THÈSE POUR OBTENIR LE GRADE DE DOCTEUR
DE L'UNIVERSITÉ DE MONTPELLIER

En Systèmes Automatiques et Micro-Électroniques

École doctorale : Information, Structures, Systèmes

Unité de recherche : LIRMM

Model Predictive Tracking Control of Cable-Driven Parallel Robots: From Concept to Real-Time Validation

Présentée par João Cavalcanti Santos

Le 5 octobre 2020

Sous la direction de Marc Gouttefarde

Devant le jury composé de

Stéphane Caro, Directeur de Recherche CNRS, LS2N, Nantes	Président du jury
Jacques Gangloff, Professeur des Universités, Université de Strasbourg, ICube	Rapporteur
Ionela Prodan, Maître de Conférence, INP, Grenoble	Examinateur
Lars Grüne, Professor of Applied Mathematics, University of Bayreuth	Examinateur
Ahmed Chemori, Chargé de Recherche CNRS, LIRMM	Co-encadrant

UNIVERSITÉ DE MONTPELLIER

Contents

List of Figures

List of Tables

Acknowledgments

Firstly, I'd like to thank my thesis director, Marc Gouttefarde, for his supervision. By the end of this doctorate, I find myself a better professional and a better person. Much of these professional and personal growths were motivated and guided by him. I also thank my co-supervisor, Ahmed Chemori, for his support.

From the technical point of view, I relied on the support of Pierre-Elie Herve, Benjamin Navarro, Robin Passama, Arnaud Meline and Olivier Tempier. As for most of the members of our department, their immense background was crucial for the development of my works.

It was a pleasure working with the members of the Hephaestus project. In particular, Hussein Hussein introduced several interesting contributions with which I had the honor of collaborating. Equally fruitful was the time working with Jean Baptiste Izard, Jose Gorrotxategi, Christoph Martin, Ibon Eskudero Gurrutxaga, Kepa Iturralde and Marc Fabritius.

During my stay in France, I also learned to admire people who know how to deal efficiently with paperwork. Particularly, besides being a good friend, Nicolas Serrurier greatly helped me on these endeavors. I also express my gratitude to Cécile Lukasik, Gwladys Imbert, Anne-Laure Irastorza and Najima Maatoug.

I greatly thank the colleagues and friends that I met in Montpellier: Vinayak, Jr, Thibault, Karine, Virginie, Maxime, Lucas, Maxence, Guillaume, Benjamin, Laure and Nicolas, Mouna and Ghina. We could laugh, share our achievements, discuss about engineering, math, politics and, occasionally, less serious things. And these occasions were an important part of my doctorate, even though they are not detailed on this manuscript for confidential reasons.

The same applies for my Brazilian friends Dogra, Gualazzi, Fofão, Duque, Abu, among others. Even though I've been mostly away from them, a great part of what I could do within these last three years was thanks to what learned with them and their support. Obrigado pela amizade!

Minha família. Sinto, sinceramente, que tudo o que vivi (este doutorado incluso) é pura e simplesmente uma consequência do que vivenciei com meus pais e minha irmã. Essas três pessoas, adicionando Tales, Gael, Teresa e Mariângela, são protagonistas nesses agradecimentos.

It is funny to realize that the last person to thank could be cited in many other

paragraphs on these acknowledgments. She is a colleague, a friend, she helps me on the paperwork, she directly helped me on this thesis, she's part of my family. But I write about Mégane in this last paragraph because she's also something more than all this. I feel immensely lucky to have met you and to count on you. I hope we will face together many other complicated and funny endeavors!

General Introduction

Cable-Driven Parallel Robots (CDPRs) represent a subclass of parallel kinematic manipulators in which rigid links are replaced by cables. They consist mainly of a mobile platform driven by cables, which transmit the forces generated by winches. Accordingly, the platform motion is managed controlling the winch motors.

Since a large length of cable may be wound on the drum of each winch, CDPRs can have a large workspace. Furthermore, thanks to the reduced inertia of the cables, CDPRs can attain high velocities and accelerations. In addition, cables are able to transmit large forces, which make possible the handling of heavy payloads. Among other advantages, these characteristics motivated studies applying CDPRs for high speed manipulation, simulations in virtual reality, handling of heavy payloads in large workspaces, etc. As a matter of fact, the design, control and analysis of CDPRs are well established research subjects [11].

