\overline{\Theta}(y)$ refers to a path-parameter corresponding to the point on the path, which is nearest to the point $y \in B_r$. To derive the tangential state η_1, which represents the traveled distance of θ from θ_0 to θ^* along the path $\mathcal{P}$. To this end, we define $p'(\theta) = \frac{\partial p(\theta(t))}{\partial \theta}$, then the tangential coordinate along the path is

$$\eta_1 = \boldsymbol{tang}(y) := \int_{\theta_0}^{\theta^*} \|p'(s)\|\, ds\Big|_{\theta^* = \overline{\Theta}(y)}\,. \tag{4.4}$$

Hence, the *tangential unit-vector* on the path corresponding to η_1 can be found as follows:

$$\hat{V}_\eta(\theta^*) = \frac{p'(\theta^*)}{\|p'(\theta^*)\|}. \tag{4.5}$$

Accordingly, the *transversal (normal) unit-vector* $\hat{V}_\xi(\theta^*)$ can be derived by rotating the vector $\hat{V}_\eta(\theta^*)$ 90° counter-clockwise such that

$$\hat{V}_\xi(\theta^*) = \mathbf{R}(\pi/2)\hat{V}_\eta(\theta^*), \tag{4.6}$$

where $\mathbf{R}(\cdot)$ is a rotation matrix.

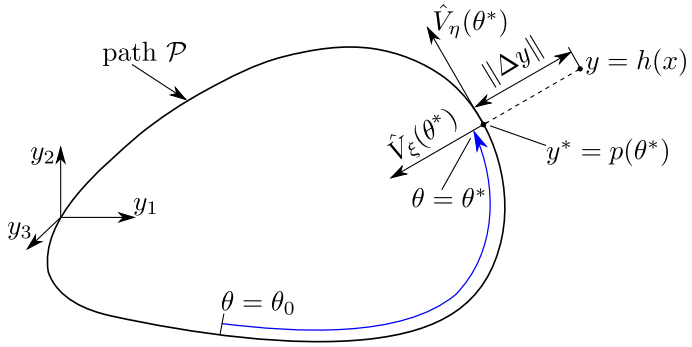


Figure 4.1: Path description [46]

Then, the current state of the output, i.e. $y = h(x_p) \in \mathbb{R}^{n_y}$, can be represented by shifting the point on the parametrized-path corresponding to θ^*, i.e. $p(\theta^*)$ by the distance $\Delta y = h(x_p) - p(\theta^*)$ parallel to the unit-vector $\hat{V}_\xi(\theta^*)$, that is

$$y = p(\theta^*) + \hat{V}_\xi(\theta^*)\,\Delta y, \tag{4.7}$$

where,

$$\Delta y = \boldsymbol{trans}(y) = \hat{V}_\xi(\theta^*)^T(y - p(\theta^*)), \tag{4.8}$$

represents the shortest distance of the output state from the curve $\mathcal{P}$ along the unit-vector $\hat{V}_\xi(\theta^*)$. Hence, Δy is nothing else but the first transversal state ξ_1. To complete the coordinate transformation, the time derivatives of η_1 and ξ_1 are required. Despite of that θ^* form (4.3) can only be computed numerically, however, the time derivative of θ^* can be calculated analytically from (4.3) using the necessary condition of optimality [46]. According to the necessary condition of optimality of (4.3),

$$(y - p(\theta^*))^T p'(\theta^*) = 0. \tag{4.9}$$

By taking the time derivative of (4.9) as follows

$$(\dot{y} - p'(\theta^*)\dot{\theta}^*)^T p'(\theta^*) + (y - p(\theta^*))^T p''(\theta^*)\dot{\theta}^* = 0,$$

then rewrite (4.6) and (4.8) to get $(y - p(\theta^*)) = R(\pi/2)V_\eta(\theta^*)\Delta y$ and take the time derivative of (4.4) to obtain

$$\dot{\eta}_1 = \|p'(\theta^*)\|\,\dot{\theta}^* = \frac{1}{1 - \vartheta'\Delta y}\,\frac{(p'(\theta^*))^T}{\|p'(\theta^*)\|}\,\dot{y}, \tag{4.10}$$

where

$$\vartheta' = \frac{(R(\pi/2)\,p'(\theta^*))^T\,p''(\theta^*)}{\|p'(\theta^*)\|^3}.$$

Similarly, the time derivative of ξ_1 is derived as follows

$$\dot{\xi}_1 = \xi_2 = \frac{(R(\pi/2)\, p'(\theta^*))^T}{\|p'(\theta^*)\|}\ \dot{y}. \tag{4.11}$$

Hence, the coordinate transformation $\mathcal{T}$ becomes:

$$\mathcal{T} = \begin{bmatrix} \eta_1 \\ \eta_2 \\ \xi_1 \\ \xi_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{tang} \circ h(x_p) \\ \mathrm{d}\eta_1\, \mathrm{d}h\, \dot{x} \\ \boldsymbol{trans} \circ h(x_p) \\ \mathrm{d}\xi_1\, \mathrm{d}h\, \dot{x} \end{bmatrix}. \tag{4.12}$$

The new virtual output following [46, 56] is chosen as

$$\tilde{\mathbf{y}} = \begin{bmatrix} \boldsymbol{tang} \circ h(x_p) \\ \boldsymbol{trans} \circ h(x_p) \end{bmatrix} = \begin{bmatrix} \Lambda_\eta(x_p) \\ \Lambda_\xi(x_p) \end{bmatrix}. \tag{4.13}$$

Then, the feedback linearization can be derived by taking the first and second time derivative of (4.13):

$$\dot{\tilde{\mathbf{y}}} = \begin{bmatrix} \Lambda'_\eta(x_p)\, x_v \\ \Lambda'_\xi(x_p)\, x_v \end{bmatrix}, \tag{4.14}$$

where $\Lambda'(x_p) = \frac{\partial \Lambda(x_p)}{\partial x_p}$ and

$$\ddot{\tilde{\mathbf{y}}} = \underbrace{\begin{bmatrix} \dot{\Lambda}'_\eta(x_p)\ x_v\ +\ \Lambda'_\eta(x_p)\, f(x) \\ \dot{\Lambda}'_\xi(x_p)\ x_v\ +\ \Lambda'_\xi(x_p)\, f(x) \end{bmatrix}}_{\mathcal{A}_1(x)} + \underbrace{\begin{bmatrix} \Lambda'_\eta(x_p)\, g(x) \\ \Lambda'_\xi(x_p)\, g(x) \end{bmatrix}}_{\mathcal{A}_2(x)}\ \mathbf{u}\ -\ \underbrace{\begin{bmatrix} \Lambda'_\eta(x_p) \\ \Lambda'_\xi(x_p) \end{bmatrix}}_{\mathcal{A}_3(x)}\ f_e. \tag{4.15}$$

By choosing $\ddot{\tilde{\mathbf{y}}}$ as

$$\ddot{\tilde{\mathbf{y}}} = \begin{bmatrix} \tilde{u}_\eta \\ \tilde{u}_\xi \end{bmatrix} = \tilde{\mathbf{u}}, \tag{4.16}$$

where $\tilde{\mathbf{u}}$ is the new input vector, the linearizing feedback is as follows

$$\mathbf{u} = \mathcal{A}_2^{-1}(x)\ [\, -\mathcal{A}_1(x)\ +\ \mathcal{A}_3(x)\ f_e\ +\ \tilde{\mathbf{u}}\,]\ |_{x=\mathcal{T}^{-1}(\eta,\,\xi)}\ . \tag{4.17}$$

By substituting (4.17) and (4.12) into (4.1), which gives linearized and decoupled tangential and transversal subsystems respectively as follows

$$\dot{\eta}_1 = \eta_2, \quad \dot{\eta}_2 = \tilde{u}_\eta, \qquad \dot{\xi}_1 = \xi_2, \quad \dot{\xi}_2 = \tilde{u}_\xi.$$

To control forces along tangent and transversal (normal) direction with respect to a given path, the contact-force f_e is projected along these directions, i.e.

$$\hat{f}_e = \begin{bmatrix} \hat{f}_{e_\eta} \\ \hat{f}_{e_\xi} \end{bmatrix} = \begin{bmatrix} \hat{V}_\eta^T(\theta^*)\, f_e \\ \hat{V}_\xi^T(\theta^*)\, f_e \end{bmatrix}, \tag{4.18}$$

and the position errors along these unit-vectors are defined as $e_\eta = y_\eta - y_\eta^d$ and $e_\xi = y_\xi - y_\xi^d$, where the y_η^d and y_ξ^d refer to the tangential and transversal desired state, respectively. The desired impedance (admittance) dynamic law is then given by

$$\underbrace{\begin{bmatrix} m_\eta & 0 \\ 0 & m_\xi \end{bmatrix}}_{M_d} \begin{bmatrix} \ddot{e}_\eta \\ \ddot{e}_\xi \end{bmatrix} + \underbrace{\begin{bmatrix} b_\eta & 0 \\ 0 & b_\xi \end{bmatrix}}_{B_d} \begin{bmatrix} \dot{e}_\eta \\ \dot{e}_\xi \end{bmatrix} + \underbrace{\begin{bmatrix} k_\eta & 0 \\ 0 & k_\xi \end{bmatrix}}_{K_d} \begin{bmatrix} e_\eta \\ e_\xi \end{bmatrix} = \hat{f}_e. \tag{4.19}$$

As one can see, in this approach the locally invertible transformation matrix is necessary to achieve linearization and decomposition of the nonlinear system along a given path. In the following, we describe our approach to solve the path-following and force control problem simultaneously in a model predictive scheme while considering constraints on states and input.

4.2 Predictive Path-Following Force-Feedback Control in The Output-space

Here, we describe the solution of the path-following and force control problems using nonlinear model predictive control. First, we define the admittance dynamics, which governs the interaction between the robot and the environment. Second, we explain the proposed method to update the followed path depending on the output of the desired admittance law. Based on that, we rewrite the optimal control problem (2.26) by including the admittance dynamics in the augmented dynamic system and using the new updated path. Consider a nonlinear system of the form

$$\dot{x}(t) = f(x(t), u(t)), \quad x(t_0) = x_0, \tag{4.20a}$$
$$y(t) = h(x_p(t)), \qquad y \in \mathbb{R}^{n_y} \tag{4.20b}$$
$$x \in \mathcal{X} \subseteq \mathbb{R}^{n_x}, \tag{4.20c}$$
$$u \in \mathcal{U} \subseteq \mathbb{R}^{n_u}, \tag{4.20d}$$

where $x(t), y(t), u(t)$ are defined as before. The sets (4.20c), and (4.20d) are constraints on state and input. The initial condition is denoted by x_0. The state trajectory starting from x_0 at time t_0 is defined by $x(\tau, u(\cdot)|x(t_0))$, where $u(\cdot) : [t_0, t_f] \to U$ and $\tau \in [t_0, t_f]$, t_f refers to a final time, which is not necessarily specified a priori.

The output trajectory is defined by $y(\tau|x_p(t_0)) = h(x_p(\tau, u(\cdot)|x(t_0)))$. The maps $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$ and $h : \mathbb{R}^{n_x} \to \mathbb{R}^{n_y}$ are sufficiently often continuously differentiable. For the sake of simplicity, we assume the system is square, i.e. $n_u = n_y$.

4.2.1 Extension Towards Admittance Force Control

In this section, we briefly mention the principle of admittance control. Then, we derive the admittance dynamics. Furthermore, we describe in detail the proposed approach to include force control in the predictive path-following controller. Basically, this approach can be summarized as, using the trajectory generated by the desired admittance dynamics to update the followed path.

4.2.1.1 Admittance Control

As we mentioned in Chapter 3, the principle of admittance control is to modify the position of the end-effector, such that the desired robot-environment interaction is achieved. The contact-force in a space is represented by a tangent, normal, and bi-normal direction with respect to a given path. We can extend the map (4.18) to include the vector that is bi-normal to the tangent and the normal vector (see Figure 4.2). The motivation behind this extension is that some robotics tasks require following a path in a plane while controlling the force in a direction perpendicular to this plane, e.g. writing on a board needs to follow a path in the board-plane while controlling the force in the normal direction on the board-plane. To this end, the bi-normal unit-vector is derived by preforming the cross-product of the tangent (4.5) and the normal (4.6) unit-vectors,

$$\hat{V}_\nu(\theta^*) = \hat{V}_\eta(\theta^*) \times \hat{V}_\xi(\theta^*), \tag{4.21}$$

where the bi-normal unit-vector is normal to the plane containing $\hat{V}_\eta(\theta^*)$ and $\hat{V}_\xi(\theta^*)$, i.e. $\hat{V}_\nu(\theta^*)$ is parallel to the y_3-axis in Figure 4.1. To define the third coordinate, the displacement along the bi-normal unit-vector $\hat{V}_\nu(\theta^*)$, which is identical to the projection of vector $(y - p(\theta^*))$ on the unit-vector $\hat{V}_\nu(\theta^*)$, so the first lateral coordinate along the unit-vector $\hat{V}_\nu(\theta^*)$ is chosen as

$$\nu_1 = \boldsymbol{lat}(y) = \hat{V}_\nu(\theta^*)^T (y - p(\theta^*)). \tag{4.22}$$

The forces along the tangent, transversal (normal), and lateral (bi-normal) direction with respect to a given path can be done by projecting the contact-force f_e along these directions

$$\hat{f}_e = \begin{bmatrix} \hat{f}_{e_\eta} \\ \hat{f}_{e_\xi} \\ \hat{f}_{e_\nu} \end{bmatrix} = \begin{bmatrix} \hat{V}_\eta^T(\theta^*)\, f_e \\ \hat{V}_\xi^T(\theta^*)\, f_e \\ \hat{V}_\nu^T(\theta^*)\, f_e \end{bmatrix} \tag{4.23}$$

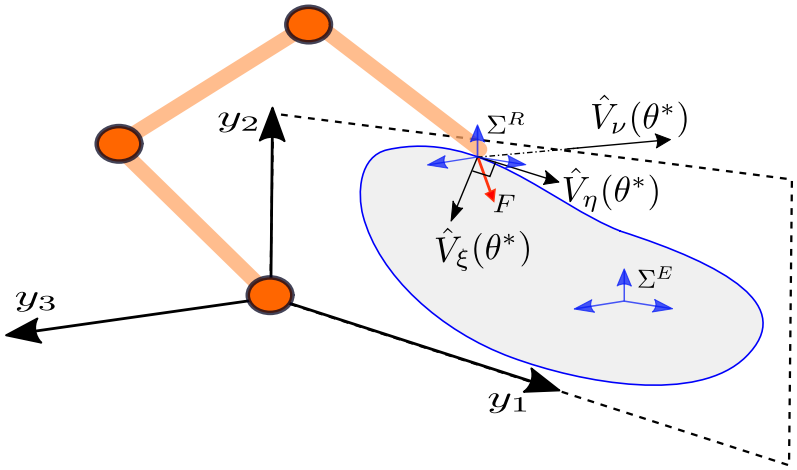


Figure 4.2: Force mapping

and the position errors along these unit-vectors are defined as $e_\eta = y_\eta - y_\eta^d$, $e_\xi = y_\xi - y_\xi^d$, and $e_\nu = y_\nu - y_\nu^d$, where the y_η^d, y_ξ^d and y_ν^d refer to the tangential, transversal, and lateral desired state, respectively. As common, the desired admittance is modeled as a second order mass-spring-damper system [108, 122]. The desired admittance dynamics can be written as

$$\underbrace{\begin{bmatrix} m_\eta & 0 & 0 \\ 0 & m_\xi & 0 \\ 0 & 0 & m_\nu \end{bmatrix}}_{M_d} \begin{bmatrix} \ddot{e}_\eta \\ \ddot{e}_\xi \\ \ddot{e}_\nu \end{bmatrix} + \underbrace{\begin{bmatrix} b_\eta & 0 & 0 \\ 0 & b_\xi & 0 \\ 0 & 0 & b_\nu \end{bmatrix}}_{B_d} \begin{bmatrix} \dot{e}_\eta \\ \dot{e}_\xi \\ \dot{e}_\nu \end{bmatrix} + \underbrace{\begin{bmatrix} k_\eta & 0 & 0 \\ 0 & k_\xi & 0 \\ 0 & 0 & k_\nu \end{bmatrix}}_{K_d} \begin{bmatrix} e_\eta \\ e_\xi \\ e_\nu \end{bmatrix} = \hat{f}_e, \tag{4.24}$$

where M_d, B_d, and K_d are diagonal $\mathbb{R}^{n_y \times n_y}$ positive definite matrices of the desired mass, damping, and stiffness gains defined by the user. To achieve force regulation, the desired force vector $F^d \in \mathbb{R}^{n_y}$ is defined as a constant. Hence, the admittance dynamics is driven by the force error $E_f = F^d - \hat{f}_e$. We write the admittance dynamics along one-dimension as

$$m_i\left(\ddot{y}_i(t) - \ddot{y}_i^d(t)\right) + b_i\left(\dot{y}_i(t) - \dot{y}_i^d(t)\right) + k_{di}\left(y_i(t) - y_i^d(t)\right) = e_{fi}(t), \quad i \in \{\eta, \xi, \nu\}, \tag{4.25}$$

where m, b, k_d, e_f are the diagonal elements of the matrices M_d, B_d, K_d, E_f, respectively. $y_i(t) \in \mathbb{R}^{n_y}$ is the current desired Cartesian position of the end-effector based on the admittance dynamics, and $y_i^d(t) \in \mathbb{R}^{n_y}$ represents the reference position trajectory -*virtual trajectory*- that depends on the desired contact force, environment position, and environment stiffness, as shown later. In addition, to achieve passivity of the system during interaction with the environment, the first and second time derivatives of the reference position y_i^d are set to zero [8, 24]. Hence, the admittance along the

one-dimensional Cartesian coordinate is reduced to

$$m_i\,\ddot{y}_i(t) \;+\; b_i\,\dot{y}_i(t) \;+\; k_{di}\,(y_i(t) - y_i^d(t)) \;=\; e_{fi}(t), \quad i \in \{\eta, \xi, \nu\}\,. \tag{4.26}$$

We follow the lines of [108] to study the convergence of the last equation to the steady-state. First, we write the environment model (3.9) with y_i as a function of e_{fi} remembering that the output function $h(x_p)$ maps the states to the output space as

$$y_i = \frac{1}{k_e}\hat{f}_{e_i} + y_{ei} = \frac{1}{k_e}\left[F_i^d - e_{fi}\right] + y_{ei}. \tag{4.27}$$

Combining (4.26) - (4.27) yields

$$m_i\ddot{e}_{fi} + b_i\dot{e}_{fi} + (k_{di} + k_e)e_{fi} = m_i\ddot{F}_i^d + b_i\dot{F}_i^d + k_{di}F_i^d - k_e k_{di}(y_i^d - y_{ei}). \tag{4.28}$$

As mentioned above, F_i^d is a constant, therefore, the steady-state force tracking error is inferred from (4.28) as

$$e_{fi}^{ss} = \frac{k}{k_{di} + k_e}\left[F_i^d + k_e(y_{ei} - y_i^d)\right] = k_{eq}\left[\frac{F_i^d}{k_e} + y_{ei} - y_i^d\right], \tag{4.29}$$

where $k_{eq} = \frac{k_{di}k_e}{k_{di}+k_e}$ is the equivalent stiffness of the environment and the desired admittance dynamics. To this end, careful investigation of (4.29) suggests that by setting the -virtual- reference trajectory y_i^d as

$$y_i^d = y_{ei} + \frac{F_i^d}{k_e} \tag{4.30}$$

yields

$$e_{fi}^{ss} = k_{eq}\left[\frac{F_i^d}{k_e} + y_{ei} - (y_{ei} + \frac{F_i^d}{k_e})\right] = 0. \tag{4.31}$$

That is, if both position and stiffness of the environment are known precisely, one can use (4.30) to calculate the reference trajectory y_i^d to apply the desired contact force F_i^d on the environment. However, this is not the case in practice, where the values of k_e and y_{ei} are not known accurately, so the desired force will not be achieved. There are many approaches to deal with robustness and parameter uncertainties of the admittance/impedance control, see, e.g., [67, 108]. In [108], the authors use two online schemes for generating the reference position y_i^d without needing a priori knowledge of the position and stiffness of the environment. While, in [67], the stiffness term of the desired impedance/admittance is equated to zero, hence the error force goes to zero for any k_e. In this formulation the reference trajectory in (4.25) is replaced by the environment position y_{ei} such that

$$m_i\,\ddot{y}_i(t) \;+\; b_i\,\dot{y}_i(t) \;+\; k_{di}\,e_i(t) \;=e_{fi}(t), \quad i \in \{\eta, \xi, \nu\}\,, \tag{4.32}$$

where $e_i = (y_i - y_{ei})$. In [67], the proposed scheme consists of a two-phase control algorithm; the first is free-motion control, i.e. before the interaction, the second, which starts when contact with the environment is established. During the free-motion phase, the force control law is reduced to

$$m_i\,\ddot{y}_i(t) + b_i\,\dot{y}_i(t) + k_{di}\,e_i(t) = F_i^d, \quad i \in \{\eta, \xi, \nu\}\,, \tag{4.33}$$

since the contact-force equals zero, i.e. $\hat{f}_{e_i} = 0$. Obviously, if we want to bring the robot to only make contact with the environment we have to set the desired force to zero and assuming that the exact environment position is given. However, in (4.33), the desired force specifies the force exerted on the environment by penetrating the environment with a depth $(y_i - y_{ei}) > 0$ proportional to the desired force. During the interaction phase, [67] proposes that the stiffness term of the desired admittance k_d is equated to zero, hence the force error goes to zero, i.e. $\hat{f}_{e_i} = F_i^d$ for any k_e. Then, the desired admittance becomes

$$m_i\,\ddot{y}_i(t) + b_i\,\dot{y}_i(t) - F_i^d + \hat{f}_{e_i} = 0, \quad i \in \{\eta, \xi, \nu\}\,. \tag{4.34}$$

By substituting $\hat{f}_{e_i} = k_e\,(y_i - y_{ei})$ in (4.34), the desired admittance becomes,

$$m_i\,\ddot{y}_i(t) + b_i\,\dot{y}_i(t) + k_e\,e_i(t) = F_i^d, \quad i \in \{\eta, \xi, \nu\}\,. \tag{4.35}$$

We denote this admittance as *zero-stiffness admittance* to distinguish it from the nominal admittance (4.26). The dynamic system (4.35) is asymptotically stable despite that k_e is not known accurately, and the tuning of gains m and b allows a smooth convergence [67]. The simulation and experimental validations in Chapter 5 show that the robustness and stability of this approach are fine and practice.

Remark 4.1 (Tuning parameters of the admittance dynamic)
As mentioned in Section 3.5, the desired impedance/admittance parameters are chosen such that the response is dual to the environment [9], e.g. if the environment is stiff, then the admittance dynamics is designed to be non-stiff. In general, for achieving fast and smooth convergence to the desired force, the other parameters of the admittance (i.e. mass and damping coefficient) are chosen such that the dynamic response is critically damped.

4.2.1.2 Updating the Path Using the Desired Admittance Dynamics

So far, we derived the desired admittance dynamics to govern the dynamic interaction of the robot with the environment. As mentioned before, in the admittance control scheme the desired position is updated according to the measured force using the admittance dynamics. In our case, we are using the predictive path-following controller

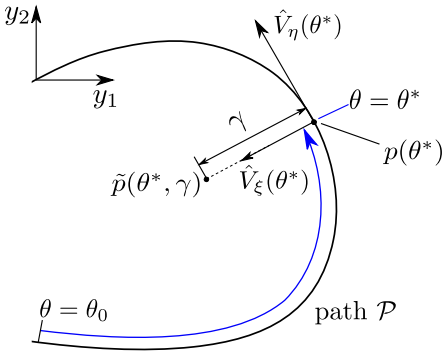


Figure 4.4: Illustration of scaling of the geometric path by the admittance dynamics.

as a position controller to keep the robot's end-effector moving along the desired path $\mathcal{P}$. Assuming perfect path-following, the force tracking is achieved by updating the path $\mathcal{P}$ using the trajectory generated by the admittance dynamics, see Figure 4.3.

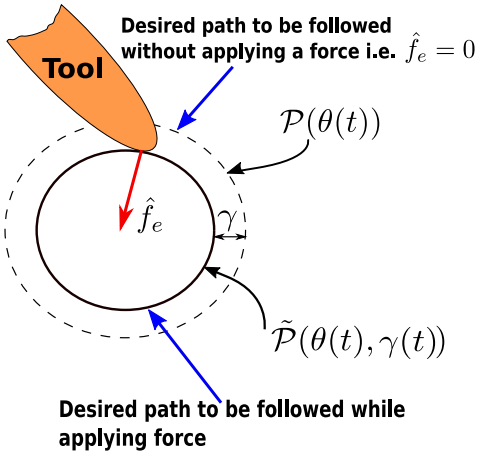


Figure 4.3: The robot's end-effector follows a circular path while applying a force.