The work presented in this thesis has been made in the framework of the european H2020 project called Hephaestus [12]. The main goals of this project include the design and control of a CDPR able to install the Curtain Wall Modules (CWMs) of a building facade. Inspired by the specificities of the Hephaestus project, this thesis contributes to the development of advanced control techniques.

Clearly, the mounting of CWMs demands high positioning precision (10 mm). In addition, safety and disturbance rejection capabilities are major concerns considering that the robot should work in a construction site. Moreover, the control scheme should be compatible with a specific set of industrial hardware and software. Finally, due to design constraints, the robot needs to operate close to the boundaries delimiting the safe conditions of the system operation.

Considering this context, it was noted that the state-of-the-art control schemes would lead to a crucial issue. Namely, the operation close to system constraints with a significant influence of disturbances may lead to undesired responses. This is due to the fact that the existing strategies do not consider system constraints as an integral part of the main control scheme. In order to overcome this issue, this thesis introduces Model Predictive Control (MPC) schemes (linear and nonlinear) able to explicitly handle the system constraints within the formulation of the controller. Numerical and experimental results validate the proposed strategies. Moreover, the stability of a nonlinear MPC (NMPC) scheme is also analyzed.

Before applying the proposed MPC schemes, several issues related to the experimental implementation had to be solved. Some of them are addressed in this manuscript. In particular, a chapter is dedicated to contributions related to the cable tension control and forward kinematics of CDPRs. In addition, the programming solutions proposed to the development of the control schemes in an industrial software are briefly examined.

Organization of the Thesis

A brief description of the chapters and appendices of this thesis are presented in the following.

- **Chapter 1** presents the context of this thesis and review the pertinent state-of-the-art on the control of CDPRs. Considering this context, an issue raised by the existing control schemes is identified and discussed. It is noted that a control scheme suitable for the project Hephaestus should consider the system constraints as an integral part of the main controller. Some details on MPC are reviewed, demonstrating that this control architecture may be suitable in the context of this thesis;
- **Chapter 2** introduces the implemented kinematic model and the control of cable tensions. A forward kinematic algorithm considering the pulley geometry is proposed. These results were published in (I). Regarding the cable tension control, two different approaches are considered. Numerical simulations and experimental tests indicate that a velocity-based scheme is suitable for the studied systems;
- **Chapter 3** introduces a linear MPC for the position tracking control of CDPRs. The strategy is validated numerically and experimentally, indicating that the proposed MPC controller overcomes the issues described in Chapter 1. Simulation results were published in (III) and experimental ones in (II);
- **Chapter 4** proposes a nonlinear MPC scheme. In contrast to the linear MPC, this approach leads to satisfying results even with an increased incidence of nonlinearities. The stability of the nominal closed-loop system is analyzed. The comparison between the linear and nonlinear MPC is performed through real-time experimental tests and numerical simulations;
- **Appendix A** briefly discusses the development of a C++ numerical library compatible with the industrial software TwinCAT. Thanks to this library, complex matrix operations can be programmed in an intuitive manner, facilitating the implementation of the proposed MPC schemes;
- **Appendix B** presents a closed-form tension distribution algorithm able to minimize the infinity-norm of the cable tensions vector. The optimality of the algorithm output is proved and experimental results are discussed. The proposed algorithm is based on the results published in (IV).

- **Appendix C** addresses the stability criterion that is used in the analysis of the proposed NMPC scheme. The results introduced in [13] that are used Chapter 4 are discussed more in detail.

List of Publications - International Conference Proceedings

(I) J. C. Santos and M. Gouttefarde, "A Real-Time Capable Forward Kinematics Algorithm for Cable-Driven Parallel Robots Considering Pulley Kinematics," in Advances in Robot Kinematics (ARK), 2020.

(II) J. C. Santos, A. Chemori, and M. Gouttefarde, "Redundancy Resolution integrated Model Predictive Control of CDPRs : Concept , Implementation and Experiments," in Proceedings - IEEE International Conference on Robotics and Automation, 2020, pp. 3889–3895.

(III) J. C. Santos, A. Chemori, and M. Gouttefarde, "Model Predictive Control Applied to Large-Dimension Cable-Driven Parallel Robots," in Cable-driven parallel robots, 2019.

(IV) H. Hussein, J. C. Santos, and M. Gouttefarde, "Geometric Optimization of a Large Scale CDPR Operating on a Building Facade," in IEEE International Conference on Intelligent Robots and Systems, 2018, pp. 5117–5124.

1

Introduction and State of the Art

Cable-Driven Parallel Robots (CDPRs) present several particular advantages when compared to rigid link robots. Section 1.1 discusses the reasons that lead to these advantages and exemplify some of the various applications of CDPRs that take advantage of them. Section 1.2 then introduces the main goals of the project Hephaestus that form the context of this thesis. The main models of CDPRs proposed in the literature are briefly presented in Section 1.3. Notably, this section examines the various assumptions that may be taken and the main consequences on the resulting models. A summary of the state of the art on position tracking of CDPRs is introduced in Section 1.4. The vast majority of the studies presented in this section are coupled to some tension distribution algorithm. This subject is addressed in Section 1.5. Section 1.6 presents existing cable tension control schemes. The very beginning of Section 1.7 highlights an important limitation of the coupling between schemes presented in Section 1.4 and algorithms of Section 1.5. Model Predictive Control (MPC) is then proposed as a solution to overcome this drawback. Section 1.7 presents details on this type of feedback control strategies. Finally, Section 1.8 summarizes the main contributions presented in this thesis.

Contents

1.1 Cable-Driven Parallel Robots

The main components of a Cable-Driven Parallel Robot (CDPR) are cables, winches and a mobile platform. Each cable has one end attached to the platform and the other one wound on a drum. Each one of these drums is coupled to a motor, composing the winches. Figure 1.1 depicts a CDPR and highlights these main elements. Altogether, the cables transmit the torques applied on the drums so that the platform motion may be driven by the actuation of the motors.

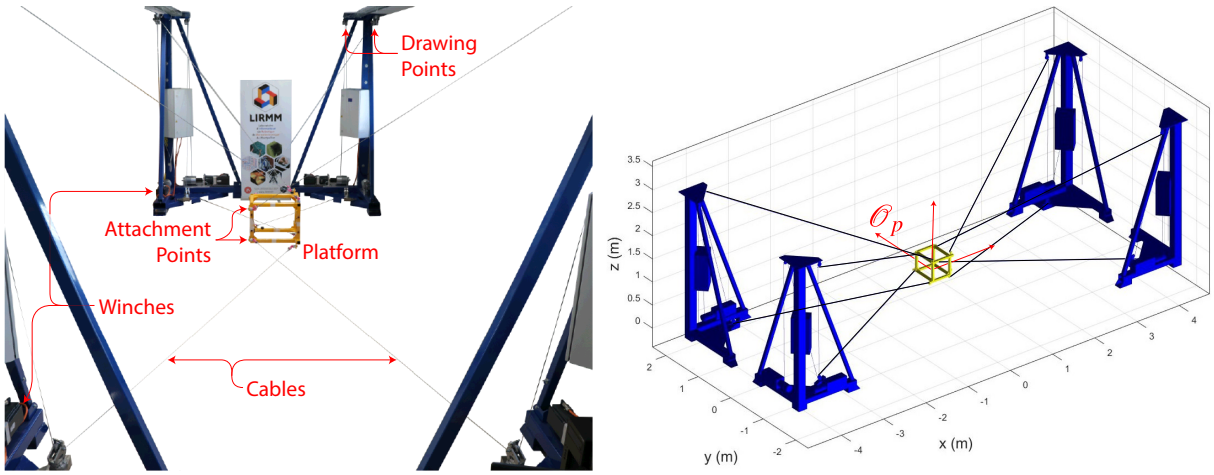

Figure 1.1 – *HRPCable prototype and its CAD view.*

1.1.1 CDPR characteristics and main applications

CDPRs have several advantages compared to rigid link robots. Since the rigid links are replaced by cables that may present a large length wound on the drums, the mobile platform may be displaced over a large workspace. A few examples among the numerous robots that take advantage of this feature are the well known *Skycam* [14] and large-scale radio telescopes [2, 15]. Figure 1.2 (a) depicts the large dimension telescope FAST installed in China. Moreover, the use of cables may be very efficient to handle heavy payloads, rendering the application of CDPRs favorable for the handling of heavy objects [3, 6], as the CoGiRo prototype shown in Figure 1.2 (b).