In the following, we present an approach to update the geometric path based on the admittance dynamics. To this end, we present the following definition and assumptions.

Definition 4.1 (Shrunken/extended parametrized-path)
$\tilde{p} : [\theta_0, \theta_f] \rightarrow \tilde{\mathcal{P}}$ is called a geometric shrunken/extended parametrized-path such that each point on the path $p : [\theta_0, \theta_f] \rightarrow \mathcal{P}$ shifted along the normal vector of the tangent at this point to a specified value γ defined as follows

$$\forall \theta \in [\theta_0, \theta_f], \ \|\tilde{p}(\theta) - p(\theta)\| = \gamma, \tag{4.36}$$

holds.

Assumption 4.2 The parametrizations $p(\cdot)$ and $\tilde{p}(\cdot)$ are smooth and embedded submanifolds of $\mathbb{R}^2$.

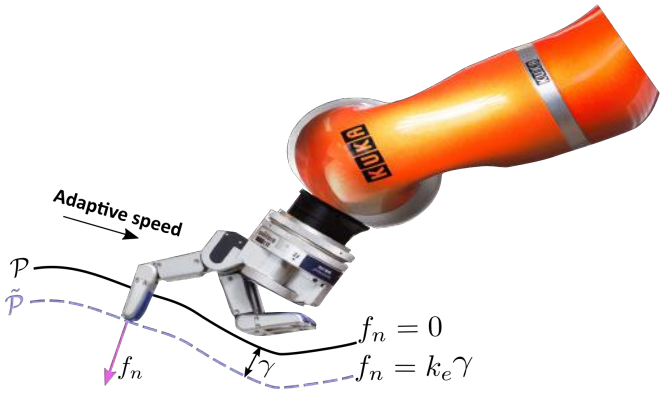


Figure 4.5: Following an updated geometric path.

Furthermore, we consider that the tangential (4.5) and the normal (4.6) unit-vectors are given. Here, we consider the path as a closed curve, i.e.

$$\theta \in [\theta_0,\, \theta_f]\,, \quad p(\theta + \theta_f) = p(\theta).$$

Now, to control the force along the transversal unit-vector, we use admittance dynamics along this direction as follows

$$m_d\,\ddot{\gamma} \,+\, b\,\dot{\gamma} + \,\hat{f}_{e_\xi} \,=\, F_\xi^d, \tag{4.37}$$

where $\gamma \in [0, \gamma_d]$ and $\gamma_d = F_\xi^d/k_e$, which represents the depth of penetration corresponding to the desired force. Furthermore, the desired force is assumed to fulfill the following assumption

Assumption 4.3 *The desired force F_ξ^d is a constant and chosen such that Assumption 4.2 holds.*

The state space representation of (4.37) can be written as

$$\dot{\boldsymbol{\gamma}} = \Pi(\boldsymbol{\gamma}, F_\xi^d) = \begin{bmatrix} \gamma_2 \\ 1/m_d\big(\,-\,b\,\gamma_2\,-\,\hat{f}_{e_\xi}\,\big) \end{bmatrix} + \begin{bmatrix} 0 \\ 1/m_d \end{bmatrix} F_\xi^d, \tag{4.38}$$

where $\boldsymbol{\gamma} = [\gamma_1,\, \gamma_2]^T \,=\, [\gamma,\, \dot{\gamma}]^T$. Hence, based on the Definition 4.1 we can scale the geometric path $\mathcal{P}$ by shifting each point on the path along the normal unit-vector (4.6) by γ (see Figure 4.5) as follows

$$\tilde{p}(\theta, \gamma) : [\theta_0, \theta_f] \times [0, \gamma_d] \to \tilde{\mathcal{P}} := p(\theta) + \hat{n}(\theta)\,\gamma. \tag{4.39}$$

The first argument of $\tilde{p}(\theta, \gamma)$ controls the evolution along the path (i.e. path-following), while the second one changes the path's coordinates (i.e. force control).

Problem 4.1 (Path Following Force Regulation Problem)
Given a system (4.20)*, and a path* $\tilde{\mathcal{P}}$ (4.39)*, design a controller that drives the system output* (4.20b) *to fulfill:*

- ***Convergence to path***: *The system output* $y = h(x_p)$ *moves toward the updated-path* $\tilde{\mathcal{P}}$ *such that*

$$\lim_{t\to\infty} \|h(x_p(t)) - \tilde{p}(\theta, \gamma)))\| = 0.$$

- ***Convergence on path***: *The system output keeps going along the path monotonically in the increasing direction of* $\theta(t)$*, i.e.* $\dot{\theta}(t) \geq 0$ *holds and* $\lim_{t\to\infty} \theta(t) = \theta_f$*.*

- ***Feasibility***: *The constraints on the states* $x(t) \in \mathcal{X}$ *and on inputs the* $u(t) \in \mathcal{U}$ *are satisfied for all time* $t \geq t_0$*.* ■

To solve Problem 4.1, we reformulate the augmented system (2.24) to include the desired admittance dynamics (4.38). That can be done by adding (4.38) to (2.24a),

$$\begin{bmatrix} \dot{x} \\ \dot{\hat{x}} \\ \dot{\mathbf{y}}_i \end{bmatrix} = \begin{bmatrix} f(x,u) \\ g(\hat{x},\hat{u}) \\ \Pi(\boldsymbol{\gamma}, F_\xi^d) \end{bmatrix} \qquad \begin{bmatrix} x(t_0) \\ \hat{x}(t_0) \\ \boldsymbol{\gamma}(t_0) \end{bmatrix} = \begin{bmatrix} x_0 \\ \hat{x}_0 \\ \boldsymbol{\gamma}_0 \end{bmatrix} \tag{4.40a}$$

$$\begin{bmatrix} \tilde{e} \\ \theta \end{bmatrix} = \begin{bmatrix} h(x_p) - \tilde{p}(\theta(t), \gamma(t)) \\ \hat{x}_1 \end{bmatrix}. \tag{4.40b}$$

Where $\tilde{e}$ is the new path-following error based on the updated path. We rewrite (2.26) using (4.40) instead of (2.24), then, the new optimization problem is written as

$$\min_{u,v} \int_{t_i}^{t_i+N} L\big(\bar{y} - \tilde{p}(\theta,\gamma), \bar{\theta} - \theta_f, \bar{u}, \bar{v}\big) d\tau + E\big(\bar{x}(t_i+N), \bar{\theta}(t_i+N)\big) \tag{4.41a}$$

$$\text{s.t.} \quad \dot{\bar{x}}(\tau) = f(\bar{x}(\tau), \bar{u}(\tau)), \quad \bar{x}(t_k) = x(t_k) \tag{4.41b}$$

$$\dot{\bar{\hat{x}}}(\tau) = g(\bar{\hat{x}}(\tau), \bar{\hat{u}}(\tau)), \quad \bar{\hat{x}}(t_k) = \hat{x}(t_k) \tag{4.41c}$$

$$\dot{\bar{\boldsymbol{\gamma}}} = \Pi(\bar{\boldsymbol{\gamma}}, F_\xi^d), \quad \bar{\boldsymbol{\gamma}}(t_k) = \boldsymbol{\gamma}(t_k) \tag{4.41d}$$

$$\bar{\tilde{e}}(\tau) = h(\bar{x}_p(\tau)) - \tilde{p}(\bar{\hat{x}}_1(\tau), \bar{\gamma}_1(\tau)) \tag{4.41e}$$

$$\bar{x}(\tau) \in \mathcal{X}, \ \bar{u}(\tau) \in \mathcal{U} \tag{4.41f}$$

$$\bar{\hat{x}}(\tau) \in \hat{\mathcal{X}}, \ \bar{\hat{u}}(\tau) \in \hat{\mathcal{U}}. \tag{4.41g}$$

In (4.41) the desired admittance dynamics (4.41d) is included as a constraint. The evolution of the desired admittance $\gamma(t)$ appears as a second argument of the parametrized path $\tilde{p}(\theta, \gamma)$. Hence, by following the updated path $\tilde{p}(\theta, \gamma)$, the desired force on the environment is applied.

Summary

In this chapter, we outlined the main contribution of this thesis. First, we showed the existed approach for combined path-following and force-feedback control using feedback linearization. Then, we explained our approach, which is exploiting the model predictive control scheme to solve the path-following and the force feedback problems simultaneously within an one optimization problem. In the next chapter, we present simulations and experimental work for verifying the proposed approach.

5 Validation

In this chapter, we give a description of simulation and experimental studies, which are carried out to verify the proposed approach. Before performing the experimental work, the proposed approach is tested via simulation. The simulations were performed in two steps. The first step was performed assuming full-state measurement without feedback-delay. In contrast, the second step assumed that the velocity state is not measured, so it has to be estimated. Furthermore, we assumed that there exists a feedback-delay for the measurements. Here, the second step emulates the real experimental setup. The comparison between the two simulation steps gives us the ability to tune and safely test the proposed controller before connecting it to the real robotic system.

5.1 Simulation Experiment

As we mentioned above, we have tackled the simulation verification via two steps. First, we tested the proposed algorithm in MATLAB assuming full-state measurement using the identified model [14] of the robot used in the real experiment, i.e. KUKA LWR IV robot arm. Second, we used an open-source toolbox providing emulation of the robot arm and its communication system [14]. Moreover, the velocity state is estimated and a delay in force sensor measurement is considered. To achieve this, we derive the dynamic model and the kinematics transformation of the robot. Afterwards, we sketch the predictive path-following problem (2.26) and the predictive path-following with force feedback problem (4.41) for the robot arm.

5.1.1 Modeling of Robot Dynamics and Desired Admittance

To keep the formulation simple, we just consider two degrees of freedom of the KUKA LWR IV robot arm, i.e. joint 2 and 4 are actuated while the others are kept fixed, (see Figure 5.1). In general, a model of a robotic system can be written in joint space as follows (see Chapter 3),

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) + \tau_F(\dot{q}) = \tau_e - J_a^T(q)\hat{F}. \tag{5.1}$$

Here $q = [q_2,\, q_4]^T$ is the vector of joint angular positions of joints 2 and 4, and their time derivatives $\dot{q}$ and $\ddot{q}$ represent the angular velocity and acceleration, respectively. The vector $\tau_e = [\tau_{e2}, \tau_{e4}]$ refers to the applied torques on the corresponding joints.

The inertia matrix $M(q)$ is a symmetric positive definite matrix; $C(q, \dot{q})$ stands for the Coriolis and centrifugal effects; $g(q)$ represents the gravitational torque; and $\tau_F(\dot{q})$ is the friction torque in the joints, while J_a is the Jacobian matrix, which maps the Cartesian external force $\hat{F}$ into the joint-space (see Section 3.2.1).
The parameters of (5.1) are taken from the open-source toolbox [14], and the kinematics parameters are provided in Table C.1 in the Appendix C. To include the model (5.1) in the controller framework, we formulate (5.1) in the state space, setting $x_1 = q$, $x_2 = \dot{q}$, and $u = \tau_e$. This yields,

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} x_2 \\ M(x_1)^{-1} \left(u - C(x_1,\, x_2)x_2 - g(x_1) - \tau_F(x2) - J_a^T(q)\hat{F} \right) \end{bmatrix} \tag{5.2a}$$

$$y = T_{ca}(x_1), \tag{5.2b}$$

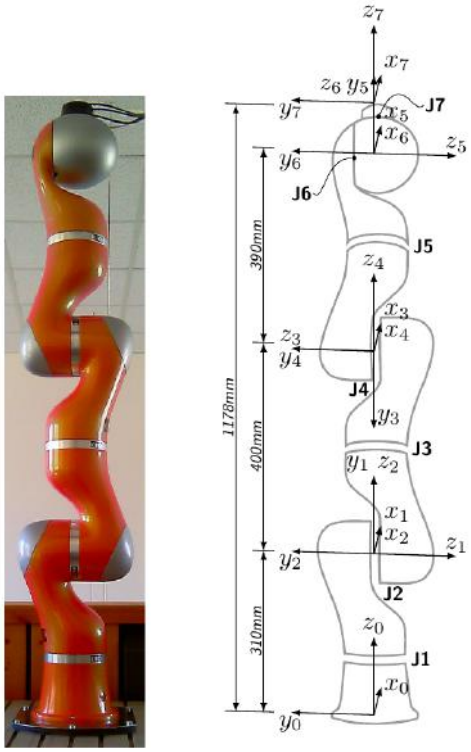


Figure 5.1: Joints of KUKA LWR IV

where the output (5.2b) maps the joint positions of the robot into Cartesian space using the forward kinematic transformation $T_{ca} : \mathbb{R}^2 \to \mathbb{R}^2$ defined by

$$T_{ca}(x_1) = \begin{bmatrix} l_1 \cos(q_2) \;+\; l_2 \cos(q_1 - q_2) \\ l_1 \sin(q_2) \;+\; l_2 \sin(q_1 - q_2) \end{bmatrix}, \tag{5.3}$$

where $l_i, i \in \{1, 2\}$ represent the lengths of the robot's links. To engage the path following problem (2.26), we construct the augmented model (4.40) using the robot model (5.2) and the timing law dynamics (2.15) designed as a second order integrator chain.
The cost function is identical for both path-following (2.26) and path-following with force feedback (4.41) problems, except in (4.41) the error term is $\tilde{e}$. So, the cost function is written as follows

$$L(e, \theta, u, \hat{u}) = \left\| (e\,,\, \theta - \theta_f)^T \right\|_W^2 + \left\| (u\,,\, \hat{u})^T \right\|_R^2, \tag{5.4}$$

where the weighting matrices W and R are symmetric and positive definite. In cost function (5.4), the first term penalizes the output error e to achieve the convergence to the path, while by penalizing $(\theta - \theta_f)$ the convergence on the path is achieved. In the second term of (5.4), the system input and the virtual input are penalized to ensure smooth convergence.
To verify our approach, we have used simple paths, e.g. sine- and circular-path. The mathematical representation of these parametrized paths are as follows

$$\text{Sine-path} \begin{cases} y_1 = y_{0_1} + a\, \sin(\theta), \\ y_2 = y_{0_2}\,(1 - \theta/2\pi) + y_{f_2}\, \theta/2\pi, \end{cases} \quad \theta \in [-2\pi,\, 0], \tag{5.5a}$$

$$\text{Circular-path} \begin{cases} y_1 = y_{1c} + r_c\, \sin(\theta), \\ y_2 = y_{2c} + r_c\, \cos(\theta), \end{cases} \quad \theta \in [-2\pi,\, 0]. \tag{5.5b}$$

where y_1 and y_2 are the Cartesian coordinates; $(y_{0_1},\, y_{0_2})$ and $(y_{f_1},\, y_{f_2})$ are the initial and final points of the path, respectively, and a is a constant. While $(y_{1c},\, y_{2c})$ and r_c are the center and the radius of the circle, respectively.

5.1.2 Nominal Simulations

In this step, the proposed approach is tested on an industrial robot arm, specifically, we used the identified dynamic model of the robot used in the real experiment, namely, the KUKA LWR IV robot arm from [14] with the parameters provided therein.

The tasks of the robot are, first to follow a desired path, i.e. Problem 2.2, and second to follow a path while applying a desired force on the environment, i.e. Problem 4.1. The optimal control problems (2.26) and (4.41) are solved using the *ACADO Toolkit* [59], which is a software environment and algorithm collection for automatic control and dynamic optimization.

As we discussed in Chapter 2, the computation time of the NMPC problem is the lower bound on the sampling time (2.42) and typically a small sampling time

is desirable. To achieve fast computation times, we used the so-called *ACADO Code Generation tool* [59, 60, 123], which exports highly efficient C-code for solving NMPC problem via a real-time iteration algorithm with Gauss-Newton Hessian approximation [123]. To this end, the continuous ODEs are discritized using shooting methods. Then the resulting sparse large quadratic program is passed to the qpOASES solver [40], which is a dense linear algebra quadratic program solver using an active set method. However, to employ this solution strategy the optimal control problem has to be written in a specific form, namely

$$\min_{\substack{x_0,\cdots,x_N\\ u_0,\cdots,u_N}} \sum_{k=0}^{N-1} \left\|L(x_k,u_k)-y_k^{ref}\right\|_{W_k}^2 + \left\|L_N(x_N)-y_N^{ref}\right\|_{W_N}^2 \tag{5.6a}$$

$$\text{s.t.} \quad x_0 = \tilde{x}_0 \tag{5.6b}$$

$$x_{k+1} = f(x_k,u_k,s_k), \ \text{for}\, k=0,\cdots,N-1 \tag{5.6c}$$

$$\underline{x}_k \le x_k \le \overline{x}_k, \ \text{for}\, k=0,\cdots,N \tag{5.6d}$$

$$\underline{u}_k \le u_k \le \overline{u}_k, \ \text{for}\, k=0,\cdots,N-1 \tag{5.6e}$$

$$\underline{r}_k \le r_k(x_k,u_k) \le \overline{r}_k, \ \text{for}\, k=0,\cdots,N-1 \tag{5.6f}$$

$$\underline{r}_N \le r_N(x_n) \le \overline{r}_N. \tag{5.6g}$$

Here, $x \in \mathbb{R}^{n_x}$ represents the state, $u \in \mathbb{R}^m$ is the control input, $s \in \mathbb{R}^{n_s}$ denotes the algebraic variable, and $\tilde{x}_0$ denotes the current state measurement. The reference functions in (5.6a) are denoted with $L \in \mathbb{R}^{n_l}$ and $L_N \in \mathbb{R}^{n_l,N}$, where N is the prediction horizon, and the weighting matrices are called $W_k \in \mathbb{R}^{n_l \times n_l}$ and $W_N \in \mathbb{R}^{n_l,N \times n_l,N}$. $y_k^{ref} \in \mathbb{R}^{n_l}$ and $y_N^{ref} \in \mathbb{R}^{n_l,N}$ represent time-varying references. (5.6e) and (5.6d), with $\underline{u} \le \overline{u}$ and $\underline{x} \le \overline{x}$, are constraints on control inputs and states, respectively. Equations (5.6f) and (5.6g) are the new representations of the path and terminal region, respectively, with $r_k \in \mathbb{R}^{n_r,k}$ and $r_N \in \mathbb{R}^{n_r,N}$ being some constraint functions. The function f represents the discretization of the ordinary differential model.

After reformulating (2.26) and (4.41) in the form (5.6), the code-generation tool is automatically generating a complete real-time iteration algorithm [60, 72] with optimized C-code of specified dimensions and static memory requirements. In addition, it chooses constant step-sizes, which ensures a deterministic run-time of the integration [59].

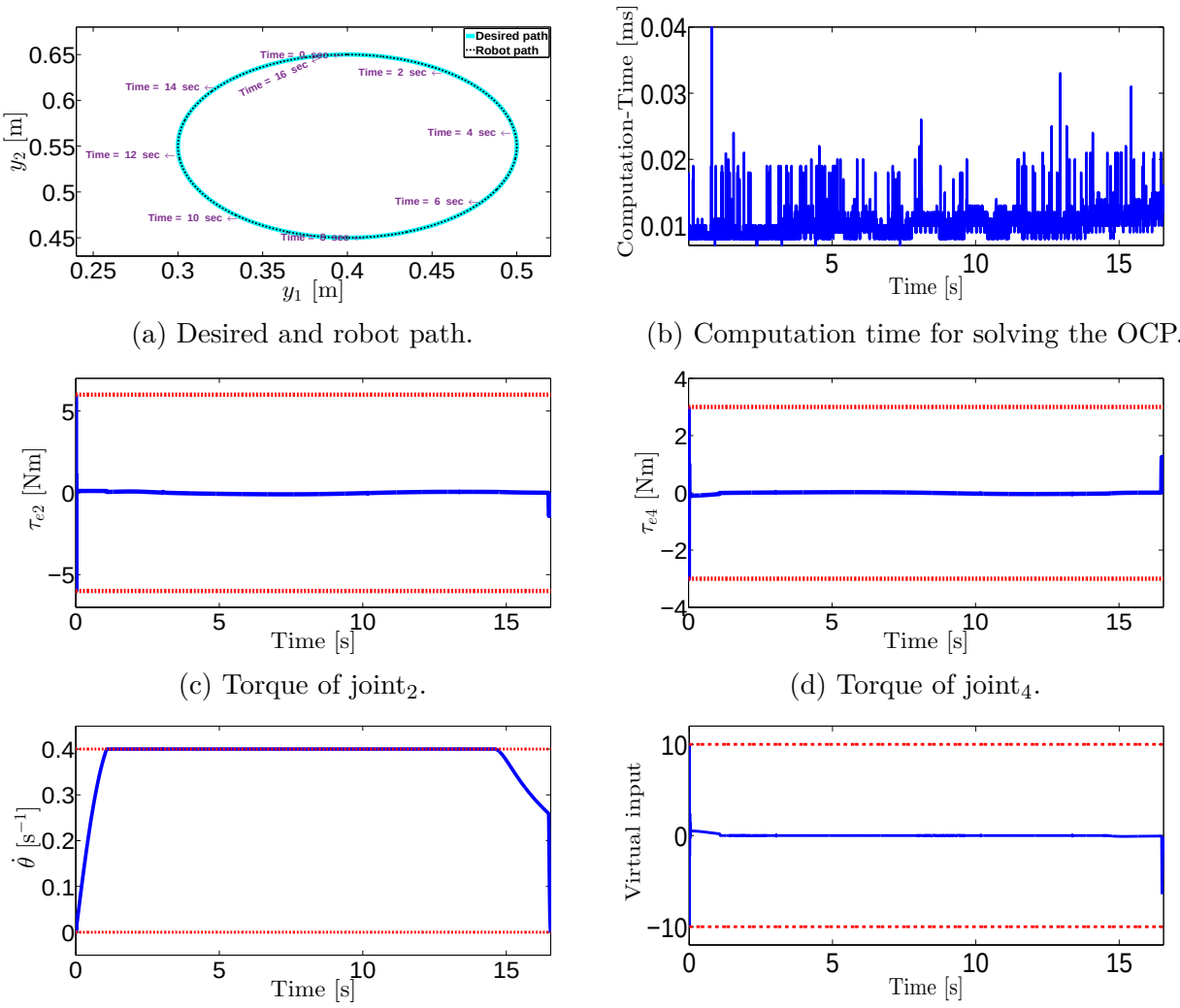
We use the prediction horizon $N = 30$ ms and a sampling rate $\delta = 3$ ms. The weight matrices in the cost function, i.e. W and R, and the constraint sets (box constraints) on input and state variables are listed in Appendix C.

For this setup, the solution of the path following problem (2.26) is tested and the results are depicted in Figure 5.3 and 5.2. In Figure 5.2 the end-effector follows a

circular-path, while in Figure 5.3 the end-effector follows a sine-path. This step is necessary to tune the OCP parameters and check the path followability accuracy before adding the admittance dynamics as a constraint in the OCP. The results of Figures 5.2 and 5.3 show the desired path is followed while the imposed constraints on states, inputs and path parameters are respected. The average value of the path following errors after converging to the path (after 0.1 [s]) for both circular- and sine-path are listed in Table 5.1. The parameters of the path-following problem are provided in Appendix C.

Table 5.1: **Path-following average error**

Circular-path	3.444 e-6
Sine-path	1.284 e-6



(a) Desired and robot path.

(b) Computation time for solving the OCP.

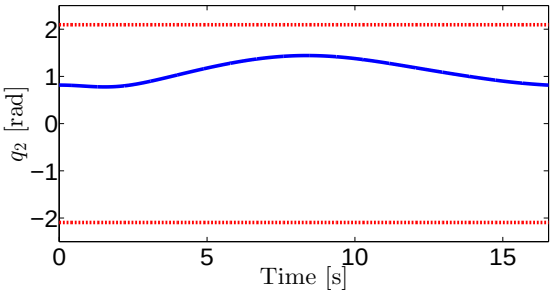
(c) Torque of joint$_2$.

(d) Torque of joint$_4$.

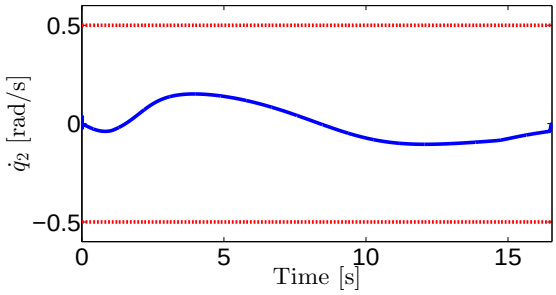
(e) Evolution of path-parameter w.r.t. time.

(f) Virtual input of timing-law.

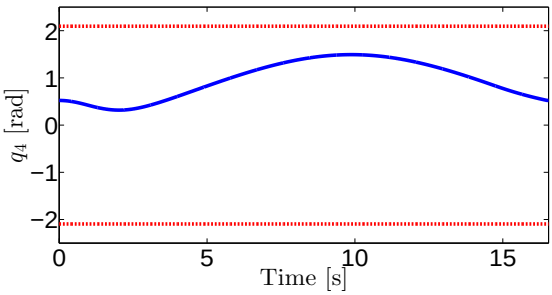
Figure 5.2: The robot's end-effector follows a circular-path.



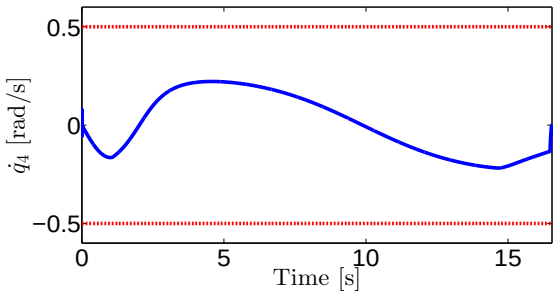
(g) Position of joint$_2$.



(h) Velocity of joint$_2$.

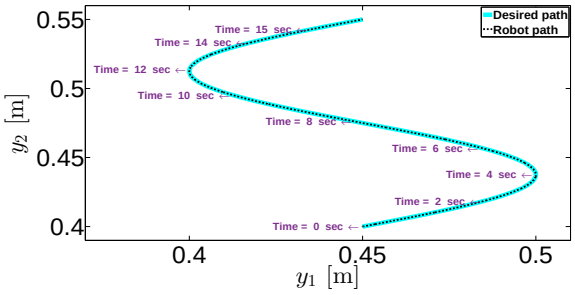


(i) Position of joint$_4$.

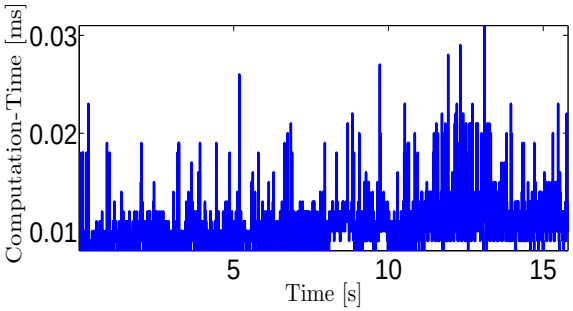


(j) Velocity of joint$_4$.

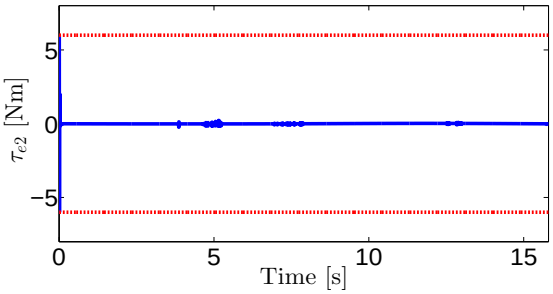
Figure 5.2: The robot's end-effector follows a circular-path. (Cont.)



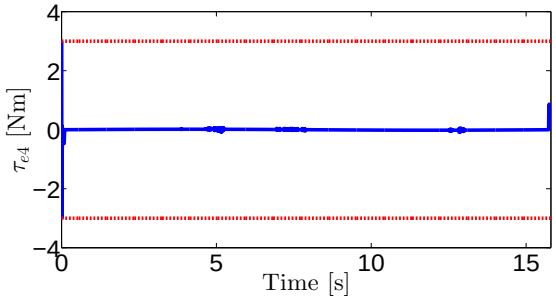
(a) Desired and robot path.



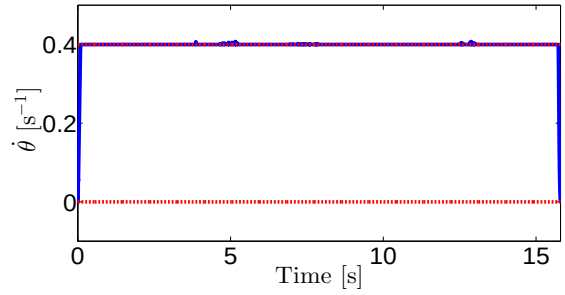
(b) Computation time for solving the OCP.



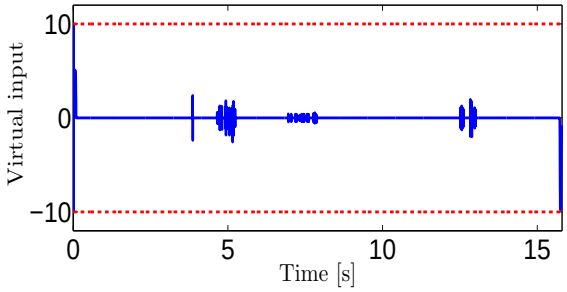
(c) Torque of joint$_2$.



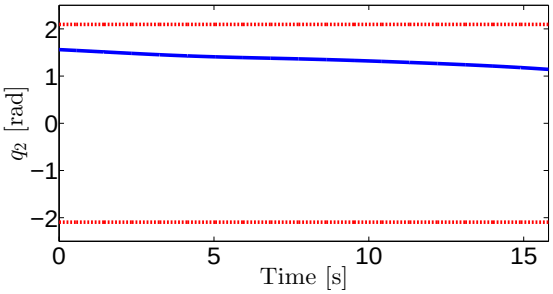
(d) Torque of joint$_4$.



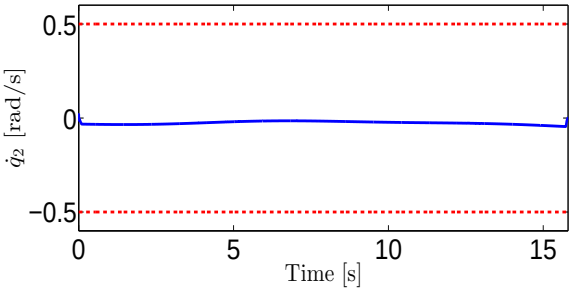
(e) Evolution of path-parameter w.r.t. time.



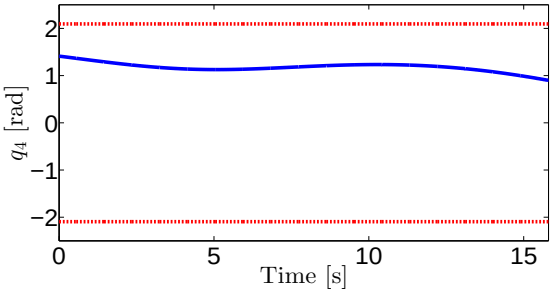
(f) Virtual input of timing-law.



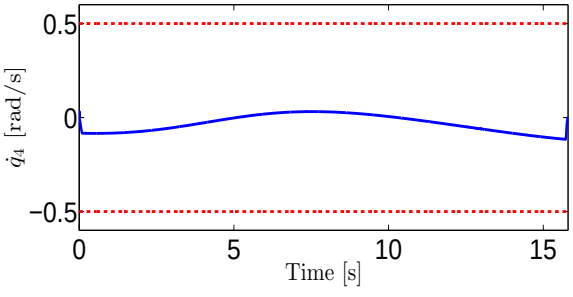
(g) Position of joint$_2$.



(h) Velocity of joint$_2$.



(i) Position of joint$_4$.

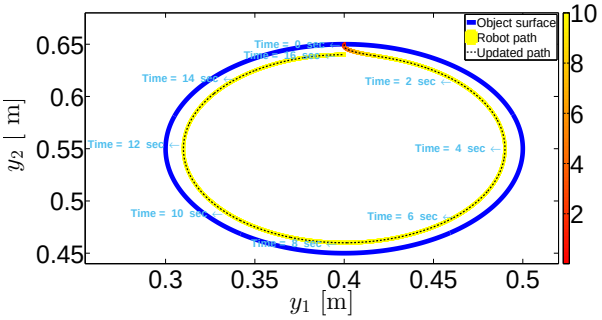


(j) Velocity of joint$_4$.

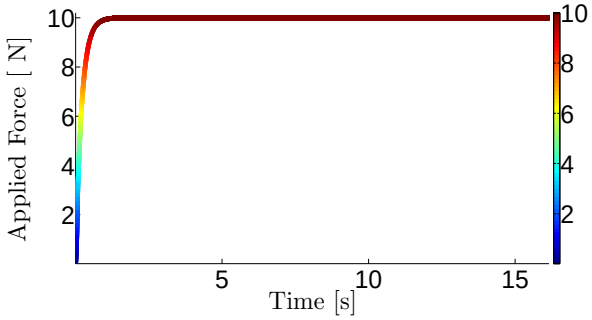
Figure 5.3: The robot's end-effector follows a sine-path.

Next, the solution of (4.41) (i.e. path-following and force control) is tested with different desired forces and a wide range of stiffness degrees. Here, the zero-stiffness admittance dynamics (4.35) is used. The desired admittance dynamics is designed based on the duality principle (see Section 3.5) and tuned to be a second-order critically damped system (see Remark 4.1). Which ensures smooth and fast convergence to the desired force. The parameters of the desired admittance for the different simulation setups are listed in Appendix C.

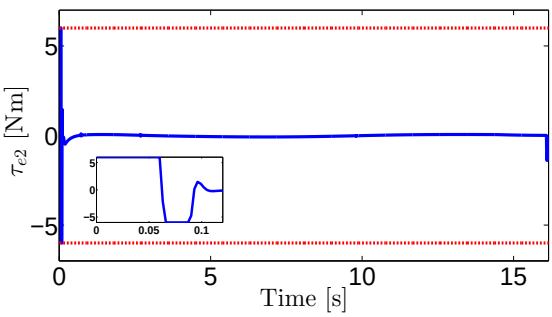
The first task is to follow a circular-path while applying a 10 [N] normal force (i.e. normal on the tangent of the path) and the result is presented in Figure 5.4. As shown in this figure, the updated-path (i.e. black-dashed line in Figure 5.4a) is followed, consequently the desired force is achieved as depicted in Figure 5.4b. Furthermore, the constraints on inputs (i.e. joint torques and virtual input of the timing-law) are respected as shown in Figures 5.4c, 5.4d and 5.4k, respectively. In addition, the position and velocity constraints are met (see Figures 5.4e- 5.4h). The path-parameter θ is constrained to change from -2π to 0 as shown in Figure 5.4i. While, the speed along the path, i.e. $\dot{\theta}$ is constrained between [0, 0.4], which ensures a forward motion along the path, Figure 5.4j shows fulfillment of this limit. The computation time is much less than the sampling time (i.e. $\delta = 3ms$) as shown in Figure 5.4j, this ensures feasibility of the real-time implementations and fast convergence of the state estimator error (see Section 2.3.1).



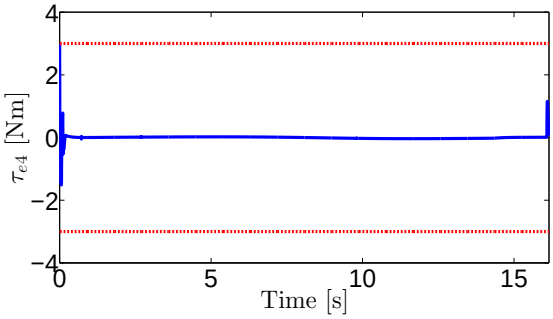
(a) Desired and robot path.



(b) Contact force.

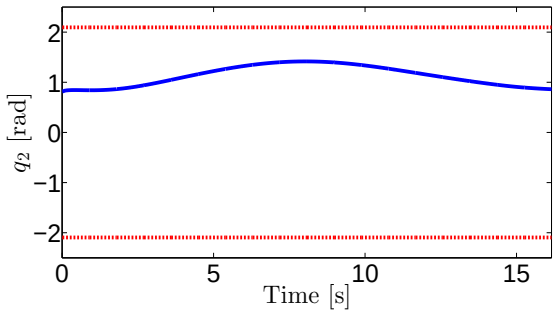


(c) Torque of joint$_2$.

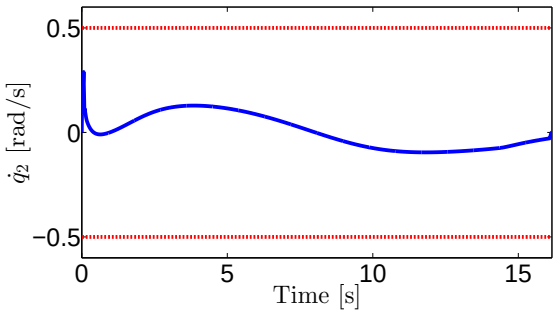


(d) Torque of joint$_4$.

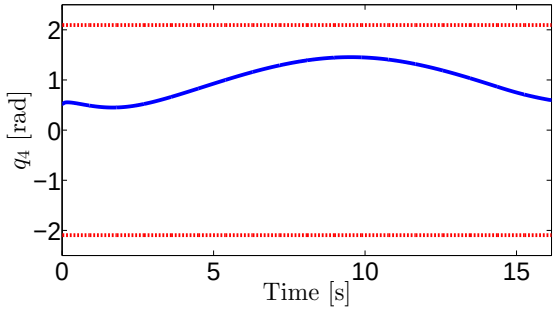
Figure 5.4: The robot's end-effector follows a circular-path and applying 10 [N] force.



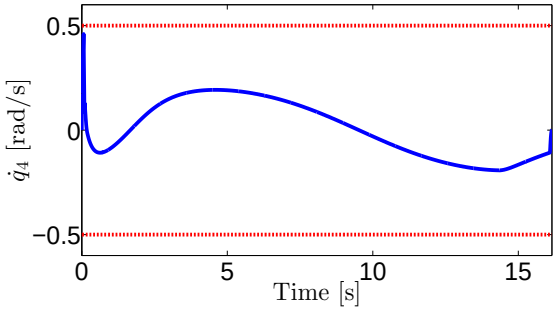
(e) Angular position of joint$_2$



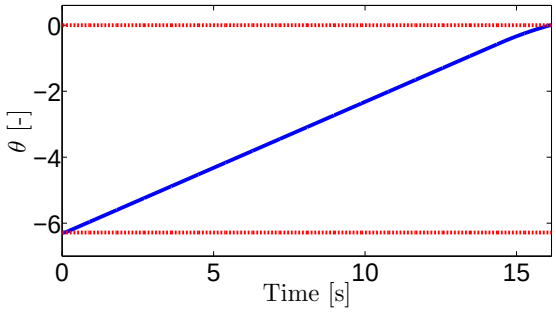
(f) Angular velocity of joint$_2$



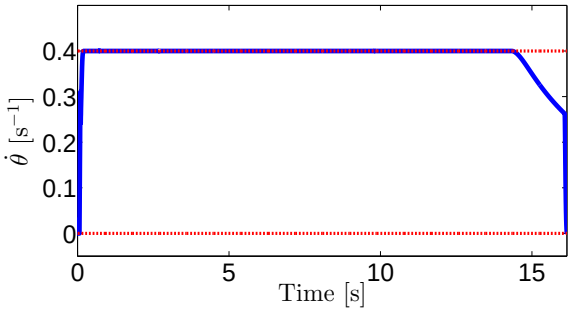
(g) Angular position of joint$_4$.



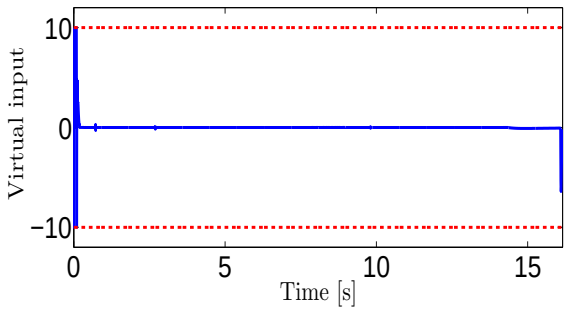
(h) Angular velocity of joint$_4$.



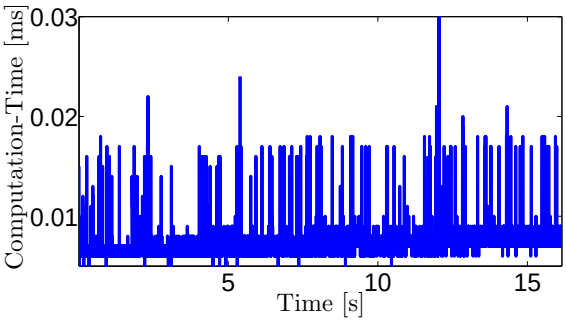
(i) Path-parameter variable.



(j) Evolution of path-parameter w.r.t. time.



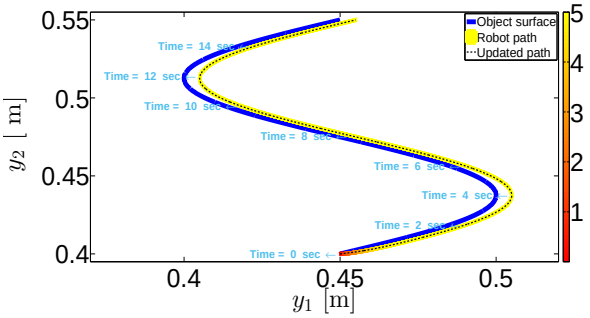
(k) Virtual input of the timing-law.



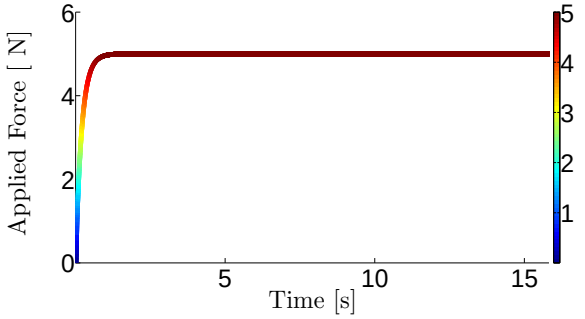
(l) Computation of the OCP.

Figure 5.4: The robot's end-effector follows a circular-path and applying 10 [N] force. (**Cont.**)

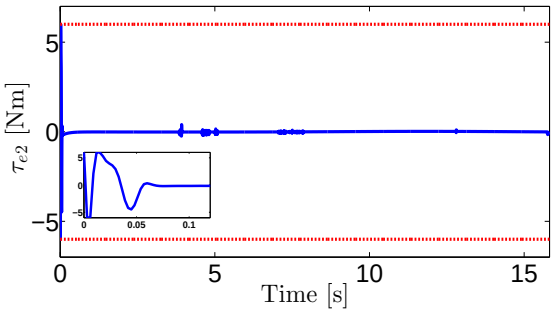
Alternatively, in Figure 5.5, the end-effector follows a sine-path and exerting a 5 [N] force along the y_1 direction, see Figure 5.5a. Here, we used the same constraints and parameters as before for solving the OCP (4.41). Figure 5.5 shows the following of the updated-path, cf. Figure 5.5a, when a 5 [N] force is applied, cf. Figure 5.5b. All other constraints, i.e. states, inputs and parameters constraints, are fulfilled as depicted in Figure 5.5.



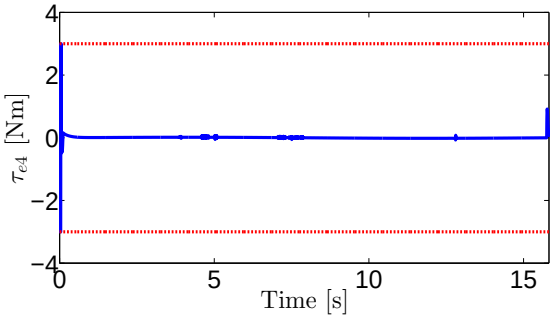
(a) Desired and robot path.



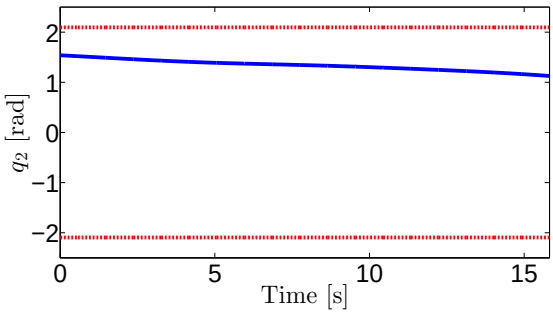
(b) Contact force.



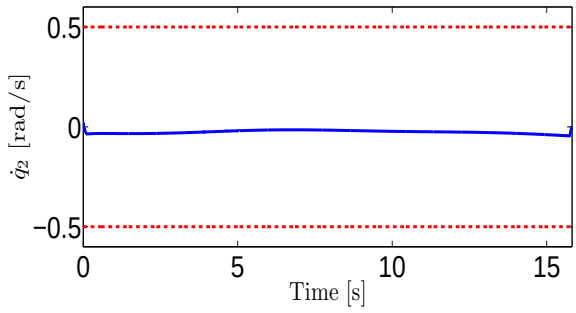
(c) Torque of joint$_2$.



(d) Torque of joint$_4$.

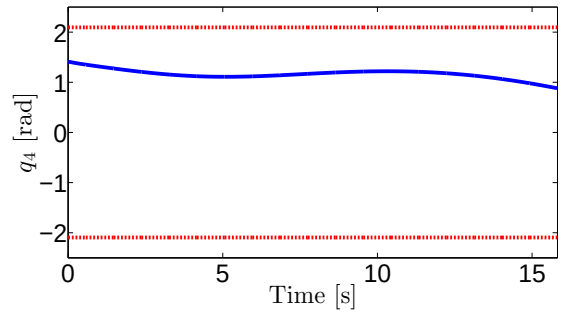


(e) Position of joint$_2$

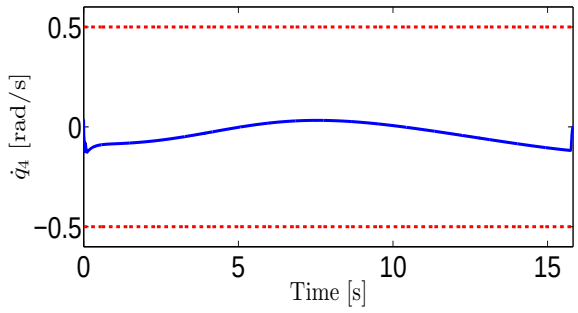


(f) Velocity of joint$_2$

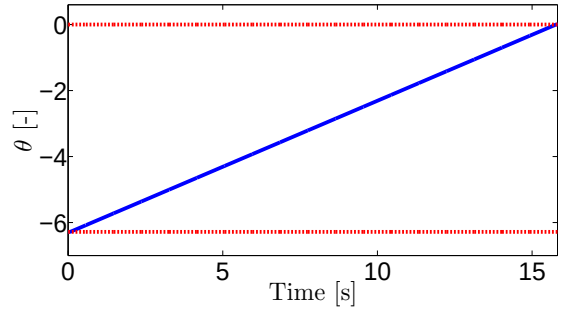
Figure 5.5: The robot's end-effector follows a sine-path and applying 5 [N] force.



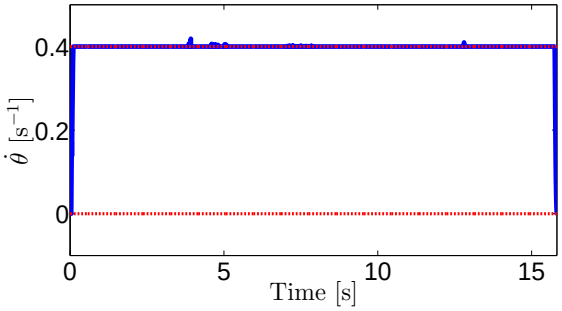
(g) Position of joint$_4$.



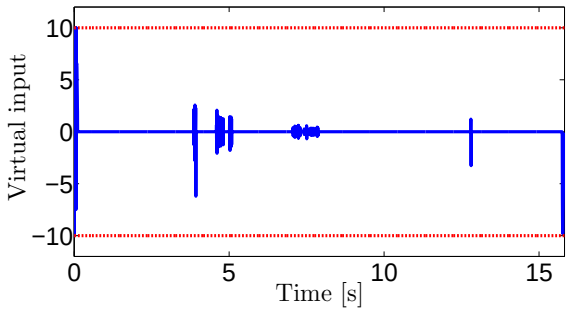
(h) Velocity of joint$_4$.