The combination of a large workspace with the reduced visual interference obtained using cables instead of rigid links makes cable-driven parallel mechanisms a pertinent solution for haptic devices [4, 16–18]. Rehabilitation is a typical application of such solutions. The haptic device Inca, commercialized by Haption, is shown in Figure 1.3.

As a subclass of parallel robots, CDPRs typically present their actuators fixed to the inertial base. In addition, the inertia of the moving links may be drastically reduced using cables instead of rigid links. As a result, several works focused on the capability of CDPRs to reach high velocities and accelerations [19–22]. For instance, the FALCON prototype is able to generate 13 m/s of maximal velocity and 43 g of maximal acceleration [19]. Similarly, motion simulators may take advantage of the high dynamic

(a) (b)

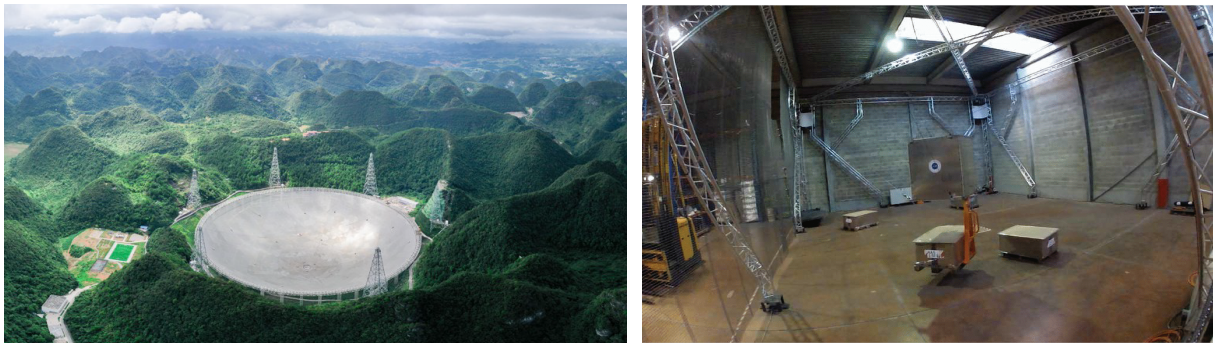

Figure 1.2 – *(a) The five-hundred-meter aperture spherical radio telescope FAST [1, 2] and (b) the prototype built in the framework of the research project CoGiRo (Control of Giant Robots) - TECNALIA/LIRMM [3].*

Figure 1.3 – *The commercial solution Inca, a cable-driven haptic device from Haption [4].*

capabilities obtained with the architecture of CDPRs [5] (see Figure 1.4).

In contrast to these advantages, CDPRs also have important drawbacks. The cable elasticity may introduce undesired vibrations [23–26]. Similarly, substantial degradation of the positioning precision may be obtained when neglecting cable elongation [27]. Moreover, the cable tensions should be kept sufficiently high in order to avoid the presence of slack cables, which not only may lead to the malfunctioning of the winches but also degrades positioning accuracy.

1.1.2 CDPR types

Whereas rigid links may transmit both compression and tensile forces, cables can only operate in tension. Considering this limitation, the Wrench Closure Workspace (WCW) defined in [28] is "*the set of poses of the platform for which any wrench can be gener-*

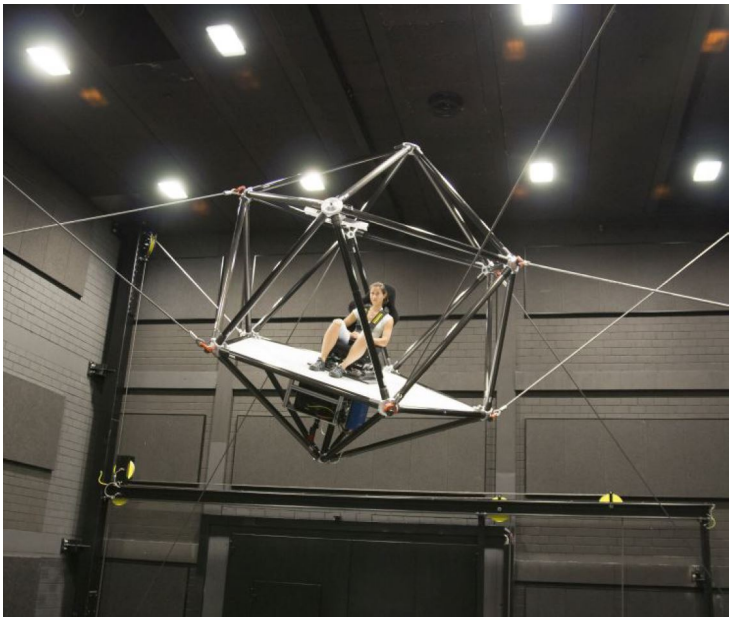

Figure 1.4 – *A motion simulator prototype developed in the Max Planck Institute for Biological Cybernetics [5].*

ated at the platform by tightening the cables". Here, the term *pose* refers to the combination of position and orientation. A mobile platform pose within the WCW is said to be fully-constrained.

It is interesting to note that many CDPRs present nonexistent WCW. This is the case of the *suspended* CDPRs, such as the ones depicted in Figures 1.2 (a) and (b). Since every cable force applied on the platform is directed upwards, this type of CDPR can not generate a force directed downwards. Therefore, suspended CDPRs rely on the gravitational forces in order to constraint the platform.

In contrast, if the WCW represents a substantial part of the robot workspace, the CDPR is classified as *fully-constrained*. While suspended CDPRs may have superior payload capacity, fully constrained CDPRs typically present better precision, stiffness and dynamic capabilities. Robots depicted in Figures 1.3, 1.4, and 1.5 are examples of fully-constrained CDPRs. It is worth noting that a fully-constrained CDPR necessarily presents a number of cables greater than the number of Degrees of Freedom (DoF) of the mobile platform. This actuation redundancy yields some particularities on the control. These particularities play an important role on the main results presented in this thesis. Sections 1.5 and 1.7 discuss these particularities more in detail. At any rate, even though the studies presented in this thesis are focused mainly on redundant CDPRs, they may be directly applied for non-redundant and suspended CDPRs.

Conversely, the platform pose cannot be determined geometrically for CDPRs with less cables than DoF. In addition to the kinematic model, a static or dynamic model should be considered in order to determine the platform pose. Accordingly, such robots are classified as under-constrained CDPRs. This class of CDPRs was studied, for instance, in [29, 30]. In spite of the interesting research subjects devoted to under-constrained CDPRs, this thesis does not deal with this kind of robot.