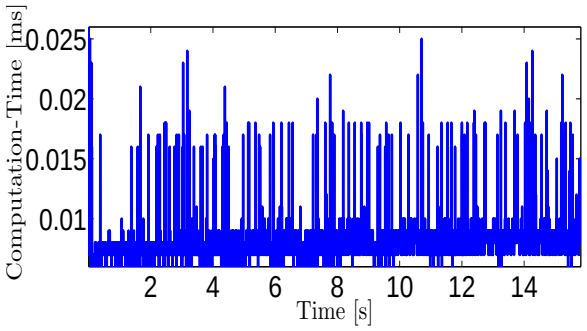
(i) Path-parameter variable.



(j) Evolution of path-parameter w.r.t. time.



(k) Virtual input of the timing-law.



(l) Computation of the OCP.

Figure 5.5: The robot's end-effector follows a sine-path and applying 5 [N] force. (**Cont.**)

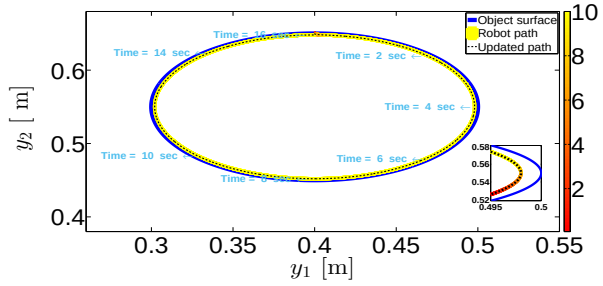
Figure 5.6 shows the applicability of the proposed approach for different degrees of stiffness. Here, we assumed that the model of the environment is known inaccurately, therefore, we used the zero-stiffness admittance dynamics (4.35). In addition, we used the same constraints on states, inputs and parameters for all simulations in Figure 5.6, are listed in Table C.2. As we mentioned in Section 4.2.1.1, the zero-stiffness admittance (4.35) is asymptotically stable despite that the environment stiffness k_e is not known accurately, however, the tuning of gains m and b is required to achieve a smooth convergence [67]. Therefore, we fixed the mass parameter to $m = 1[\text{kg}]$, while changing the damping parameter according to the rule

$$b = 2\,\zeta\,\sqrt{k_e\,m},$$

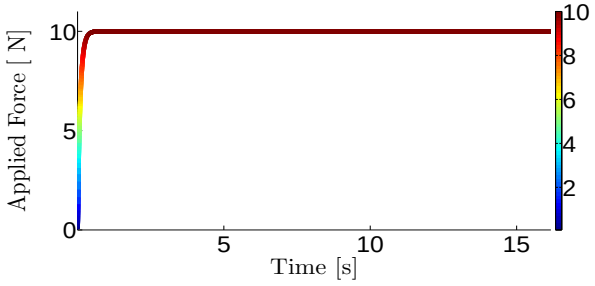
where ζ is a dimensionless constant called the damping ratio. Here, we choose $\zeta = 3.25$, which ensures overdamped behavior of the admittance dynamic system, which is preferable, especially for stiff environments. The evolutions of states, inputs and parameters were omitted in Figure 5.6, since all constraints are met. To exemplify this, the evolutions for the case in Figure 5.6i and Figure 5.6j (i.e. for the environment $k_e = 5\times10^5$ [N/m]) is depicted in Figure 5.7. Figure 5.6 depicts the results for different degrees of environmental stiffness, i.e.

$$k_e \in \left\{5\times10^3, 1\times10^4, 5\times10^4, 1\times10^5, \text{and}\, 5\times10^5\right\}.$$

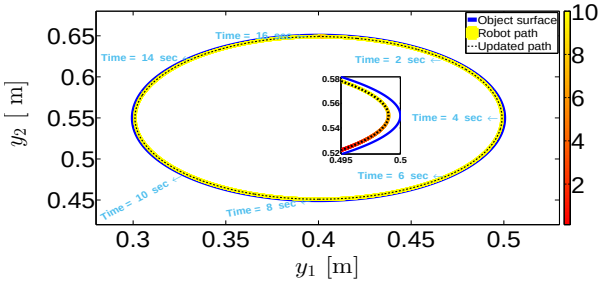
The comparative results, depicted in Figure 5.6, show the convergence to the desired force can be achieved for a wide range of environments using the zero-stiffness admittance approach. It is worth mentioning that in Figure 5.6 the time derivative of the path-parameter (i.e. $\dot{\theta}$) is bounded between [0, 0.4] [s^{-1}] (see Figure 5.7h).



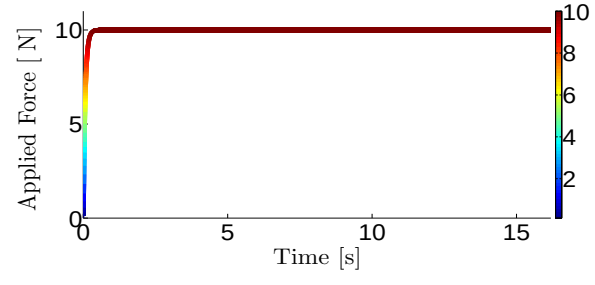
(a) $k_e = 5 \times 10^3$ [N/m]



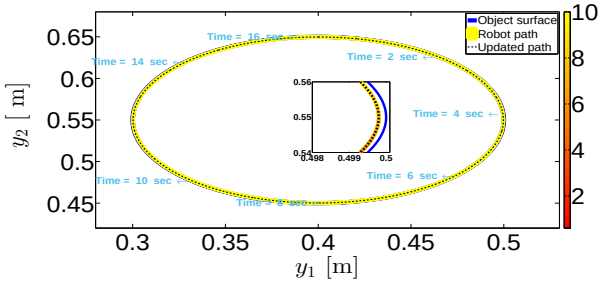
(b) F^d= 10 [N] and $k_e = 5 \times 10^3$ [N/m]



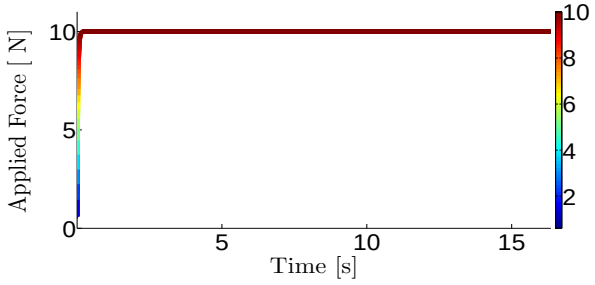
(c) $k_e = 10 \times 10^4$ [N/m]



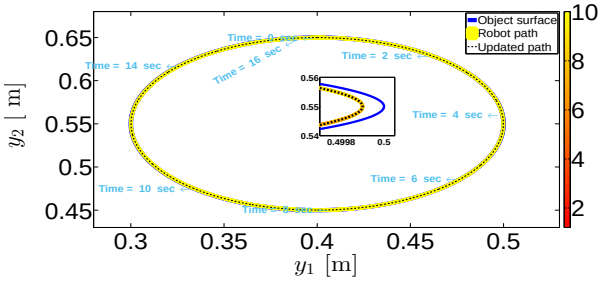
(d) F^d= 10 [N] and $k_e = 1 \times 10^4$ [N/m]



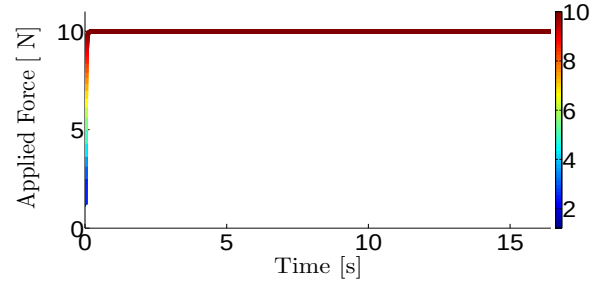
(e) $k_e = 5 \times 10^4$ [N/m]



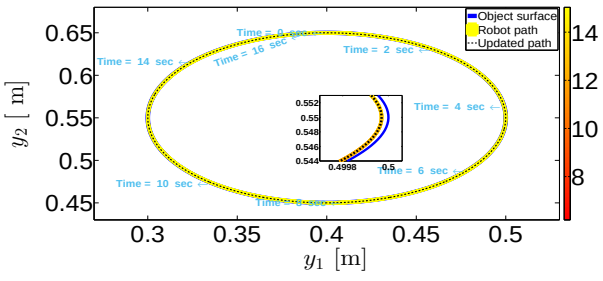
(f) F^d= 10 [N] and $k_e = 5 \times 10^4$ [N/m]



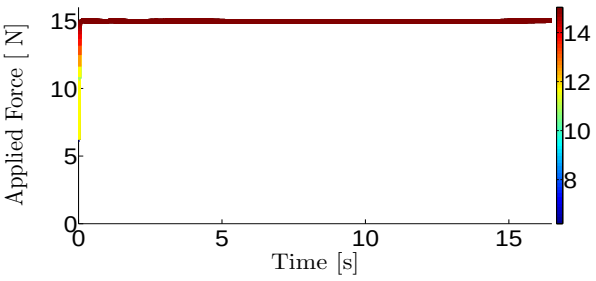
(g) $k_e = 1 \times 10^5$ [N/m]



(h) F^d= 10 [N] and $k_e = 1 \times 10^5$ [N/m]

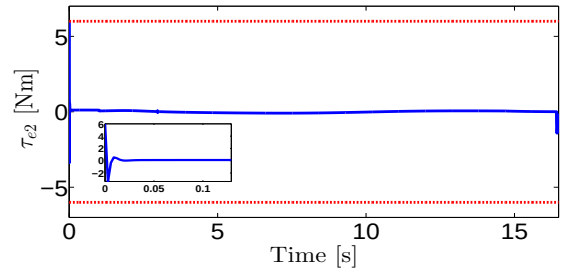


(i) $k_e = 5 \times 10^5$ [N/m]

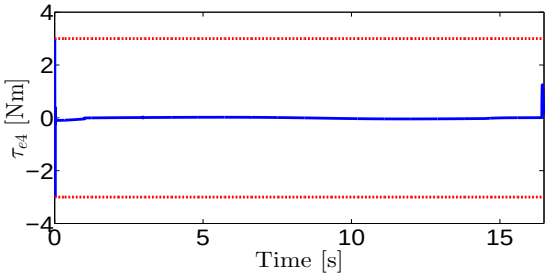


(j) F^d= 15 [N] and $k_e = 5 \times 10^5$ [N/m]

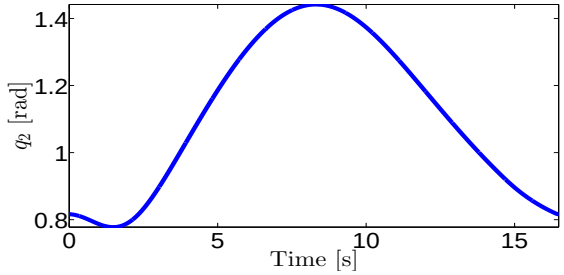
Figure 5.6: Comparison among different degrees of stiffness, (left) Cartesian path, (right) the corresponding applied force.



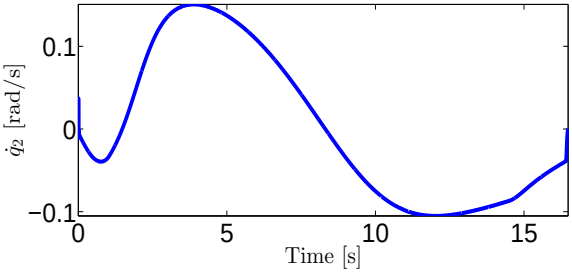
(a) Torque of joint$_2$.



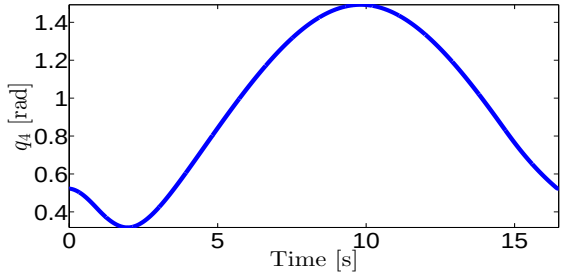
(b) Torque of joint$_4$.



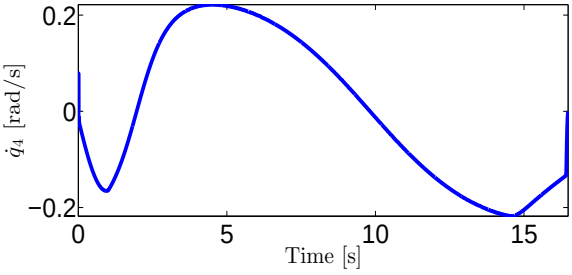
(c) Angular position of joint$_2$



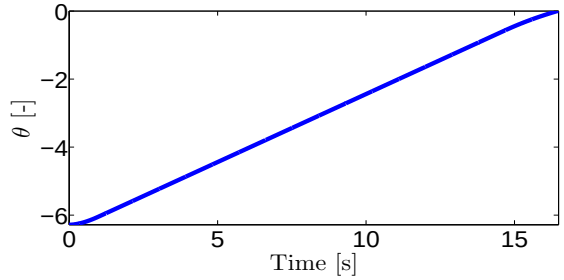
(d) Angular velocity of joint$_2$



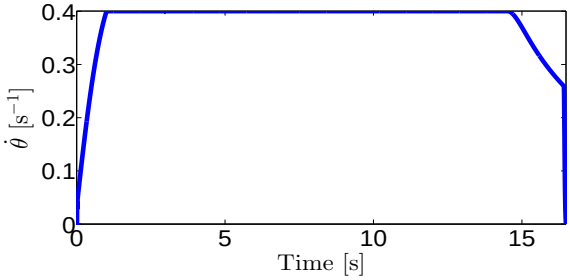
(e) Angular position of joint$_4$.



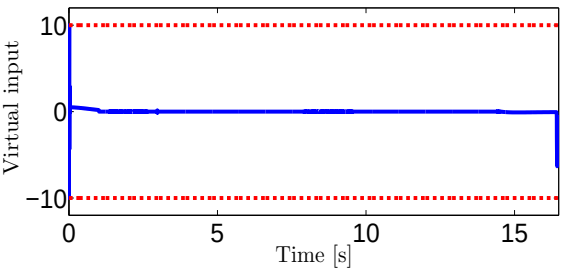
(f) Angular velocity of joint$_4$.



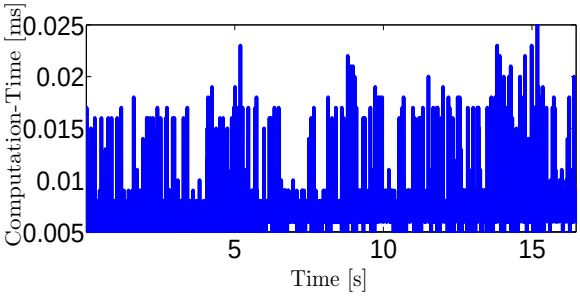
(g) Path-parameter variable.



(h) Evolution of path-parameter w.r.t. time.



(i) Virtual input of the timing-law.



(j) Computation of the OCP.

Figure 5.7: States, inputs and parameters evolutions corresponding to the environment stiffness $k_e = 5 \times 10^5$ in Figure 5.6i.

5.1.3 Non-nominal Simulations

In the following simulation setup, we tested applicability of our approach when the environment model is uncertain and a sensor-measurement delay exists. This setup is important to study the effects of uncertainty and communication quality on the stability and performance of the controller before the proposed approach is tested on the real robot. Figure 5.8 shows the schematic of the second simulation setup, where the toolbox [14] provides the emulation of the real robot arm as depicted in Figure 5.8. This toolbox contains the identified model of the robot used in the real experiment, i.e. KUKA LWR IV robot arm, as well as a library that emulates the communication interface used in the real setup, i.e. the so-called *Fast Research Interface* [104]. The comparison between the result of this simulation with the previous -ideal- simulation step gives us the ability to tune and safely test the proposed controller.
In this step, we tackle the following issues:

1. Uncertain environment model.
2. Time-delayed and noisy measurements.

In the following we explain these issues in details.

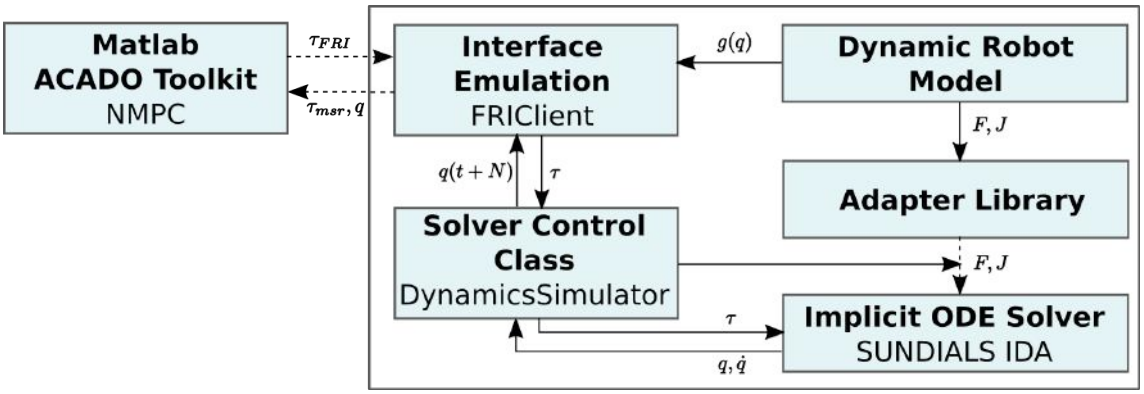


Figure 5.8: Toolbox emulates the KUKA LWR IV robot arm [14].

5.1.3.1 Uncertain environment model

To show the robustness of our approach, we test the proposed optimal control problem using nominal and zero-stiffness admittance under uncertain environment stiffness. Where in this experiment, the environment stiffness is changing sinusoidally from 1000 to 1200 [N/m]. Figure 5.9 confirms the robustness of the zero-stiffness admittance dynamics to the uncertainties in the environment model.

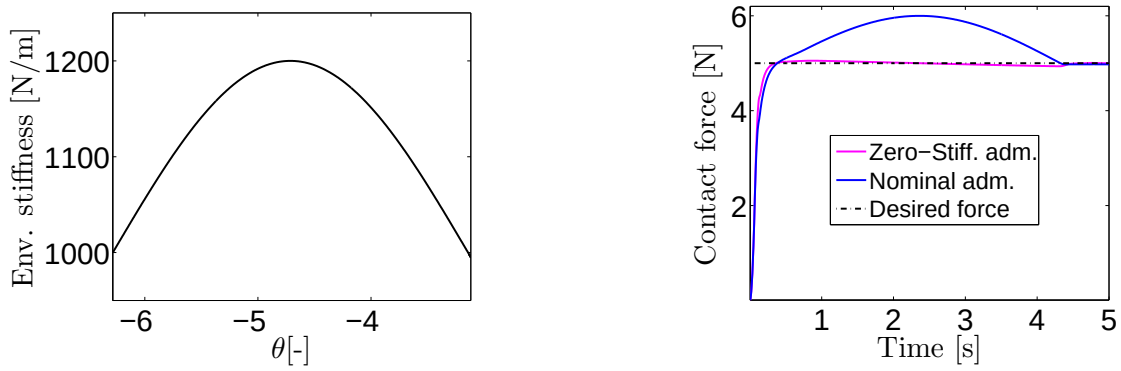
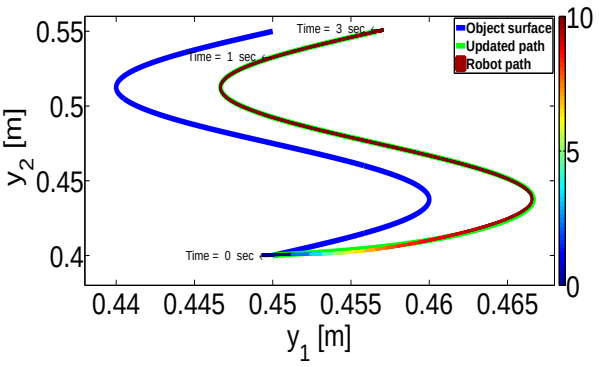


Figure 5.9: Variable environment stiffness (left), contact force (right).

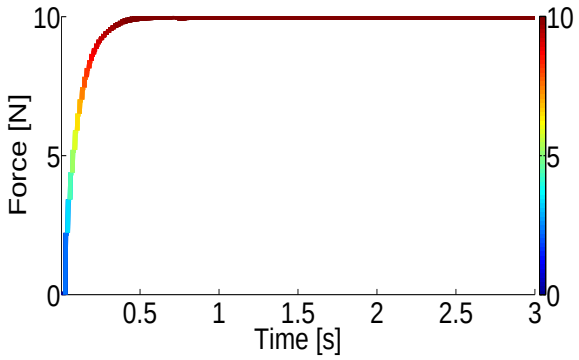
5.1.3.2 Time-delayed and noisy measurements.

Time-delays in measurement happen due to dropped packets during communication or because of slow computations in the data acquisition process. In Figure 5.10, we show that the proposed approach is robust to measurement time-delays. In Figure 5.10, we assumed the sampling time is $\delta = 3$ ms, while the time delay for the force sensor equals $5 \times \delta$ and $10 \times \delta$. The convergence to the desired force is achieved, however, there is chattering in the force trajectory. This chattering is proportional to the time-delay in the force measurement because the update of the followed path depends on the admittance position trajectory, which is in turn depending on the measured force. That is, if the force signal is delayed for t_d seconds, consequently the position and force errors in the nominal admittance is kept constant for that time.

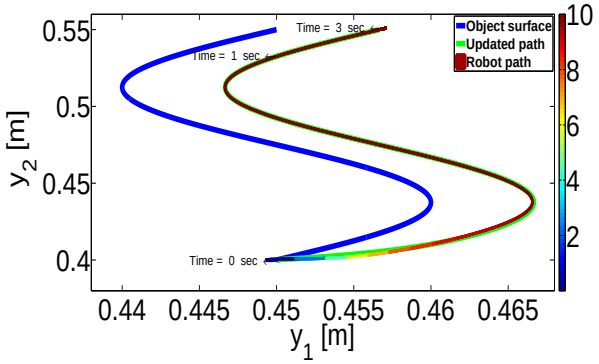
Remark 5.1 (Discrete force sensor model)
Recall that the admittance dynamics is just a virtual model, i.e. the coupling between robot dynamic model and admittance dynamic model does not exist in reality. However, inside the controller both models are coupled. This coupling leads to incorrect computations of the input because our controller is model-based. Actually, a simple investigation of the OCP in (4.1) *indicates that the coupling between models is due to the force sensor model. Where, the force sensor model is a function of the end-effector position. To decouple the dynamic models, we used a different sensor model. This model delivers the current value of the force at each sampling time and keeps it constant until the next sampling time. The new model of the force sensor is more realistic since the measurement of the real force sensor is delivered to the controller at each sampling time.*



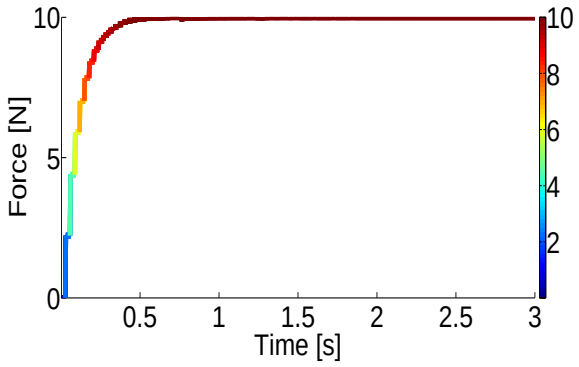
(a) Sensor time-delay $= 5 \times \delta = 0.015$ [s]



(b) Sensor time-delay $= 5 \times \delta = 0.015$ [s]



(c) Sensor time-delay $= 10 \times \delta = 0.03$ [s]



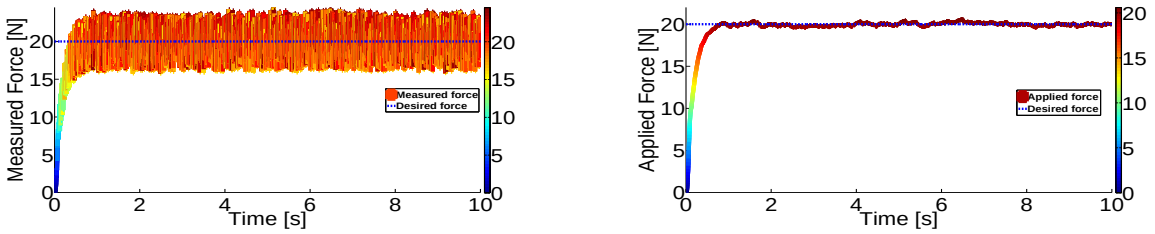
(d) Sensor time-delay $= 10 \times \delta = 0.03$ [s]

Figure 5.10: Effect of force sensor time-delay while following a circular-path and applying 10 [N] with $k_e = 15 \times 10^2$ [N/m]

With respect to the force sensor noise, we tested the robustness of the proposed approach by adding Gaussian noise to the measured force. For the nominal admittance dynamics, cf. Figure 5.11 and the zero-stiffness admittance, cf. Figure 5.12 with parameters as listed in Table 5.2 and the other constraints/parameters as in Appendix C. Both figures show the robustness for a 20% Gaussian noise with low environment stiffness, i.e. $k_e = 2000$ [N/m]. However, the zero-stiffness admittance is more robust than the nominal admittance for the same environment stiffness and desired force, cf. Figure 5.11 and Figure 5.12b. While for high environmental stiffness degrees, both approaches are not robust, cf. Figures 5.12c and 5.12d. Basically, noisy force measurements lead to a mismatch between real and estimated dynamic parameters, consequently, the residual $\tilde{r}$ (3.14) becomes non-zero and becomes larger for high values of stiffness and forces. In general, the proposed NMPC scheme is robust against uncertainties in the system model. Due to our observation, the effect of noisy measurements can be eliminated by increasing the weight on the path-parameter error in the objective function. This prioritizes the convergence on the path over the convergence to the path, where the last is affected by the noisy measurements.

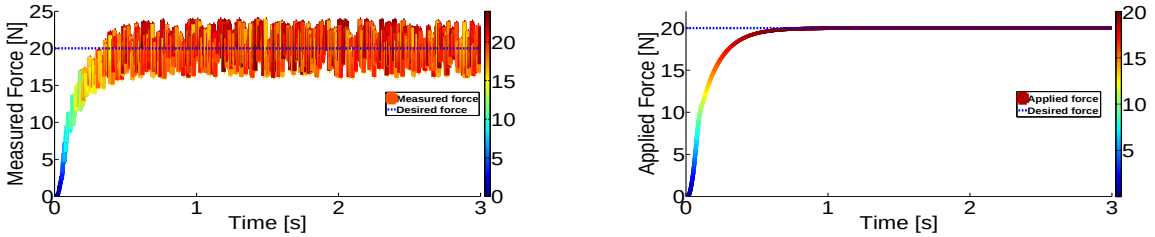
Table 5.2: **Desired admittance parameters**

	Mass	Damping	Stiffness
Nominal Admittance	1 [kg]	440 [N.s/m]	300 [N/m]
Zero-stiffness Admittance	1 [kg]	$6.5\sqrt{k_e m}\,[N.s/m]$	0 [N/m]

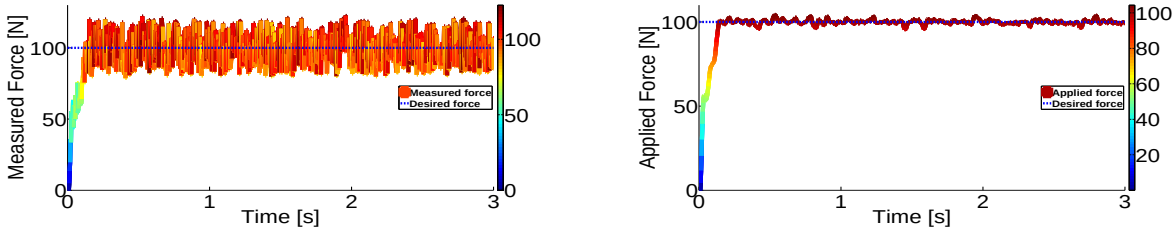


(a) Measured force with Gaussian-noise= 20 % . (b) Applied force.

Figure 5.11: Compensating of measurement noise using nominal admittance, measured noisy force signal (left), corresponding applied force (right).



(a) Measured force with Gaussian-noise= 20 % . (b) Applied force.



(c) Measured force with Gaussian-noise= 20 %. (d) Applied force.

Figure 5.12: Compensating of measurement noise using zero-stiffness admittance, measured noisy force signal (left), corresponding applied force (right).

5.2 Experimental Validation

In this section, the real-time experimental validation of the proposed approach is performed. To this end, we used the KUKA LWR IV robot arm (cf. Figure 5.1) to test the implementation of the presented path-following (i.e. the OCP (2.26)) and path-following with force feedback control (i.e. the OCP (4.41)). The task is to write

on a plastic ball (see Figure 5.13). Figure 5.14 shows the schematic of the experimental setup. Where the controller send torques to the Fast Research Interface, which in turn adds a gravity torque to the commanded torques and feeds back the real measured torques and positions.

Figure 5.13: KUKA robot writing on a plastic ball.

Figure 5.14: Schematic of the experimental setup

5.2.1 Estimation of unmeasured states

In the experimental work, we can not measure all states. So to complete the initial states vector

$$\mathcal{X}_{ini} = [\underbrace{q_0}_{position} \quad \underbrace{\dot{q}_0}_{velocity} \quad \underbrace{\hat{x}_0}_{timing-law} \quad \underbrace{\gamma_0}_{admittance}],$$

In the real experiment, we do not have velocity state measurement, therefore we must estimate the velocity state using the measured position states. Since the position is measured at each sampling instance, i.e. discrete-time variable, we use the difference quotient method to estimate the velocity state:

$$\dot{q}(t_k) = \frac{\partial q}{\partial t} \mid_{t=t_k} \approx \frac{q_k - q_{k-1}}{t_k - t_{k-1}}.$$

The fast sampling rate and the small computation time of the controller allow to use this simple estimation method. Furthermore, to overcome the noise effects, we add a

low-pass filter. Then, the estimated velocity state is delivered to the controller with the other measured and constructed internal states, i.e. $\hat{x}_0, \delta_0$, which are constructed using the timing-law and the desired admittance dynamics respectively.

5.2.2 Environmental parameters

To design the desired admittance, we need to know the environment parameters. First, we measure the environment stiffness. To this end, we apply a force on the plastic ball along x-axis (see Figure 5.15), where the Barrett 6-Axis Force/Torque sensor [118] is used to measure the force. We repeat this experiment for different depths (i.e. $\gamma = \{2, 4, 6, 8, 10, 15\}$ [mm]) as shown in Figure 5.16. Then we use linear regression method to estimate the environment stiffness from the measured force and corresponding depth of penetrations. As shown in Figure 5.17, environment stiffness increases with the depth of penetration, this is reasonable because the air pressure is increasing proportionally to the depth of compression.

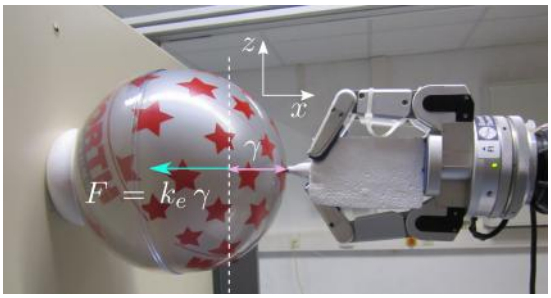


Figure 5.15: Measuring environment stiffness.

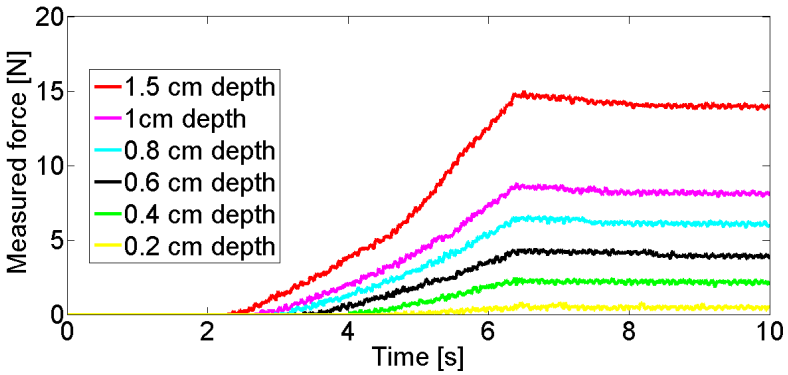


Figure 5.16: Measured forces corresponding to different depths.

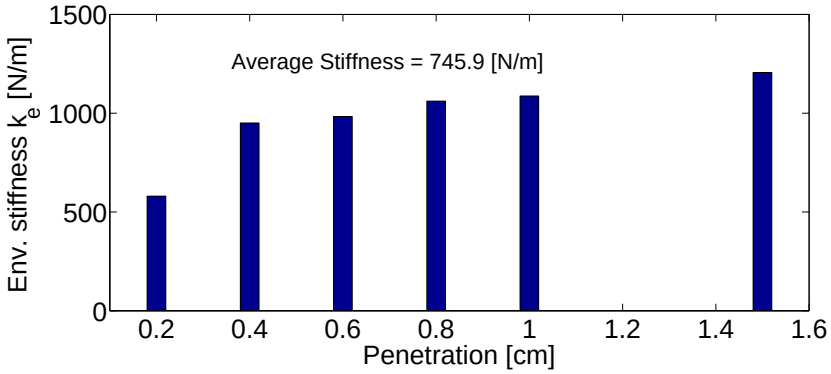


Figure 5.17: Average environment stiffness.

Second, we investigate the accuracy of the environment position (i.e. position of ball's surface). To this end, we perform an experiment for following a circular-path on the ball's surface while applying 0 [N]. Here, we use our approach with the zero-stiffness admittance. This experiment shows that the radius of the ball is not homogenous (see Figure 5.18).

These last two experiments show the inaccuracy in the environment parameters. Next, we perform a task of writing on the plastic ball while maintaining a 2 [N] contact force. Where, the pen attached to the robot's end-effector moves along a circular-path on the ball's surface and keeping 2 [N] contact force (see Figure 5.13). To achieve this, we used our proposed approach with nominal admittance (Figures 5.19−5.22) and zero-stiffness admittance (Figures 5.23−5.26). The parameters of the experimental work are listed in Table C.4.

As shown in Figure 5.19, the desired force is not achieved by using nominal admittance. The reason for that is due to inaccuracies in the environment parameters. Where, as we discussed in Chapter 4, convergence of the nominal admittance dynamics requires accurate stiffness and position of the environment. In contrast, the zero-stiffness admittance is achieved the desired force as shown in Figure 5.23. IN these both experiments, the path-parameter θ is constrained to change from -1.65π to -1.4π as shown in Figure 5.20 and Figure 5.24.

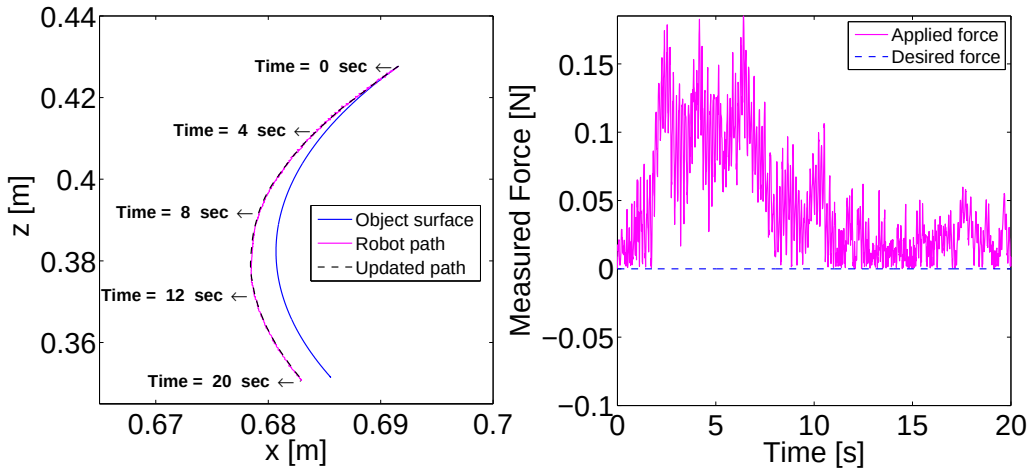


Figure 5.18: Moving along the ball's surface with 0 [N] desired force, followed path (left), measured contact force (right).

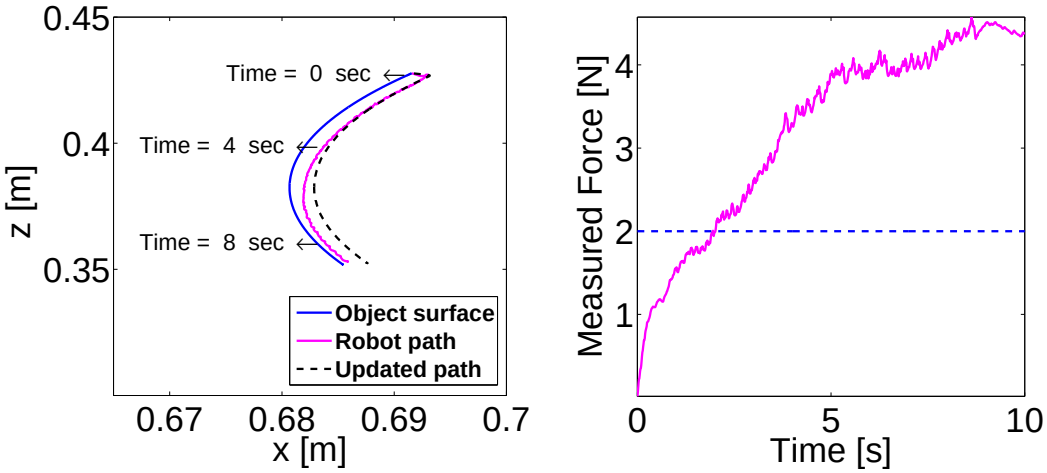


Figure 5.19: Moving along the ball's surface with 2 [N] desired force , followed path (left), measured contact force (right). (Nominal admittance)

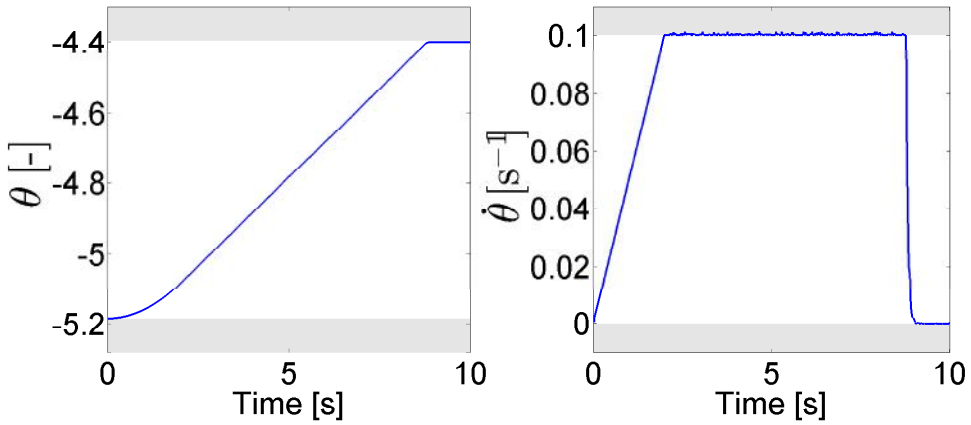


Figure 5.20: Path-parameter (left), evolution of path-parameter (right). (Nominal admittance)

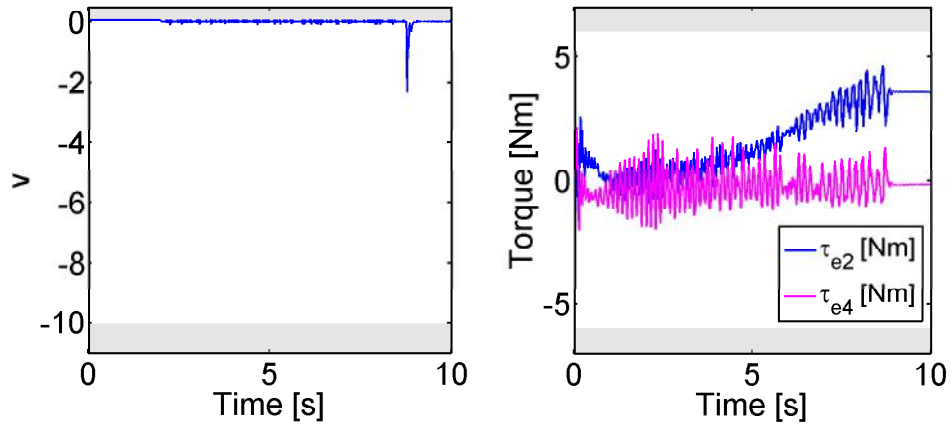


Figure 5.21: Timing-law input (left), robot input torques (right). (Nominal admittance)

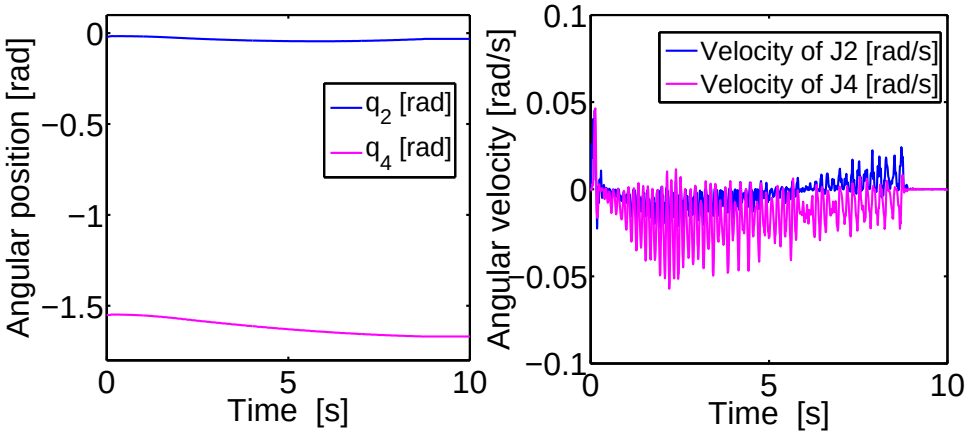


Figure 5.22: Angular positions (left), estimated angular velocity (right). (Nominal admittance)

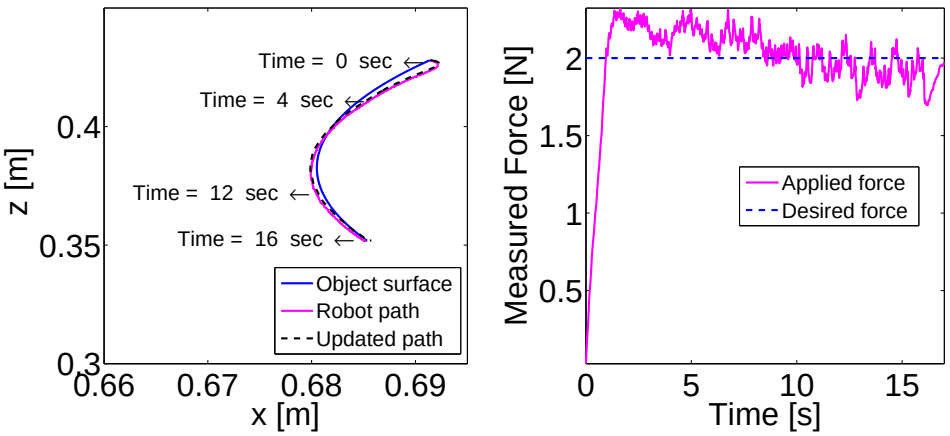


Figure 5.23: Moving along the ball's surface with 2 [N] desired force , followed path (left), measured contact force (right). (Zero-stiffness admittance)

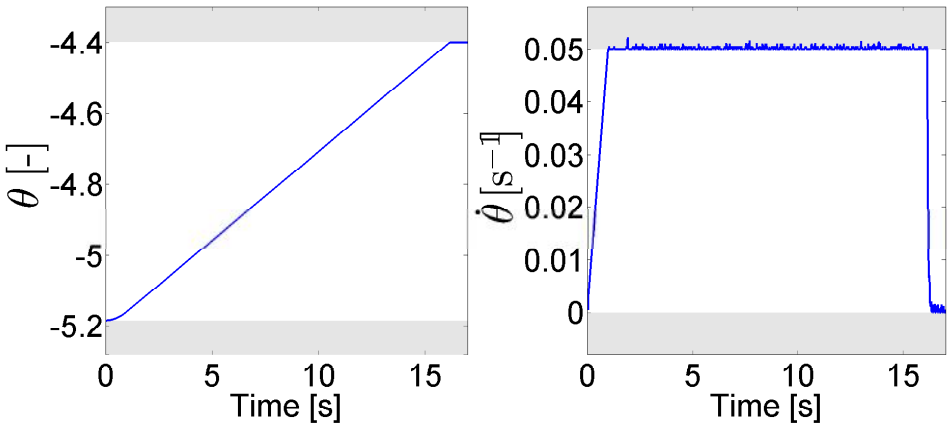


Figure 5.24: Path-parameter (left), evolution of path-parameter (right). (Zero-stiffness admittance)

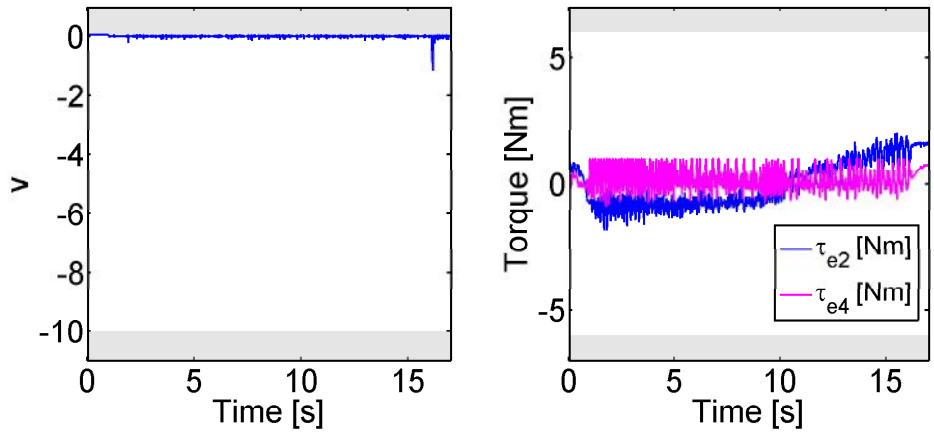


Figure 5.25: Timing-law input (left), robot input torques (right). (Zero-stiffness admittance)

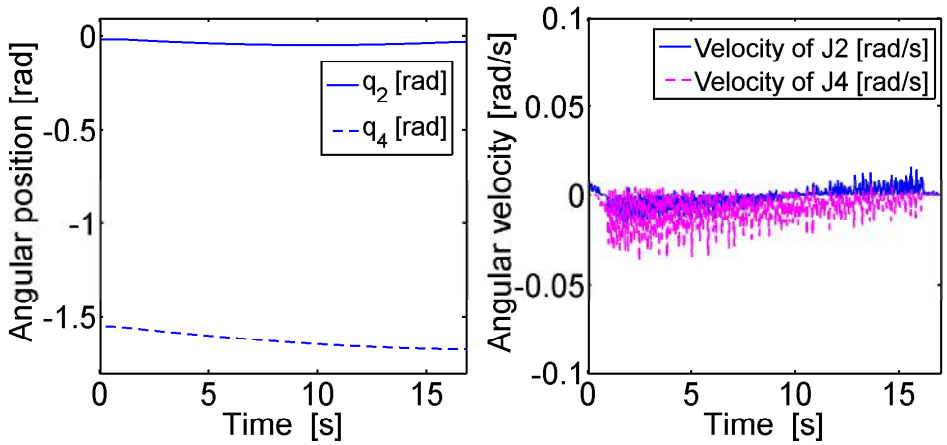


Figure 5.26: Angular positions (left), estimated angular velocity (right). (Zero-stiffness admittance)

5.3 Discussion

Here, we discuss the main observations from the simulation and experimental results. As described in Chapter 4, in our method the indirect force control approach −admittance force control− is included inside the predictive path-following problem. This led to solving the path-following and force control problem simultaneously in one optimization problem within the model predictive scheme. Basically, we have exploited the features of model predictive control and path-following to include the force control.