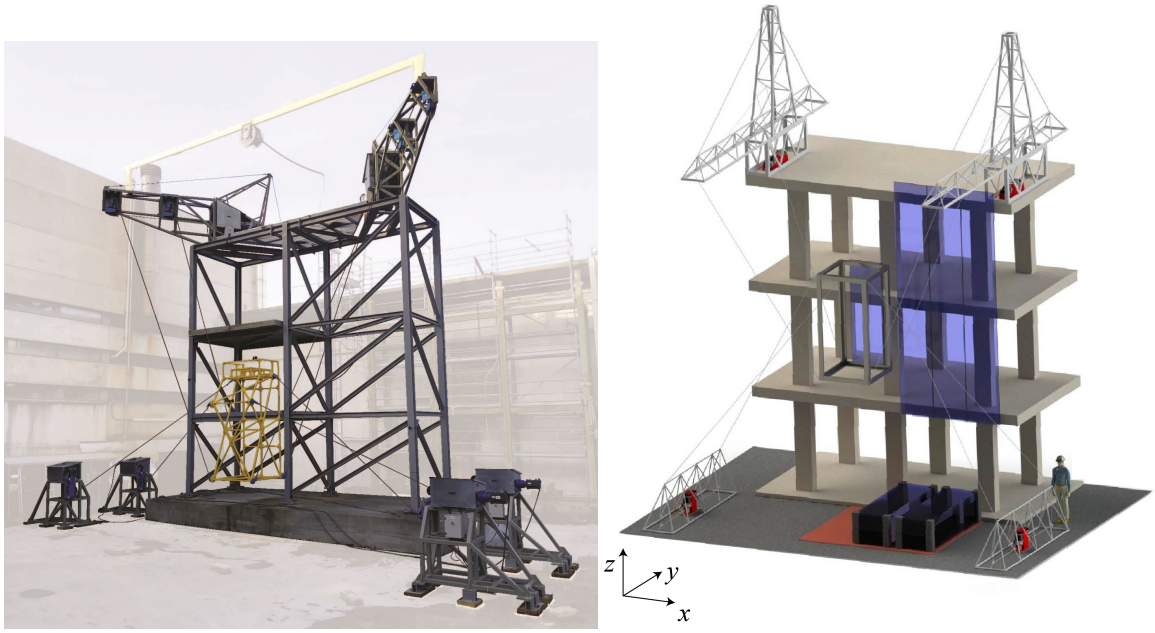

Figure 1.5 – *Hephaestus prototype and an illustration of its main application, the installation of curtain wall modules.*

1.2 Context of the Thesis

The contributions of this thesis were developed within the framework of the European Horizon 2020 project Hephaestus [12]. The main goal of this project is to develop a robotic solution for the installation of Curtain Wall Modules (CWM) on building facades. Figure 1.5 presents the Hephaestus CDPR prototype and illustrates its main application. Indeed, one may note that the choice of using a CDPR for this application is in accordance with the advantages introduced in the previous section, notably the possibility to reach large workspaces (the size of a building facade) and the capability to handle heavy payloads (approximately 1300 kg in this specific case).

The positioning of the CWMs is performed with the aid of suction cups installed on the mobile platform. Furthermore, the prototype also assists the installation of the brackets on the building slabs. This task includes the drilling of the slabs, positioning and mounting of the brackets. In spite of the interesting challenges involved in the bracket installation, the contributions presented in the present manuscript are not dependent on technical details related to them. For this reason, among the extensive technical data involved in the whole project, just a few relevant inputs are detailed in Table 1.1.

The main contributions of LIRMM in the Hephaestus project are (i) the geometric design optimization of the CDPR and (ii) the development of advanced control techniques. Since most of the remaining tasks of the project depend on the outputs of (i), the results related to this task were delivered in the first half of the project. These results are presented in [31] and my contributions in this task were minor. Therefore, this thesis is mainly focused on the development of CDPR control strategies.