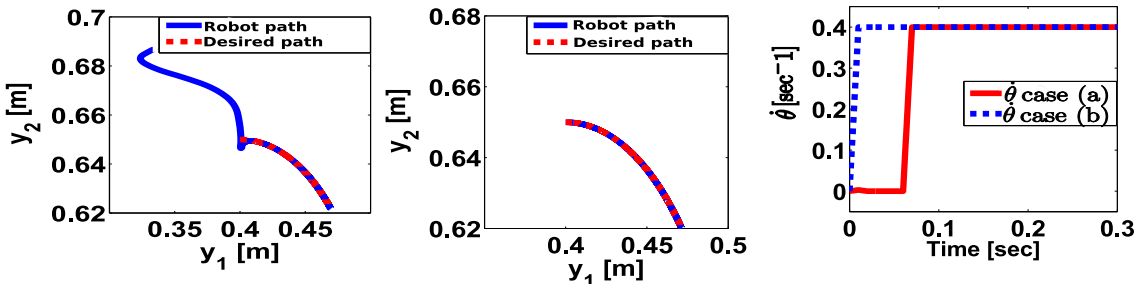
With respect to the model predictive control, as explained in Chapter 2, for obtaining the optimal solution of the OCP the dynamic model of the system is used to predict the future evolution of the system within the prediction horizon. Then, repeatedly solving the optimal control problem using new measurements as initial states will cope with disturbances or uncertainties in the dynamic model. Regarding the path-following, the path-parameter evolution is left as a free variable to make the output error (**??**) as small as possible.

In the following, we discuss the simulation results in detail.

Path-following and force-feedback

In (OCP) (4.41) when the path is updated using the desired admittance dynamics, the output error increases, consequently, the evolution of the path-parameter slows down to reduce the output error. This is due to the fact that the performance index penalizes the deviations of the system output from the path and the distance of the path-parameter from its final value. So, by increasing the weights in the performance index on the deviation from the path, one can prioritize convergence to the path.

That is, the evolution of the path-parameter variable slows down when the deviation from the path is large, (see Figure 5.27). This property ensures fast convergence to the desired force as shown in the simulation results. Additionally, by including the augmented system dynamics (4.40) (i.e timing-law (2.23), desired admittance (4.26), and system model (2.21a)) as constraints in the OCP (4.41), the optimal solution will compromise among these dynamics to follow the updated-path as precise as possible. For example, Figures 5.4−5.5 show that the updated-path (i.e. black dashed-line in Figures 5.4a−5.5a) is followed, hence, the desired contact force is achieved as well.



(a) (b) (c)

Figure 5.27: Priority of convergence to the path, in (a) the initial state is not on the path, (b) the initial state is on the path, (c) the path-parameter evolutions corresponding to (a) and (b)

Due to the very short sampling time, i.e. 3 [ms], the applied torques and the virtual input of the timing law seem to chatter sharply, but if we zoom in the figures, the trajectory changes smoothly as shown in Figure 5.28.

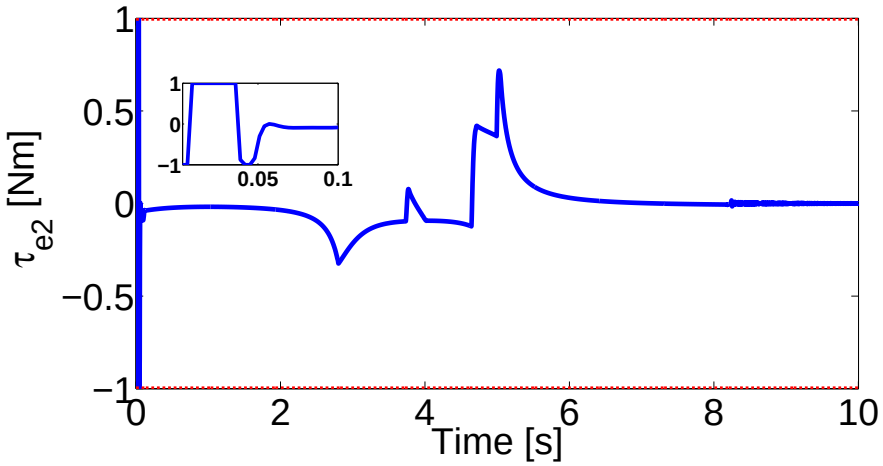


Figure 5.28: Torque$_1$ zoomed in.

Notes on nominal and zero-stiffness admittance dynamics

As we described in the last chapters, the admittance dynamic is designed as a second-order dynamic system. This system is non-homogeneous, since it is driven by the force error (nominal admittance (4.26)) or the desired force (zero-stiffness admittance (4.35)). In both cases the system is asymptotically stable as explained in Chapter 4. However, based on our observation, the response to the zero-stiffness admittance is sensitive to the desired force. In essence, this input function is a step-function while the input of the nominal admittance (i.e. force error) is decaying over time. Furthermore, for the nominal admittance, the position error and force error converge to zero simultaneously. For the zero-stiffness admittance, the position error grows proportionally to the desired force (see Section 4.2.1.1). Therefore, to achieve a good dynamic response for the zero-stiffness admittance, it is required to tune the damping and mass parameters.

Summary

In this chapter, we performed the practical work to verify the proposed approach for solving path-following and force feedback using model predictive control. We presented the model of the robot, which is adopted in simulation and experimental work. The applicability of the proposed approach was demonstrated with simulation studies and real experiments on a KUKA LWR IV robot arm. To do so, we employed different desired paths and desired contact-forces with different degrees of environmental stiffness. In the following chapter, we discuss safety conditions in manipulation processes.

6 Safety of Manipulation Processes

In this chapter, we discuss how the manipulation process performed the task safely. Where the problems of collision avoidance and uncontrollable approach to the environment are discussed. The motivation behind the first problem, is that the redundant robot can approach the same point with different configurations (see Figure 6.1), and some of these configurations are infeasible kinematically or kinetically. Therefore, it is required to avoid these kinds of configurations by including appropriately chosen constraints in the optimization problem. While the motivation for the second problem is that avoiding of uncontrollable approaching of the environment is necessary because this might be lead to instability. We solve the second problem by computing the so-called inevitable set.

6.1 Obstacle avoidance in the framework of predictive control

Obstacle avoidance control refers to the task of actively planning/controlling a path/trajectory while avoiding collisions with stationary or moving objects. We utilize an obstacle avoidance approach that is based on predictive control as presented in [103]. In this approach, the dynamic or static obstacles are considered as additional static or dynamic constraints in the optimal control problem.

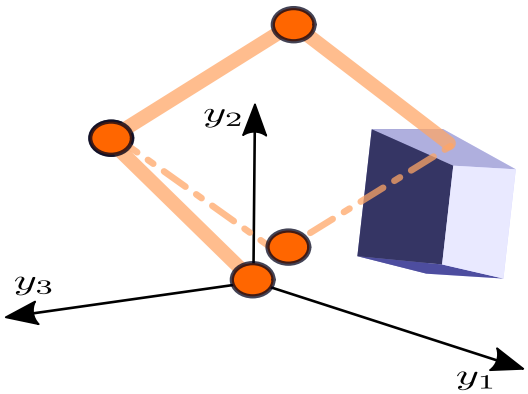


Figure 6.1: Redundant configuration

However, the method presented in [103], does in general not guarantee repeated feasibility for small prediction horizons, i.e. if the controller cannot foresee the complete path to the goal. To achieve guaranteed repeated feasibility is in general a hard task.

We tackle this problem by guaranteeing that for each final position achieved always a feasible trajectory exists, which does not intersect with the obstacles. To this end, we consider the following two sets/problems.

Problem 6.1 (Unreachable set)
Determine the set os states, which are unreachable by the system without hitting an obstacle due to the kinematic and geometry constraints of the robot. ■

Problem 6.2 (Inevitable set)
Determine the set of states, where a collision with an obstacle cannot be avoided. ■

With unreachable set Problem 6.1, we denote the set that a robot cannot reach without hitting the obstacle. As an example consider Figure 6.2 depicting a 2-DOF robot. In Figure 6.2, the robot - denoted as Cartesian robot - can move link_1 only vertically, and link_2 only horizontally. As can be seen, a Cartesian robot is unable to reach any point, which is on the right hand side of the obstacle, since otherwise link_2 hits the obstacle.

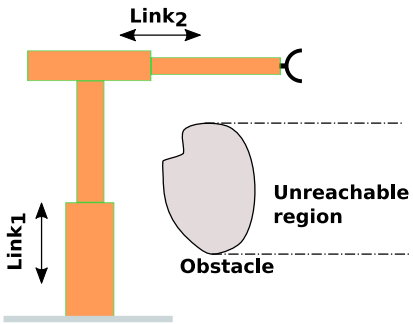


Figure 6.2: Sketch of a Cartesian 2-DOF robot. link 1 can only move vertically, and link_2 can only move horizontally. For the depicted obstacle the Cartesian robot is unable to reach anything on the right hand side of the obstacle.

Problem 6.2 refers to the set of states for which the dynamic system will eventually hit an obstacle no matter which input is applied. For example, when a robot arm moves very fast and faces an obstacle at the end of the predicted field of vision, there might not be enough space left to stop or maneuver around the obstacle, hence, a collision will occur, cf. Figure 6.3.

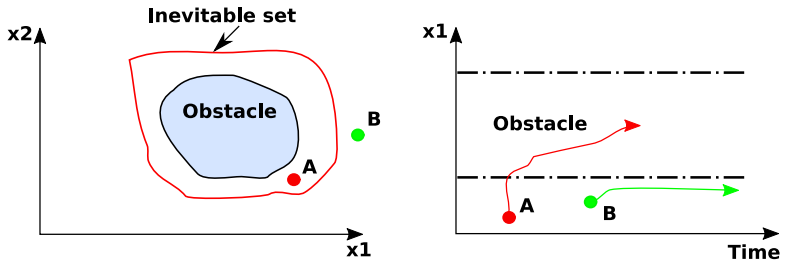


Figure 6.3: Inevitable set of initial conditions, for which a collision cannot be avoided. The blue area represents the obstacle, the red encircled area the inevitable set. For any initial condition taken from the inevitable set the robot will collide with the obstacle (red line), while for any initial condition outside, it can be guaranteed that the robot will never collide with the obstacle (green line).

Both sub-problems are described in more details in the next sections and illustrated with examples. Solving Problems 6.1 and 6.2 is challenging. However, we can solve Problem 6.1 using feasibility problems [16]. While Problem 6.2 can be solved by computing the set of initial conditions for which the reachable set is inside of the obstacle no matter what input is applied, i.e. one can employ the strategy of Rumschinski et al. [101] for systems with disturbances as seen in the following.

To this end, we consider a linear time invariant system describing the discrete-time dynamics of a robot

$$\begin{aligned} x(k+1) &= A\,x(k) + B\,u(k), \quad x(0) = x_0 \in \mathbb{R}^n, \\ y(k) &= C\,x(k), \end{aligned} \tag{6.1}$$

where $x(k) \in \mathcal{X} \subset \mathbb{R}^{n_x}$ denotes the states, e. g. the joint positions and their velocities, $u(k) \in \mathcal{U} \subset \mathbb{R}^{n_u}$ the control input, e. g. applied torques or forces, $y(k) \in \mathbb{R}^{n_y}$ the output, e. g. measured positions and velocities of the robot end-effector, and $k \in \mathbb{N} \cup \{0\}$ the time index. The initial condition is denoted by x_0, the state trajectory starting from x_0 at time k denoted by $\phi(k, u_k|x_0)$ for the input sequence $u_k = (u(0) = u_0, u(1) = u_1, \ldots, u(N) = u_N)$ and the output trajectory by $\phi_y(k, u_k|x_0) = C\phi(k, u_k|x_0)$. The real matrices A, B, C correspond to the physical connection of the states, input and output and are of appropriate dimensions.

6.1.1 Computation of the unreachable set

The Problem 6.1 corresponds to joint positions of the robot for which a link between the joints violates the obstacle, i.e. the obstacle is hit by a link. The main difficulty to overcome is the needed guarantee that no points are missed. Therefore, sampling based approaches, e. g. based on Monte Carlo experiments, are not satisfactory as one would need to check an infinite amount of points to guarantee the safety.

In [101], a similar problem is solved for proving model consistency of discrete-time

systems with measurements given as sets. It is shown that checking model consistency can be reformulated as a feasibility problem. In other words, one is interested in checking whether points fulfill all the constraints or not. We proposed a similar strategy in which the computation of the unreachable set corresponds to a feasibility problem. Basically, we compute the set of points for which the obstacle is violated by a part of the robot's link.
Before explaining the solution of Problem 6.1, we define the following sets. First, the obstacle to be avoided, e. g. a second robot or a human operator, is defined as follows

$$\mathcal{O} := \{x \in \mathbb{R}^{n_x} | Hx \leq h\} \subset \mathcal{R}^{n_x}, \tag{6.2}$$

where $H \in \mathbb{R}^{S \times n_x}$ and $h \in \mathbb{R}^S$ define the edges of a polytope.

The next equation corresponds to the geometry of the robot. Where, we define $\mathcal{L}_i$ to be the set of all points on link i, which connects the two joints denoted by i and $i+1$. The position of joint i in Cartesian space is denoted by j_i. The set $\mathcal{L}_i$ is defined as

$$\mathcal{L}_i = \{x \in \mathcal{X} : \exists (j_i, j_{i+1}) \in \mathcal{X} \mid \; x = \; \lambda\, j_i \; + \; (1-\lambda)\, j_{i+1};\; \forall \lambda \in [0,1]\} \tag{6.3}$$

where $\mathcal{L}_i$ for a specific value of λ indicates a point on link i of the robot arm. The indices are such that $i \in \{1, ..., I\}$, the total number of links is I, and the total joints number is $I+1$ including the end-effector. To prevent a violation of the obstacle w.r.t. the links of the robot (cf. Problem 6.1), one has to ensure that there exists no point x such that

$$\mathcal{L}_i \cap \mathcal{O} \neq \emptyset \tag{6.4}$$

holds. However, from a practical point of view, it is easier to derive first all the points for which a collision occurs and then exclude this set from the control problem (similar to an obstacle), cf. also to Section 6.1.2.1 for a more detailed discussion. Therefore, Problem 6.1 is written as follows

Problem 6.3 (Points of collision) *Determine the set $\mathcal{S}$ for which the* (6.4) *doesn't hold, where $\mathcal{S}$ is defined as follows*

$$\mathcal{S} := \{l_i \in \mathcal{L}_i \mid \; l_i = x_0;\; \forall\; x_0 \in \mathcal{O};\; j_i, j_{i+1} \in \mathcal{X};\; i \in \{1, 2, \cdots, I\}\} \supseteq \mathcal{O}. \tag{6.5}$$

■

To solve Problem 6.3, which corresponds to the geometry of a robot and is solved via

the following feasibility problem.

$$\begin{aligned} \text{find} \quad & x, \\ \text{s.t.} \quad & l_i = x_0, \\ & x \in \mathcal{X} \subset \mathbb{R}^{n_x}, \\ & x_0 \in \mathcal{O} \subset \mathcal{X}, \\ & l_i \in \mathcal{L}_i \subset \mathcal{X} \ \ \forall \ \ i \in \{1, 2, \cdots, I\}, \end{aligned} \tag{6.6}$$

It can be shown that the solution space of (6.6) is equal to $\mathcal{S}$. Thus, determining the solution space of (6.6) corresponds to solving Problem 6.3 (resp. 6.1).

In the case of the Cartesian robot cf. Figure 6.2, the resulting feasibility problem is linear and, therefore, convex. This allows us to compute the feasible set exactly with the help of standard primal-dual solvers for linear programs using the Matlab toolbox ADMIT [114]. The following Figure 6.4 illustrates the sets for the 2-DOF Cartesian robot (see Figure 6.2). Here, a polytopic obstacle is considered as depicted in the Figure 6.4. In the following, we consider the obstacle and the unreachable area as one obstacle denoted by extended obstacle. In the next section, we compute the set of inevitable states corresponding to the extended obstacle.

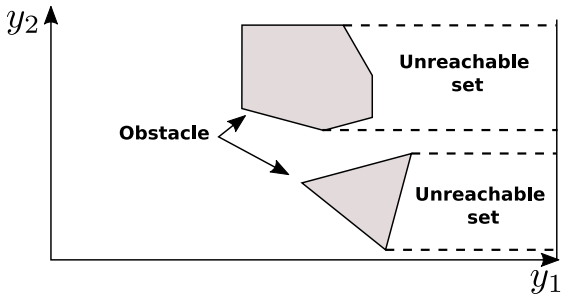


Figure 6.4: Unreachable subspace due to kinematics constraints. The gray area corresponds to the considered polytopic obstacle, while the area between dashed-lines corresponds to the unreachable set.

6.1.2 Computation of the inevitable set

We propose an approach to determine the set of initial conditions of a robot for which a collision with an obstacle cannot be avoided. Determining the inevitable set directly is, in general, difficult. Since one has to determine for an infinite amount of initial conditions whether there exists a time for which a collision occurs no matter which input is applied. To this end, we develop a new set-based recursive procedure based on the results presented in [18]. As a first step, we determine the set of initial conditions for which the robot collides with the obstacle in one-time step. Where, this set can be computed by assuming the obstacle as a target set and calculating the so-called

backward reachable set for one-time step employing the methods presented in [18]. However, the main problem in computing the backward reachable set is the condition that for any input the system has to end up in the obstacle. To avoid this problem, we treat the input as a disturbance and derive the so-called robust backward reachable set following [69].
To determine the next inevitable set we repeat the previous procedure using the first inevitable set as the new target set and again computing the backward reachable set for one-time step. This is repeated until the newly computed set is a subset of the previous one. It is obvious that if a collision can be in principle avoided, i.e. the dynamic system is controllable, then this condition will hold after a finite amount of repetitions and the procedure can be terminated. The inevitable set is formalized in the subsequent definition.

Definition 6.1 (Inevitable collision set) *The set $\mathcal{X}_{inevitable}$ is said to be an inevitable collision set, if for every initial condition $x_0 \in \mathcal{X}_{inevitable}$ there exists $t \in \mathbb{N} \cup \{0\}$ such that $x(t) \in \mathcal{O}$ no matter what input sequence $\bar{u}$ is applied, i.e.*

$$\mathcal{X}_{inevitable} := \{x_0 : \exists t \ s.t. \ \phi(t, u_k|x_0) \in \mathcal{O}, \forall u_k \in \mathcal{U}\}. \tag{6.7}$$

■

Employing Definition 6.1, we can formalize Problem 6.2 as follows

Problem 6.4 *Determine the set $\mathcal{X}_{inevitable}$ of system* (6.1). ■

To solve Problem 6.4, we start by defining the following set

$$\mathcal{X}^1_{inevitable} := \{x \in \mathcal{X} : Ax + Bu \in \mathcal{S}, \quad \forall u \in \mathcal{U}\}. \tag{6.8}$$

The set $\mathcal{X}^1_{inevitable}$ corresponds to the set of initial conditions for which the system (6.1) collides with the extended obstacle $\mathcal{S}$ in one-time step. Similar to $\mathcal{X}^1_{inevitable}$, we can recursively define the sets of initial conditions for which a collision cannot be avoided in $k + i; i \in \{1, 2, \ldots\}$ time steps, i.e.

$$\mathcal{X}^{k+1}_{inevitable} := \{x \in \mathcal{X} : Ax + Bu \in \mathcal{X}^k_{inevitable}, \quad \forall u \in \mathcal{U}\}. \tag{6.9}$$

there has to exist a time step $k^* \in \mathbb{N} \cup \{0\}$ for which

$$\mathcal{X}^{k^*+1}_{inevitable} \subseteq \mathcal{X}^{k^*}_{inevitable}, \tag{6.10}$$

holds, i.e. it is sufficient to compute iteratively the first k^* inevitable sets. The set $\mathcal{X}_{inevitable}$ is then simply the union of all sets $\mathcal{X}^k_{inevitable}, k \in \{1, \ldots, k^*\}$.

The main problem in computing $\mathcal{X}^k_{inevitable}$ is the condition for all $u \in \mathcal{U}$. To replace this difficult condition, we follow here a similar strategy as proposed in [18, 69] for the

computation of invariant sets, i.e.

$$\mathcal{X}_{inevitable}^{k+1} = \left\{ x : Ax \in \mathcal{X}_{inevitable}^{k} \sim BU \right\}, \tag{6.11}$$

where $\sim$ is the Minkowski difference, which is defined as follows,

Definition 6.2 (Minkowski difference)
The Minkowski difference of two sets $Q, W \subseteq \mathbb{R}^n$

$$Q \sim W := \{ c \in \mathbb{R}^n : c + w \in Q, \forall w \in W \}. \tag{6.12}$$

This reformulation allows us to present the following algorithm for the computation of $\mathcal{X}_{inevitable}^{k}$, where the implementation of the algorithm employs the toolbox presented in [69].

[**Inevitable Set Algorithm**]

$k \leftarrow 0$
$\mathcal{X}_{inevitable}^{k} = \mathcal{S}$
COM: *Compute* $\mathcal{X}_{inevitable}^{k+1} = \left\{ x : Ax \in \mathcal{X}_{inevitable}^{k} \sim BU \right\}$
if $\mathcal{X}_{inevitable}^{k+1} \subseteq \mathcal{X}_{inevitable}^{k}$, **then**
    terminate the algorithm and set $\mathcal{X}_{inevitable} = \bigcup_k \mathcal{X}_{inevitable}^{k}$
else
    $k \leftarrow k + 1$
    goto *COM*
end if

The result of the proposed algorithm is the union of all computed one-step inevitable sets, which might be non-convex. This fact makes the solution of the optimal control problem challenging. Therefore, only the convex hull of the inevitable sets is considered in Section 6.1.2.1. The convex hull is computed using the Matlab toolbox YALMIP [76].

We illustrate the result of the proposed algorithm by computing the inevitable collision set for the following Cartesian robot

Example 6.1 *The dynamics of 2-DOF Cartesian robot is given by*

$$x(k+1) = \begin{bmatrix} 1 & 0.05 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0.05 \\ 0 & 0 & 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} 0 & 0 \\ 0.0031 & 0 \\ 0 & 0 \\ 0 & 0.0016 \end{bmatrix} u(k), \qquad x \in \mathbb{R}^4, u \in \mathbb{R}^2. \tag{6.13}$$

Additionally, we consider the state constraints $(x_1, x_3) \in [0.5, 1]$*;* $(x_2, x_4) \in [-0.2, 0.2]$*, the input constraint* $u \in [-4, 4]$ *and the extended obstacle is considered see Figure 6.4.*

Due to the decoupled dynamic system (6.13)*, the inevitable set of each link can be computed separately. The result is depicted in Figure 6.5 for one link of the robot.*

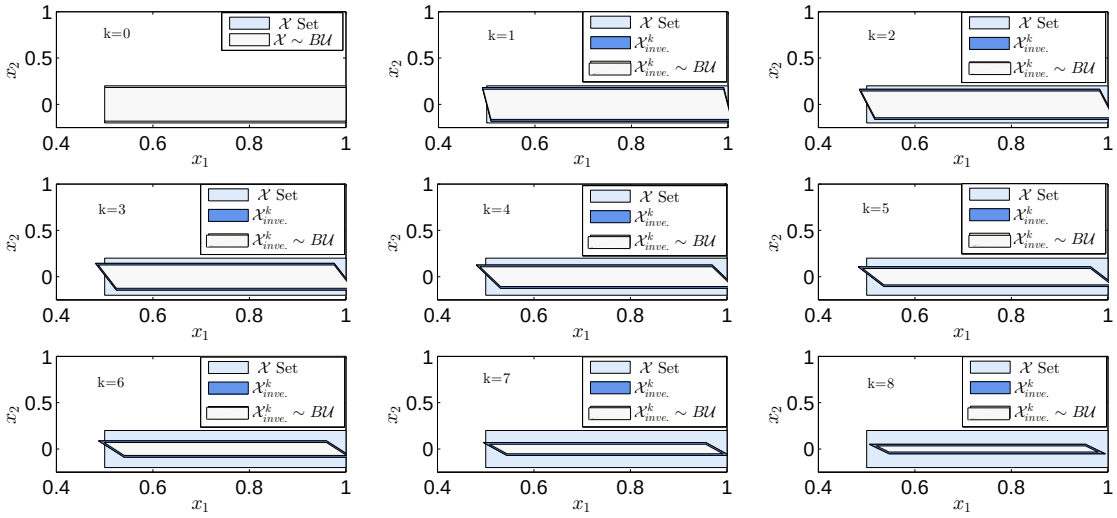


Figure 6.5: The recursive computation of the inevitable set for the link_1 of the 2-DOF Cartesian robot for 8 steps. The blue sets correspond to the inevitable sets. The white-blue sets represent the obstacle, and the gray sets are the Minkowski difference between the input set and the inevitable set.

The decoupling of the dynamic system (6.13) *allows us, furthermore, to investigate the influence of the velocity constraints on the size of the inevitable set as depicted in Figure 6.6.*

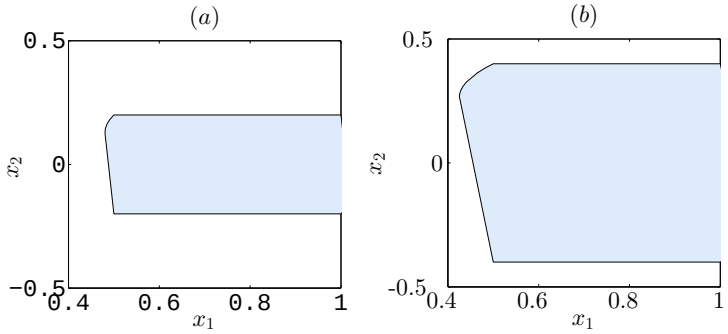


Figure 6.6: (a) Convex hull of the inevitable set when $x_2 \in [-0.2, 0.2]$, and (b) Convex hull of the inevitable set when $x_2 \in [-0.4, 0.4]$

Figure 6.6 illustrates nicely that if the robot is allowed to move faster, the inevitable set becomes larger.

6.1.2.1 Collision free, recursively feasible MPC

As mentioned before, MPC solves at each sampling instance a finite horizon control problem over a certain prediction horizon capture the time varying obstacle, the uncertainties in the dynamic model, and the path constraints. To guarantee collision avoidance, we propose to augment the system by the inevitable set as constraints, i.e. for a linear system leading to

$$
\begin{aligned}
& min_u \sum_{k=0}^{N-1} J_k(x(k), u(k)) && \text{(6.14a)}\\
s.t.\quad & x(k+1) = Ax(k) + Bu(k), \quad k = \{0, 1, ..., N-1\}, && \text{(6.14b)}\\
& x(0) = x_0, \quad x_{k+1}(N) \in \Omega, && \text{(6.14c)}\\
& x(k) \in \mathcal{X}, \quad u(k) \in \mathcal{U}, && \text{(6.14d)}\\
& x(k) \notin \mathcal{X}_{inevitable}. && \text{(6.14e)}
\end{aligned}
$$

The constraint (6.14c) force the robot to start at the initial state $x(0)$ (position, velocity) and to reach the target set $\Omega \subseteq \mathbb{R}^n$ at terminal time N. (6.14d) represents state and input constraints. Safety of the robot's motion is guaranteed by (6.14e), where $\mathcal{X}_{inevitable}$ denotes the set of points possibly occupied by obstacles at time k or the points from which a collision cannot be avoided. Finally, J_k denotes a cost function representing a performance measure whose sum is to be minimized.

To solve the resulting optimization problem, we propose to reformulate the collision constraint (6.14e) as Mixed-Integer linear inequalities using the big-M method[1] [98]. The resulting optimization problem becomes a mixed-integer linear program (MILP).

$$
\begin{aligned}
& min_u \sum_{k=0}^{N-1} J_k(x(k), u(k)) && \text{(6.15a)}\\
s.t.\quad & x(k+1) = Ax(k) + Bu(k), \quad k = \{0, 1, ..., N-1\}, && \text{(6.15b)}\\
& x(0) = x_0, \quad x_{k+1}(N) \in \Omega, && \text{(6.15c)}\\
& x(k) \in \mathcal{X}, \quad u(k) \in \mathcal{U}, && \text{(6.15d)}\\
& M \begin{bmatrix} z \\ w \end{bmatrix} \leq d, && \text{(6.15e)}\\
& z \in \{0, 1\}^{p_1}, && \text{(6.15f)}\\
& w \in \mathbb{R}^{p_2}, && \text{(6.15g)}
\end{aligned}
$$

where $M \in \mathbb{R}^{r \times (p_1 + p_2)}$, p_1 is the number of binary variables, while p_2 is the number of real state variables, and $d \in \mathbb{R}^r$. The inequality (6.15e) represents the reformulation

[1] See Appendix B for the definition of big-M method

of $\mathcal{X}_{inevitable}$ using the big-M method.

Summary

We presented solutions for the obstacle avoidance problem and the set of inevitable initial states. Where, the latter is important to avoid an uncontrolled approach of the robot to a handled object. Additionally, we illustrated the applicability of the proposed solutions with examples. Then we included the resulting sets in the MPC framework. In the following chapter, concluding remarks with future perspective are given.

7 Conclusion and Future Perspectives

In this work, we designed a unified approach to handle path-following and force control problems simultaneously using predictive-optimization based control techniques. The proposed approach considers imposed constraints on states and inputs. Basically, we expanded the structure of predictive path-following control by force control law as a constraint inside the optimization problem. As shown, path-following allows to include indirect-force control and achieves convergence to the desired force. The receding horizon feature of the predictive scheme ensures a fast and stable convergence, since the augmented dynamic model (i.e. includes system dynamics, timing law and desired admittance) is used to predict the future evolution of the updated-path due to the force effect. Furthermore, the force along normal, tangential, and bi-normal direction with respect to the followed path, can be controlled by imposing desired admittance dynamics along corresponding directions. The proposed approach is applicable for free and constrained motion subject to known and unknown environments. It allows to compensate delays due to the sensor communication as shown in the simulation and experimental work.

The proposed approach is tested for different degrees of environmental stiffness. Challenges of the approach are, the nontrivial tuning of the parameters of the NMPC and the desired admittance dynamics, to account for effects of measurement delay and state estimation. While, in general, it is preferable to use a fast sampling rate, however, this leads to larger computation times which contradicts with the need for real-time implementation. This problem, was overcome using a real-time iteration algorithm with Gauss-Newton Hessian approximation.

It is important to avoid uncontrolled contact of the robot with the environment during an interaction. To achieve this, we proposed an approach to determine the set of initial conditions for which an uncontrolled collision occurs no matter which input is applied. After computing this set, which excluded from the set of initial states guarantees obstacle avoidance. To this end, we developed a set-based recursive procedure to compute the inevitable set, calculating the so called backward reachable set for one or multiple time steps. The derived methods can be used for a wide field of applications, spanning from direct force feedback control to man-machine interaction, robot-robot cooperative control.

7.1 Directions for Future Research

The proposed approach allows solving the path-following and force-feedback problems simultaneously while meeting the constraints on states and inputs. However, still several challenges remain that leave room for further research. Here, we refer to some challenges that can be considered in the future:

- Strictly proving stability and recursive feasibility.
- Testing the proposed approach on different robotics system with different tasks to show the reliability of the solution.
- Using robust NMPC methods to tackle the uncertainties arising from inaccurate models of the robot and the environment, or uncertain environment positions.
- Using adaptive admittance control methods to improve the performance of the force feedback process and to cope with a mismatch in the environment model. Furthermore, the tuning parameters problem of the admittance dynamics could be performed optimization-based.
- Since many environments are inconsistent, i.e. it varies from soft to stiff or has a rough surface, it is preferable to use online identification techniques to get good fitting model of the environment.

Bibliography

[1] M. Abu-Ayyad and R. Dubay. Real-time comparison of a number of predictive controllers. *ISA Transactions*, 46(3):411–418, 2007.

[2] V. Adetola and M. Guay. Nonlinear output feedback receding horizon control of sampled data systems. In *Proceedings of the American Control Conference*, volume 6, pages 4914–4919, 2003.

[3] A. P. Aguiar, J. P. Hespanha, and P. V. Kokotović. Path-following for nonminimum phase systems removes performance limitations. *IEEE Transactions on Automatic Control*, 50(2):234–239, 2005.

[4] A. P. Aguiar, J. P. Hespanha, et al. Trajectory-tracking and path-following of underactuated autonomous vehicles with parametric modeling uncertainty. *IEEE Transactions on Automatic Control*, 52(8):1362–1379, 2007.

[5] A. P. Aguiar, J. P. Hespanha, and P. V. Kokotović. Performance limitations in reference tracking and path following for nonlinear systems. *Automatica*, 44(3): 598–610, 2008.

[6] A. Albu-Schäffer and G. Hirzinger. Cartesian impedance control techniques for torque controlled light-weight robots. In *IEEE International Conference on Robotics and Automation, Proceedings. ICRA'02*, volume 1, pages 657–663, 2002.

[7] F. Allgöwer and A. Zheng. *Nonlinear Model Predictive Control*, volume 26. Birkhäuser, 2012.

[8] R. J. Anderson. Dynamic damping control: Implementation issues and simulation results. In *IEEE International Conference on Robotics and Automation*, pages 68–77, 1990.

[9] R. J. Anderson and M. W. Spong. Hybrid impedance control of robotic manipulators. *IEEE Journal of Robotics and Automation*, 4(5):549–556, 1988.

[10] S. Arimoto. *Control Theory of Nonlinear Mechanical Systems*. Oxford University Press, Inc., 1996.

[11] A. Banaszuk and J. Hauser. Feedback linearization of transverse dynamics for periodic orbits in r 3 with points of transverse controllability loss. *Systems & Control Letters*, 26(3):185–193, 1995.

[12] L. F. Baptista and J. M. S. da Costa. Force and position control of robotic manipulators: An experimental approach. In *6th IFAC Workshop on Algorithms and Architectures for real-time Control*, page 19, 2000.

[13] L. F. Baptista, J. M. Sousa, and J. M. G. Sá da Costa. Fuzzy predictive algorithms applied to real-time force control. *Control Engineering Practice*, 9(4): 411–423, 2001.

[14] V. Bargsten. Implementation of a model-based control scheme for a robotic arm. Master's thesis, Otto von Guericke Universität Magdeburg, 2012.

[15] G. Bastian et al. *Theory of Robot Control*. Springer, Berlin, 1996.

[16] H. H. Bauschke and J. M. Borwein. On projection algorithms for solving convex feasibility problems. *SIAM review*, 38(3):367–426, 1996.

[17] R. Bellman. Dynamic programming and lagrange multipliers. *Proceedings of the National Academy of Sciences*, 42(10):767–769, 1956.

[18] F. Blanchini. Set invariance in control. *Automatica*, 35(11):1747–1767, 1999.

[19] M. Buss, B. Hashimoto, and J. B. Moore. Dextrous hand grasping force optimization. *IEEE Transactions on Robotics and Automation*, 12(3):406–418, 1996.

[20] E. F. Camacho and C. B. Alba. *Model predictive control*. Springer Science & Business Media, 2013.

[21] E. F. Camacho and C. Bordons. Nonlinear model predictive control: An introductory review. In *Assessment and Future Directions of Nonlinear Model Predictive Control*, pages 1–16. Springer, 2007.

[22] H. Chen and F. Allgöwer. A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. In *European Control Conference (ECC)*, pages 1421–1426. IEEE, 1997.

[23] S. Chiaverini and L. Sciavicco. The parallel approach to force/position control of robotic manipulators. *IEEE Transactions on Robotics and Automation*, 9(4): 361–373, 1993.

[24] S. Chiaverini, B. Siciliano, and L. Villani. A survey of robot interaction control schemes with experimental comparison. *IEEE/ASME Transactions On Mechatronics*, 4(3):273–285, 1999.

[25] L. Consolini, M. Maggiore, C. Nielsen, and M. Tosques. Path following for the pvtol aircraft. *Automatica*, 46(8):1284–1296, 2010.

[26] O. Dahl and L. Nielsen. Torque-limited path following by online trajectory time scaling. *IEEE Transactions on Robotics and Automation*, 6(5):554–561, 1990.

[27] K. De Oliveira, L. Simone, and M. Morari. Contractive model predictive control for constrained nonlinear systems. *IEEE Transactions on Automatic Control*, 45 (6):1053–1071, 2000.

[28] C. C. de Wit, B. Siciliano, and G. Bastin. *Theory of Robot Control*. Springer Science & Business Media, 1996.

[29] M. Diehl, H. G. Bock, J. P. Schlöder, R. Findeisen, Z. Nagy, and F. Allgöwer.

Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. *Journal of Process Control*, 12(4):577–585, 2002.

[30] S. P. DiMaio, K. Hashtrudi-Zaad, and S. E. Salcudean. Optimal selection of manipulator impedance for contact tasks. In *IEEE International Conference on Robotics and Automation*, volume 5, pages 4795–4801, 2004.

[31] N. Diolaiti, C. Melchiorri, and S. Stramigioli. Contact impedance estimation for robotic systems. *IEEE Transactions on Robotics*, 21(5):925–935, 2005.

[32] V. Duchaine, S. Bouchard, and C. M. Gosselin. Computationally efficient predictive robot control. *IEEE/ASME Transactions on Mechatronics*, 12(5):570–578, 2007.

[33] D. Erol, V. Mallapragada, and N. Sarkar. Adaptable force control in robotic rehabilitation. In *IEEE International Workshop on Robot and Human Interactive Communication ROMAN*, pages 649–654, 2005.

[34] T. Faulwasser. *Optimization-based solutions to constrained trajectory-tracking and path-following problems*. PhD thesis, Otto-von-Guericke-Universität Magdeburg, 2013.

[35] T. Faulwasser and R. Findeisen. Nonlinear model predictive path-following control. In *Nonlinear Model Predictive Control*, pages 335–343. Springer, 2009.

[36] T. Faulwasser and R. Findeisen. Predictive path following without terminal constraints. In *Proc. of 20th Int. Symposium on Mathematical Theory of Networks and Systems (MTNS)*, 2012.

[37] T. Faulwasser, B. Kern, and R. Findeisen. Model predictive path-following for constrained nonlinear systems. In *Proceedings of the 48th IEEE Conference on Decision and Control, held jointly with the 28th Chinese Control Conference. CDC/CCC*, pages 8642–8647, 2009.

[38] 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*, 2016.

[39] G. Fernández. *Predictive context-based adaptive compliance for interaction control of robot manipulators*. PhD thesis, Bremen University, 2010.

[40] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl. qpoases: A parametric active-set algorithm for quadratic programming. *Mathematical Programming Computation*, 6(4):327–363, 2014.

[41] R. Findeisen. *Nonlinear model predictive control: a sampled data feedback perspective*. PhD thesis, University of Stuttgart, 2006.

[42] R. Findeisen and F. Allgöwer. An introduction to nonlinear model predictive control. In *21st Benelux Meeting on Systems and Control*, volume 11, pages

119–141, 2002.

[43] R. Findeisen and F. Allgöwer. Computational delay in nonlinear model predictive control. In *In Proceedings of the International Symposium on Advanced Control of Chemical Processes, ADCHEM'03*, pages 427–432, 2004.

[44] R. Findeisen, L. Imsland, F. Allgöwer, and B. A. Foss. Output feedback stabilization of constrained systems with nonlinear predictive control. *International Journal of Robust and Nonlinear Control*, 13(3-4):211–227, 2003.

[45] R. Findeisen, L. Imsland, F. Allgöwer, and B. A. Foss. State and output feedback nonlinear model predictive control: An overview. *European Journal of Control*, 9(2):190–206, 2003.

[46] S. Flixeder, T. Gluck, M. Bock, and A. Kugi. Combined path following and compliance control with application to a biaxial gantry robot. In *IEEE Conference on Control Applications (CCA)*, pages 796–801, 2014.

[47] F.A.C.C. Fontes. Overview of nonlinear model predictive control schemes leading to stability. 2000.

[48] F.A.C.C. Fontes. A general framework to design stabilizing nonlinear model predictive controllers. *Systems & Control Letters*, 42(2):127–143, 2001.

[49] P. J. From, J. T. Gravdahl, T. Lillehagen, and P. Abbeel. Motion planning and control of robotic manipulators on seaborne platforms. *Control Engineering Practice*, 19(8):809–819, 2011.

[50] M. Fujiwara, Z. K. Nagy, J. W. Chew, and R. D. Braatz. First-principles and direct design approaches for the control of pharmaceutical crystallization. *Journal of Process Control*, 15(5):493–504, 2005.

[51] C. E. Garcia, D. M. Prett, and M. Morari. Model predictive control: theory and practice survey. *Automatica*, 25(3):335–348, 1989.

[52] R. Ghaemi, S. Oh, and J. Sun. Path following of a model ship using model predictive control with experimental verification. In *American Control Conference (ACC)*, pages 5236–5241. IEEE, 2010.

[53] P. B. Goldsmith, B. A. Francis, and A. A. Goldenberg. Stability of hybrid position/force control applied to manipulators with flexible joints. *International Journal of Robotics and Automation*, 14(4):146–160, 1999.

[54] L. Grüne and J. Pannek. *Nonlinear Model Predictive Control*. Springer, 2011.

[55] J. P. Hespanha, P. V. Kokotović, et al. Path-following for nonminimum phase systems removes performance limitations. *IEEE Transactions on Automatic Control*, 50(2):234–239, 2005.

[56] A. Hladio, C. Nielsen, and D. Wang. Path following controller design for a class of mechanical systems. In *18th World Congress of the International Federation of Automatic Control*, page 10, 2011.

[57] N. Hogan. Impedance control: An approach to manipulation: Part II-Implementation. *Journal of Dynamic Systems, Measurement, and Control*, 107 (1):8–16, 1985.

[58] K. S. Holkar and L. M. Waghmare. An overview of model predictive control. *International Journal of Control and Automation*, 3(4):47–63, 2010.

[59] B. Houska, H. J. Ferreau, and M. Diehl. Acado toolkit an open source framework for automatic control and dynamic optimization. *Optimal Control Applications and Methods*, 32(3):298–312, 2011.

[60] B. Houska, H. J. Ferreau, and M. Diehl. An auto-generated real-time iteration algorithm for nonlinear mpc in the microsecond range. *Automatica*, 47(10):2279–2285, 2011.

[61] L. Imsland, R. Findeisen, E. Bullinger, F. Allgöwer, and B. A. Foss. A note on stability, robustness and performance of output feedback nonlinear model predictive control. *Journal of Process Control*, 13(7):633–644, 2003.

[62] A. Isidori. *Nonlinear Control Systems*. Springer Science & Business Media, 1995.

[63] S. Ivaldi, M. Fumagalli, F. Nori, M. Baglietto, G. Metta, and G. Sandini. Approximate optimal control for reaching and trajectory planning in a humanoid robot. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1290–1296, 2010.

[64] A. Jadbabaie, J. Yu, and J. Hauser. Unconstrained receding-horizon control of nonlinear systems. *IEEE Transactions on Automatic Control*, 46(5):776–783, 2001.

[65] A. Jain, M. D. Killpack, A. Edsinger, and C. C. Kemp. Reaching in clutter with whole-arm tactile sensing. *The International Journal of Robotics Research*, 2013.

[66] S. Jung and T. C. Hsia. Neural network impedance force control of robot manipulator. *IEEE Transactions on Industrial Electronics*, 45(3):451–461, 1998.

[67] S. Jung, T. C. Hsia, and R. G. Bonitz. Force tracking impedance control of robot manipulators under unknown environment. *IEEE Transactions on Control Systems Technology*, 12(3):474–483, 2004.

[68] S. A. Keerthi and E. 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.

[69] E. C. Kerrigan. *Robust constraint satisfaction: Invariant sets and predictive control*. PhD thesis, Cambridge University, 2000.

[70] O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal of Robotics and Automation*, 3(1):43–53, 1987.

[71] M. D. Killpack, A. Kapusta, and C. C. Kemp. Model predictive control for fast reaching in clutter. *Autonomous Robots*, pages 1–24, 2015.

[72] P. Kühl, M. Diehl, T. Kraus, J. P Schlöder, and H. G. Bock. A real-time algorithm for moving horizon state and parameter estimation. *Computers & Chemical Engineering*, 35(1):71–83, 2011.

[73] P. Kulchenko and E. Todorov. First-exit model predictive control of fast discontinuous dynamics: Application to ball bouncing. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2144–2151, 2011.

[74] D. Lam, C. Manzie, and M. Good. Model predictive contouring control. In *49th IEEE Conference on Decision and Control (CDC)*, pages 6137–6142, 2010.

[75] J. Liu, D. Muñoz de la Peña, P. D. Christofides, and J. F. Davis. Lyapunov-based model predictive control of nonlinear systems subject to time-varying measurement delays. *International Journal of Adaptive Control and Signal Processing*, 23(8):788–807, 2009.

[76] J. Lofberg. Yalmip: A toolbox for modeling and optimization in matlab. In *International Symposium on Computer Aided Control Systems Design*, pages 284–289, 2004.

[77] W. S. Lu and Q. H. Meng. Impedance control with adaptation for robotic manipulations. *IEEE Transactions on Robotics and Automation*, 7(3):408–415, 1991.

[78] Z. Lu and A. A. Goldenberg. Robust impedance control and force regulation: Theory and experiments. *The International Journal of Robotics Research*, 14(3): 225–254, 1995.

[79] J. M. Maciejowski. *Predictive Control: with Constraints*. Pearson education, 2002.

[80] L. Magni, D. M. Raimondo, and F. Allgöwer. *Nonlinear Model Predictive Control*. Springer, 2009.

[81] I. R. Manchester, U. Mettin, F. Iida, and R. Tedrake. Stable dynamic walking over uneven terrain. *The International Journal of Robotics Research*, 2011.

[82] M. Matinfar. Optimization-based robot compliance control: Geometric and linear quadratic approaches. *The International Journal of Robotics Research*, 24 (8):645–656, 2005.

[83] D. Q. Mayne and H. Michalska. Receding horizon control of nonlinear systems. *IEEE Transactions on Automatic Control*, 35(7):814–824, Jul 1990.

[84] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

[85] H. Michalska and D. Q. Mayne. Robust receding horizon control of constrained

nonlinear systems. *IEEE Transactions on Automatic Control*, 38(11):1623–1633, 1993.

[86] M. Morari and J. H. Lee. Model predictive control: past, present and future. *Computers & Chemical Engineering*, 23(4):667–682, 1999.

[87] L. Moreira, T. I. Fossen, and C. G. Soares. Path following control system for a tanker ship model. *Ocean Engineering*, 34(14):2074–2085, 2007.

[88] C. Nielsen and M. Maggiore. Maneuver regulation via transverse feedback linearization: Theory and examples. In *Proceedings of the IFAC Symposium on Nonlinear Control Systems (NOLCOS), Stuttgart, Germany*, pages 59–66, 2004.

[89] C. Nielsen, C. Fulford, and M. Maggiore. Path following using transverse feedback linearization: Application to a maglev positioning system. *Automatica*, 46 (3):585–590, 2010.

[90] H. Nijmeijer and V. D. Schaft. *Nonlinear dynamical control systems*. Springer-Verlag New York, 1990.

[91] C. Pasluosta, H. Tims, and L. Chiu. Slippage sensory feedback and nonlinear force control system for a low-cost prosthetic hand. *American Journal of Biomedical Sciences*, 1(4):295–302, 2009.

[92] A. A. Patwardhan, J. B. Rawlings, and T. F. Edgar. Nonlinear model predictive control. *Chemical Engineering Communications*, 87(1):123–141, 1990.

[93] E. Polak and T. Yang. Moving horizon control of linear systems with input saturation and plant uncertainty part 1. robustness. *International Journal of Control*, 58(3):613–638, 1993.

[94] R. De J. Portillo-velez, A. Rodriguez-angeles, and C. A. Cruz-villar. Optimization-based reactive force control for robot grasping tasks. pages 186–191, 2012.

[95] R. De J. Portillo-Vélez, A. Rodriguez-Angeles, and C. A. Cruz-Villar. An optimization-based impedance approach for robot force regulation with prescribed force limits. *Mathematical Problems in Engineering*, 2015.

[96] S. J. Qin and T. A. Badgwell. An overview of nonlinear model predictive control applications. In *Nonlinear Model Predictive Control*, pages 369–392. Springer, 2000.

[97] J. Richalet. Industrial applications of model based predictive control. *Automatica*, 29(5):1251–1274, 1993.

[98] A. Richards and J. How. Mixed-integer programming for control. In *Proceedings of the American Control Conference*, pages 2676–2683, 2005.

[99] A. Richards and J. How. Mixed-integer programming for control. In *IEEE Proceedings of the American Control Conference*, pages 2676–2683, 2005.

[100] B. J. P. Roset, M. Lazar, W. P. M. H Heemels, and H. Nijmeijer. A stabilizing output based nonlinear model predictive control. page 4627–4632, 2006.

[101] P. Rumschinski, S. Borchers, S. Bosio, R. Weismantel, and R. Findeisen. Set-base dynamical parameter estimation and model invalidation for biochemical reaction networks. *BMC Systems Biology*, 4(1):69, 2010.

[102] R. J. Schilling. *Fundamentals of Robotics: Analysis and Control*. Simon & Schuster Trade, 1st edition, 1996.

[103] T. Schouwenaars, B. De Moor, E. Feron, and J. How. Mixed integer programming for multi-vehicle path planning. In *Proceedings of European Control Conference*, pages 2603–2608, 2001.

[104] G. Schreiber, A. Stemmer, and R. Bischoff. The fast research interface for the kuka lightweight robot. In *IEEE Workshop on Innovative Robot Control Architectures for Demanding (Research) Applications How to Modify and Enhance Commercial Controllers (ICRA)*, pages 15–21. Citeseer, 2010.

[105] P. O. M. Scokaert, D. Q. Mayne, and J. B. Rawlings. Suboptimal model predictive control (feasibility implies stability). *IEEE Transactions on Automatic Control*, 44(3):648–654, 1999.

[106] H. Seraji and R. Colbaugh. Adaptive force-based impedance control. In *International Conference on Intelligent Robots and Systems Proceedings of the IEEE/RSJ*, volume 3, pages 1537–1544, 1993.

[107] H. Seraji and R. Colbaugh. Force tracking in impedance control. In *IEEE International Conference on Robotics and Automation*, pages 499–506, 1993.

[108] H. Seraji and R. Colbaugh. Force tracking in impedance control. *The International Journal of Robotics Research*, 16(1):97–117, 1997.

[109] K. G. Shin and N. D. McKay. Minimum-time control of robotic manipulators with geometric path constraints. *IEEE Transactions on Automatic Control*, 30 (6):531–541, 1985.

[110] B. Siciliano and L. Villani. *Robot Force Control*, volume 540. Springer Science & Business Media, 2012.

[111] R. Skjetne, T. I. Fossen, and P. V. Kokotović. Robust output maneuvering for a class of nonlinear systems. *Automatica*, 40(3):373–383, 2004.

[112] P. Slaets, J. Rutgeerts, K. Gadeyne, T. Lefebvre, H. Bruyninckx, and J. De Schutter. Construction of a geometric 3-d model from sensor measurements collected during compliant motion. In *Experimental Robotics IX*, pages 571–580. Springer, 2006.

[113] J. J. E. Slotine and H. S. Yang. Improving the efficiency of time-optimal path-following algorithms. *IEEE Transactions on Robotics and Automation*, 5(1): 118–124, 1989.

[114] S. Streif, A. Savchenko, P. Rumschinski, S. Borchers, and R. Findeisen. Admit: A toolbox for guaranteed model invalidation, estimation and qualitative–quantitative modeling. *Bioinformatics*, 28(9):1290–1291, 2012.

[115] D. Surdilovic and J. Kirchhof. A new position based force/impedance control for industrial robots. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 629–634, 1996.

[116] M. Sznaier, R. Suárez, and J. Cloutier. Suboptimal control of constrained nonlinear systems via receding horizon constrained control lyapunov functions. *International Journal of Robust and Nonlinear Control*, 13(3-4):247–259, 2003.

[117] Y. Tassa, T. Erez, and E. Todorov. Synthesis and stabilization of complex behaviors through online trajectory optimization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4906–4913, 2012.

[118] Barrett Technology. *The Barrett 6-Axis Force/Torque Sensor*, (accessed May 23, 2016). URL `http://www.barrett.com`.

[119] G. Tonietti, R. Schiavi, and A. Bicchi. Optimal mechanical/control design for safe and fast robotics. In *Experimental Robotics IX*, pages 311–320. Springer, 2006.

[120] F. Towhidkhah, R. E. Gander, and H. C. Wood. Model predictive impedance control: A model for joint movement. *Journal of Motor Behavior*, 29(3):209–222, 1997.

[121] L. Villani and J. De Schutter. *Force Control*. Springer, 2008.

[122] R. Volpe and P. Khosla. The equivalence of second-order impedance control and proportional gain explicit force control. *The International Journal of Robotics Research*, 14(6):574–589, 1995.

[123] M. Vukov, A. Domahidi, H. J. Ferreau, M. Morari, and M. Diehl. Auto-generated algorithms for nonlinear model predictive control on long and on short horizons. In *IEEE 52nd Annual Conference on Decision and Control (CDC)*, pages 5113–5118, 2013.

[124] P. B. Wieber. Trajectory free linear model predictive control for stable walking in the presence of strong perturbations. In *6th IEEE-RAS International Conference on Humanoid Robots*, pages 137–142, 2006.

[125] T. Winiarski and A. Woźniak. Indirect force control development procedure. *Robotica*, 31(03):465–478, 2013.

[126] B. Yao and M. Tomizuka. Robust adaptive constrained motion and force control of manipulators with guaranteed transient performance. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 893–898, 1995.

[127] S. Yu, X. Li, H. Chen, and F. Allgöwer. Nonlinear model predictive control for path following problems. In *Nonlinear Model Predictive Control*, volume 4, pages

145–150, 2012.

[128] G. Zeng and A. Hemami. An overview of robot force control. *Robotica*, 15(05): 473–482, 1997.

A Mathematical definitions

Definition A.1 (Positive Definite Matrix)
A square, $n \times n$ symmetric matrix A is positive definite if, for all non-zero column vector $x \in \mathbb{R}^n$

$$x^T A x > 0. \tag{A.1}$$

Definition A.2 (Semi-Positive Definite Matrix)
A is called positive-semidefinite (or nonnegative-definite) if, for all column vector $x \in \mathbb{R}^n$

$$x^T A x \geq 0. \tag{A.2}$$

Definition A.3 (Negative Definite Matrix)
A matrix A is said to be negative-definite if, for all non-zero column vector $x \in \mathbb{R}^n$

$$x^T A x < 0. \tag{A.3}$$

While it is called negative-semidefinite if, for all column vector $x \in \mathbb{R}^n$

$$x^T A x \leq 0. \tag{A.4}$$

Definition A.4 ($\mathcal{K}$-function)
A continuous function $\alpha \in [0, a) \to [0, \infty)$ is of class $\mathcal{K}$ if it is strictly increasing and $\alpha(0) = 0$. It belongs to $\mathcal{K}_\infty$ if $a = \infty \ \wedge \ \alpha(r) \to \infty$ when $r \to \infty$.

Definition A.5 ($\mathcal{L}$-function)
A continuous function $\beta \in [0, \infty) \to [0, \infty)$ is of class $\mathcal{L}$ if it is monotonically decreasing and $\lim_{t\to\infty} \beta(t) = 0$.

Definition A.6 ($\mathcal{KL}$-function)
A function is class $\mathcal{K}$ with respect to the first argument and is class $\mathcal{L}$ with respect to the second argument belongs to class $\mathcal{KL}$.

Definition A.7 (Submanifolds) *Suppose that $F : N \to M$ be a smooth mapping of manifolds [62]:*

1. *F is an immersion if $rank(F) = dim(N)$ for all $x \in N$,*

2. *F is an univalent immersion if F is an immersion and is injective,*

3. *F is an embedding if F is an univalent immersion and the topology induced on $F(N)$ by the one of N coincides with the topology of $F(N)$ as a subset of M.*

Theorem A.1 (Stability of sampled-data NMPC) [45]
Suppose that
(a) the terminal region $\Omega \subseteq \mathcal{X}$ is closed with $0 \in \Omega$ and that the terminal penalty $E(x) \in C^1$ is positive semi-definite,
(b) the terminal region and terminal penalty term are chosen such that $\forall x \in \Omega$ there exists an admissible input $u_\Omega : [0, \delta] \to \mathcal{U}$ such that $x(\tau) \in \Omega \ \forall \tau \in [0, \delta]$ and

$$\frac{\partial E}{\partial x} f(x(\tau), u_\Omega(\tau)) + L(x(\tau), u_\Omega(\tau)) \leq 0, \quad \forall \tau \in [0, \delta] \tag{A.5}$$

(c) the NMPC open-loop optimal control problem is feasible at $t = 0$.
Then in the closed-loop system (2.1a) *with* (2.5) *$\lim_{t\to\infty} x(t) = 0$, and the region of attraction $\Re$ consists of the states for which an admissible input exists.*

Proof The detailed proof see [45]. ∎

Theorem A.2 (Sufficient condition of local unconstrained path followability [34])
Consider system (2.21)*, a path $\mathcal{P}$* (2.22) *and Assumptions 2.4−2.6 hold. Suppose $\Phi : \mathcal{X}_0 \times \hat{\mathcal{X}} \mapsto \mathbb{R}^n \times \mathbb{R}^{\hat{r}+1}$ is a local diffeomorphism mapping to a transverse normal form, and $\Gamma, \hat{\mathcal{X}}$ are defined in* (2.35) *and* (2.27)*.*
Then for any $(x_0, \hat{x}_0) \in \mathcal{X}_0 \times \hat{\mathcal{X}}$ with

$$\Phi(x_0, \hat{x}_0) \in \Gamma \quad \textit{and} \quad \hat{x}_0 \in \textit{int}\hat{\mathcal{X}}, \tag{A.6}$$

the path $\mathcal{P}$ is locally exactly followable by system (2.21) *s.t. $\dot{\theta} > 0$ holds.*

Proof The detailed proof is given in [34]. ∎

B Big-M Method

As an example consider a rectangular object within a two-dimensional workspace. To guarantee obstacle avoidance, at least one of the following constraints must be satisfied, see Figure B.1.

$$x \leq x_{min} \tag{B.1a}$$
$$x \geq x_{max} \tag{B.1b}$$
$$y \leq y_{min} \tag{B.1c}$$
$$y \geq y_{max} \tag{B.1d}$$

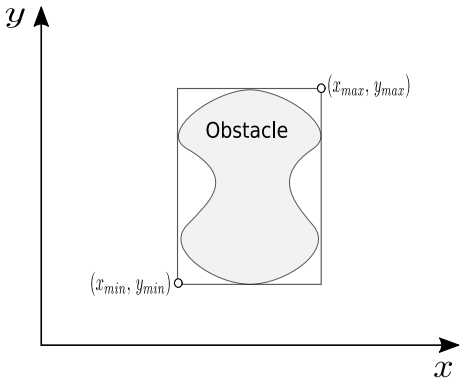


Figure B.1: Simple example for obstacle avoidance.

Since at least one constraint is required to be satisfied at all times, up to three constraints can be relaxed at any time. These modified constraints take the following form (b_i are binary variables, M is a constant) [99].

$$x \leq x_{min} + Mb_1, \tag{B.2a}$$
$$-x \leq x_{max} + Mb_2, \tag{B.2b}$$
$$y \leq y_{min} + Mb_3, \tag{B.2c}$$
$$-y \leq y_{max} + Mb_4, \tag{B.2d}$$
$$\sum_{i=1}^{4} b_i \leq 3 \tag{B.2e}$$

It is important that M is chosen to be larger than the object, so that when an individual constraint is relaxed, it makes the entire workspace available [99]. Such integer constraints can be easily integrated into predictive control approaches, leading to mixed integer MPC formulations of obstacle avoidance. However, these methods do not in general guarantee repeated feasibility, especially for insufficiently long prediction horizons, i.e. if the controller cannot look around the obstacle.

C Tables

Table C.1: Denavit-Hartenberg parameters of KUKA LWR IV [14]

Joint	a [m]	d [m]	α [rad]
1	0	0.31	$\pi/2$
2	0	0	$-\pi/2$
3	0	0.4	$-\pi/2$
4	0	0	$\pi/2$
5	0	0.39	$\pi/2$
6	0	0.078	$-\pi/2$
7	0	0	0

Table C.2: **Comparison among different environments**

Parameter name	**Path-following and Force Control**
Length of horizon N Sampling period δ No. of iterations per N	30 [ms] 3 [ms] 10
$W = diag(w_1, w_2, w_3)$	$w_1 = 5 \times 10^{10}, w_2 = 5 \times 10^{10}, w_3 = 10$
$R = diag(r_1, r_2, r_3)$	$r_1 = 0.5,\ r_2 = 0.5,\ r_3 = 1 \times 10^{-4}$
$X = [\overline{x},\ \underline{x}]; \underline{x} = -\overline{x} = -(\overline{q}_2, \overline{q}_4, \overline{\dot{q}}_2, \overline{\dot{q}}_4)^T$	$\overline{q}_2 = \overline{q}_4 = 2.09$ [rad]; $\overline{\dot{q}}_2 = 0.5$ [rad/s], $\overline{\dot{q}}_4 = 0.5$ [rad/s]
$U = [\overline{u}, \underline{u}],\ \underline{u} = -\overline{u} = -(\overline{\tau}_{e2}, \overline{\tau}_{e4})^T$	$\overline{\tau}_{e2} = 6$ [Nm], $\overline{\tau}_{e4} = 3$ [Nm]
$\hat{x}_2$ $\hat{U} = [\overline{\hat{u}}, \underline{\hat{u}}],$	$0 \leq \hat{x}_2 \leq 0.4$ $\overline{\hat{u}} = 10,\ \underline{\hat{u}} = -10$
m	2 [kg] for $k_e \in [1000 - 10000]$ and 1 [kg] for $k_e \in [50000 - 500000]$
b	$6.5\ \sqrt{k_e\, m}$ [N.s/m]
k_d	0 [N/m]
Circular-path:	$r_c = 0.1,\ \ y_{1c} = 0.4,\ y_{2c} = 0.55$

Table C.3: Parameters of **simulation studies**

Parameter name	**Path-following**	**Path-following and Force Control**
Length of horizon N Sampling period δ No. of iterations per one horizon	30 [ms] 3 [ms] 10	30 [ms] 3 [ms] 10
$W = diag(w_1, w_2, w_3)$	$w_1 = 5 \times 10^8, w_2 = 5 \times 10^8, w_3 = 10$	$w_1 = 5 \times 10^10, w_2 = 5 \times 10^10, w_3 = 10$
$R = diag(r_1, r_2, r_3)$	$r_1 = 0.5, r_2 = 0.5, r_3 = 1 \times 10^{-4}$	$r_1 = 0.5, r_2 = 0.5, r_3 = 1 \times 10^{-4}$
$X = [\overline{x},\ \underline{x}]; \underline{x} = -\overline{x} = -(\overline{q}_2, \overline{q}_4, \overline{\dot{q}}_2, \overline{\dot{q}}_4)^T$	$\overline{q}_2 = \overline{q}_4 = 2.09\,[\text{rad}]$; $\overline{\dot{q}}_2 = 0.5$ [rad/s], $\overline{\dot{q}}_4 = 0.5$ [rad/s]	$\overline{q}_2 = \overline{q}_4 = 2.09\,[\text{rad}]$; $\overline{\dot{q}}_2 = 0.5$ [rad/s], $\overline{\dot{q}}_4 = 0.5$ [rad/s]
$U = [\overline{u}, \underline{u}]$, $\underline{u} = -\overline{u} = -(\overline{\tau}_{e1}, \overline{\tau}_{e2})^T$	$\overline{\tau}_{e2} = 6$ [Nm], $\overline{\tau}_{e4} = 3$ [Nm]	$\overline{\tau}_{e2} = 6$ [Nm], $\overline{\tau}_{e4} = 3$ [Nm]
$\dot{\theta}$ $\hat{U} = [\overline{\hat{u}}, \underline{\hat{u}}]$, $\underline{\hat{u}} = -\overline{\hat{u}}$	$0 \leq \dot{\theta} \leq 0.4$ $\overline{\hat{u}} = 10$	$0 \leq \dot{\theta} \leq 0.4$ $\overline{\hat{u}} = 10$
M_d B_d K_d	– – –	1 [kg] $6.5\sqrt{k_e\, m}$ [N.s/m] 0 [N/m]
Sine-path:	$a = 0.01$, $y_{0_1} = 0.45$, $y_{0_2} = 0.4$; $y_{f_1} = 0.45$, $y_{f_2} = 0.55$	$a = 0.01$, $y_{0_1} = 0.45$, $y_{0_2} = 0.4$; $y_{f_1} = 0.45$, $y_{f_2} = 0.55$
Circular-path:	$r_c = 0.1$, $y_{1c} = 0.4$, $y_{2c} = 0.55$	$r_c = 0.1$, $y_{1c} = 0.4$, $y_{2c} = 0.55$

Table C.4: **Parameters of the Experimental work**

Parameter name	**Nominal admittance**	**Zero-stiffness admittance**
Length of horizon N Sampling period δ No. of iterations per one horizon	100 [ms] 10 [ms] 10	100 [ms] 10 [ms] 10
$W = diag(w_1, w_2, w_3)$	$w_1 = 4 \times 10^7, w_2 = 4 \times 10^7, w_3 = 3 \times 10^5$	$w_1 = 7 \times 10^7, w_2 = 7 \times 10^7, w_3 = 3 \times 10^5$
$R = diag(r_1, r_2, r_3)$	$r_1 = 5, r_2 = 5, r_3 = 10^{-6}$	$r_1 = 5, r_2 = 5, r_3 = 10^{-6}$
$X = [\overline{x},\ \underline{x}]; \underline{x} = -\overline{x} = -(\overline{q}_2, \overline{q}_4, \overline{\dot{q}}_2, \overline{\dot{q}}_4)^T$	$\overline{q}_2 = \overline{q}_4 = 2.09$ [rad]; $\overline{\dot{q}}_2 = 0.5$ [rad/s], $\overline{\dot{q}}_4 = 0.5$ [rad/s]	$\overline{q}_2 = \overline{q}_4 = 2.09$ [rad]; $\overline{\dot{q}}_2 = 0.5$ [rad/s], $\overline{\dot{q}}_4 = 0.5$ [rad/s]
$U = [\overline{u}, \underline{u}]$, $\underline{u} = -\overline{u} = -(\overline{\tau}_{e2}, \overline{\tau}_{e4})^T$	$\overline{\tau}_{e2} = 10$ [Nm], $\overline{\tau}_{e4} = 6$ [Nm]	$\overline{\tau}_{e2} = 6$ [Nm], $\overline{\tau}_{e4} = 6$ [Nm]
$\hat{x}_2$ $\hat{U} = [\overline{\hat{u}}, \underline{\hat{u}}]$,	$-10^-14 \leq \hat{x}_2 \leq 0.1$ $\overline{\hat{u}} = 0.05$, $\underline{\hat{u}} = -10$	$-10^-14 \leq \hat{x}_2 \leq 0.05$ $\overline{\hat{u}} = 0.05$, $\underline{\hat{u}} = -10$
M_d B_d K_d K_e	1 [kg] 500 [N.s/m] 5000 [N/m] 1000 [N/m]	0.5 $500[N.s/m]$ 0 –
Circular-path:	$r_c = 0.1$, $y_{1c} = 0.6915$, $y_{2c} = 0.4278$	$r_c = 0.1$, $y_{1c} = 0.6915$, $y_{2c} = 0.4278$

D The second and third parts of the proof of theorem 2.1

Before start second part of the proof, which is required that for any $x(t_i) \in_{c_1} \hat{x}(t_i) \in \psi_{c_2}$ after convergence of the observer. Based on Assumption 2.8 there always can find observer parameters such that after $\delta_{max} k_{obs}$ the observer error will be smaller than any desired e_max. If we need that

$$\alpha_V(e_{max}) \leq c_2 - c_1 \tag{D.1}$$

It is ensured that $\hat{x}(t_i) \in \psi_{c2}$ if $x(t_i) \in \psi_{c1}$.

Second part: (decrease of the value function after the convergence of the observer and finite time convergence of $\psi_{a/2}$): it is assumed that $x(t_i) \in \psi_{c1}$ and for simplicity, $u_{\hat{x}}$ represents the optimal input resulting from $\hat{x}(t_i)$ and u_x for the input resulting form the real state $x(t_i)$. Also, $x_i = x(t_i)$ and $\hat{x}_i = \hat{x}(t_i)$. According on the first part of the proof, when $x_i \in \psi_{c1}$ implies that $\hat{x}_i \in \psi_{c2}$ and

$$x(\tau) \in_c, \, x(\tau; \hat{x}_i, u_{\hat{x}}) \in \psi, \; x(\tau;\, x_i,\, u_{\hat{x}}) \in \psi_c \forall \tau \in [t_i, t_{i+1}).$$

Based on these conditions the following equality is valid:

$$\begin{aligned} V(x(\tau; x_i, u_{\hat{x}})) - V(x_i) = V(x(\tau; x_i, u_{\hat{x}})) - V(x(\tau; \hat{x}_i, u_{\hat{x}})) + V(x(\tau; \hat{x}_i, u_{\hat{x}})) \\ -V(\hat{x}_i) + V(\hat{x}_i) - V(x_i). \end{aligned} \tag{D.2}$$

It can bound the last two terms since V is uniformly continuous in compact subsets of $\Re \supset \psi_c$. As well as, the third and fourth term start from the same $\hat{x}_i$, and the first term can be bound by α_V:

$$V(x(\tau; x_i, u_{\hat{x}})) - V(x_i) \leq \alpha_V(e^{L_{fx}(\tau - t_i)} \, \|\hat{x}_i - x_i\|) - \int_{t_i}^{\tau} F(x(s; \hat{x}_i, u_{\hat{x}}))\, ds + \alpha_V(\|\hat{x}_i - x_i\|). \tag{D.3}$$

Where, the Gronwall-Bellman lemma is used to set the upper bound for $\|x(\tau; x_i, u_{\hat{x}}) - x(\tau; \hat{x}_i, u_{\hat{x}})\|$

Now, assuming that $x_i \notin \psi_{a/2}$ and that

$$\alpha_V(e_{max}) \leq \frac{\alpha}{4}, \tag{D.4}$$

then it implies that $\hat{x}_i \notin \psi_{a/4}$. Thus we get from (**??**)onsidering Fact 1 that

$$V(x(\delta; x_i, u_{\hat{x}})) - V(x_i) \leq -V_{\min}\left(c, \frac{\alpha}{4}, \delta\right) + \alpha_V\left(e^{L_{fx}\delta}\left\|\hat{x}_i - x_i\right\|\right) + \alpha_V(\|\hat{x}_i - x_i\|). \quad \text{(D.5)}$$

The right-hand side needs to be less than zero to guarantee the decreasing of x from sampling time to sampling time through the level sets, and to converge to the set $\psi_{a/2}$ within finite time. To achieve this, it is required that the observer parameters are designed such that:

$$\alpha_V\left(e^{L_{fx}\delta}\left\|\hat{x}_i - x_i\right\|\right) + \alpha_V(\|\hat{x}_i - x_i\|) - V_{\min}\left(c, \frac{\alpha}{4}, \delta\right) \leq -V_{\min}\left(c, \frac{\alpha}{4}, \delta\right) + \frac{1}{2}V_{\min}\left(c, \frac{\alpha}{4}, \delta\right). \quad \text{(D.6)}$$

If we choose the observer parameters as follows:

$$\alpha_V\left(e^{L_{fx}\delta}e_{\max}\right) + \alpha_V(e_{\max}) \leq \frac{1}{2}V_{\min}\left(c, \frac{\alpha}{4}, \delta\right) \quad \text{and} \quad \alpha_V(e_{\max}) \leq \frac{\alpha}{4} \quad \text{(D.7)}$$

the finite time convergence can be achieved starting from any point in ψ_b to the set $\psi_{a/2}$.

Third part: $(x(t_{i+1}) \in \psi_a \forall x(t_i) \in \psi_{a/2})$: If $x(t_i) \in \psi_{a/2}$ equation (D.3) is still fulfilled. Ignoring the integral part on the right we get:

$$V(x(\tau; x_i, u_{\hat{x}})) - V(x_i) \leq \alpha_V(e^{L_{fx}(\tau - t_i)}\left\|\hat{x}_i - x_i\right\|) + \alpha_V(\|\hat{x}_i - x_i\|). \quad \text{(D.8)}$$

Assuming that

$$\alpha_V\left(e^{L_{fx}\delta}e_{\max}\right) + \alpha_V(e_{\max}) \leq \frac{\alpha}{2}. \quad \text{(D.9)}$$

Then $x(t_{i+1}) \in \psi_a \forall x(t_i) \in \psi_{a/2}$. By gathering all last three steps, we get the theorem if

$$\delta_{\max} \leq \min\left\{t_{b-c_1}/k_{obs,\, t_{c_2-c}}\right\}. \quad \text{(D.10)}$$

and the observer error $e_{\max}$ is chosen such that:

$$\alpha_V\left(e^{L_{fx}\delta}e_{\max}\right) + \alpha_V(e_{\max}) \leq \min\left\{\frac{1}{2}V_{\text{mni}}\left(c, \frac{\alpha}{4}, \delta\right), \frac{\alpha}{4}\right\}. \quad \text{(D.11)}$$