The time length in which the Hephaestus prototype was fully built and available

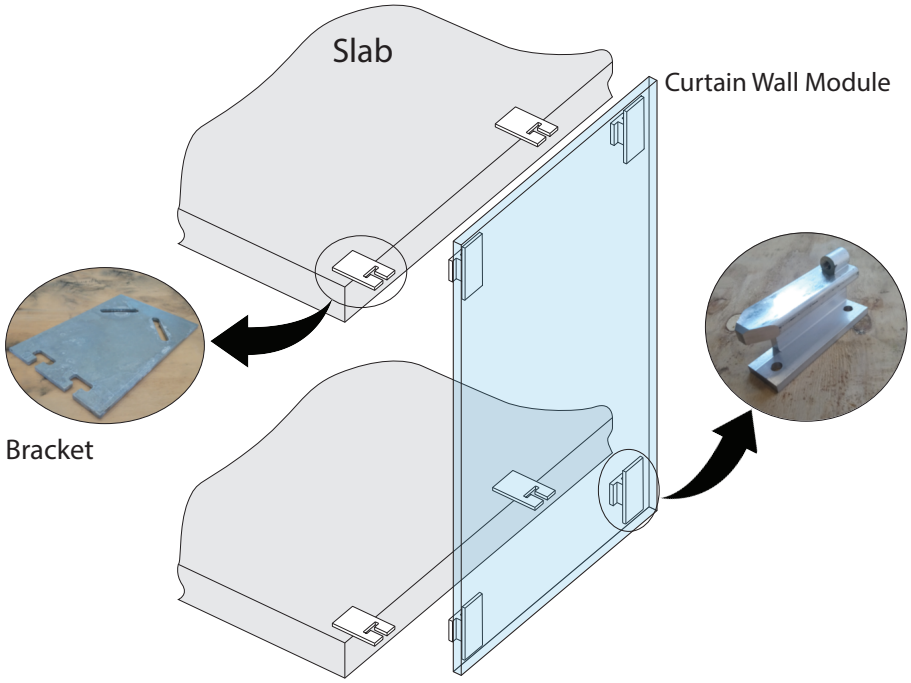

Figure 1.6 – *Illustration of the CWM assembly.*

for tests is short in comparison to the duration of the project. Moreover, this available time needs to be shared between the partners of the consortium who participate to the experimental tests. As a result, knowing that the experimental activities were crucial and might reveal non-predicted difficulties, a secondary experimental set-up was used. This set-up is a CDPR prototype, HRPCable, with characteristics rather similar to the ones of the Hephaestus prototype. The HRPCable is shown in Figure 1.1 and Table 1.1 summarizes its main features. Indeed, most of the experimental results presented within this thesis were performed with this prototype.

Table 1.1 – *Hephaestus and HRPCable main features.*

	Hephaestus	HRPCable
Payload	1300 kg	100 kg
Footprint	$14 \times 4 \times 12$ m	$8 \times 4 \times 6$ m ($x \times y \times z$)
Number of cables	8	8
Degrees of freedom	6	6
Maximum cable tension	16 kN	2 kN
Diameter of the cables (steel)	12 mm	4 mm
Diameter of the pulleys	300 mm	80 mm
Platform mass	23 kg	400 kg
Motors and Hardware	Beckhoff	Beckhoff
PLC	C6650 3.6 GHz i7	C6920 2.4 GHz i7
Motors	AM8071	AM8061
Gear trains (reduction)	Planetary in line (76.2:1)	AG2210 (25:1)

Before addressing the modeling and control of CDPRs, the main concerns related to them should be summarized considering the specificities of the Hephaestus project:

- A crucial challenge is the mounting of the CWMs on the brackets. *High precision* is necessary and a versatile control strategy may be needed, since the assembly is performed in four brackets simultaneously (see the illustration in Figure 1.6). The position of the brackets are not necessarily known with the precision required to the assembly;
- Since the robot should work outdoors, in a non-controlled environment, the proposed control scheme should possess *external disturbance rejection capabilities*;
- Given that the robot is intended to operate on a construction sites (with the possible presence of non-trained personnel) and the payload of the machine is large, *safety* is a critical concern;
- Due to the reduced available area in front of the building facade, the nominal designed workspace presents a small margin dedicated to uncertainties. In other words, some poses necessary to the accomplishment of the desired tasks are close to the nominal workspace limits. Therefore, the proposed control scheme should be able to safely handle the robot *operation close to system constraints*;
- Considering that the robot is intended to operate on a construction sites (with the possible presence of non-trained personnel) and the payload of the machine is heavy, *safety* is a critical concern;
- Knowing that the machine is intended to be commercialized, the used *hardware and software* should be *compatible* with the ones commonly used in the *industry*. Since the commonly available industrial softwares are not compatible with advanced mathematical programming libraries, this last aspect leads to remarkable constraints related to the implementation of advanced control schemes.

1.3 Modeling

Most of the control strategies discussed in Section 1.4 are model-based control schemes. Indeed, the design of these control strategies takes advantage of the well-known modeling of CDPRs. For this reason, it is interesting to present some details on the kinematic and dynamic models of CDPRs before addressing the control design itself.

The kinematic model of a CDPR correlates the pose and velocity of the mobile platform to the motor positions and velocities. The model relating a set of joint positions to a known pose of the platform is the *Inverse Kinematic* (IK) model. Conversely, the calculation of a platform pose for a known set of motor positions is the *Forward Kinematic* (FK) model. The different mathematical models used to implement et IK and FK may be classified with respect to their assumptions, leading to different levels of precision.

One may neglect the cable elasticity and mass, considering that the profile of each cable is a straight line linking an attachment point to the corresponding drawing point. Even though these may seem rather naive assumptions, several important studies were based on the corresponding model [32–34]. On the other hand, several approaches were proposed seeking more precise models, as discussed in the following.

Typically, previous works search for a compromise between model simplicity and precision in the context of each specific study. Firstly, one may address the problem that

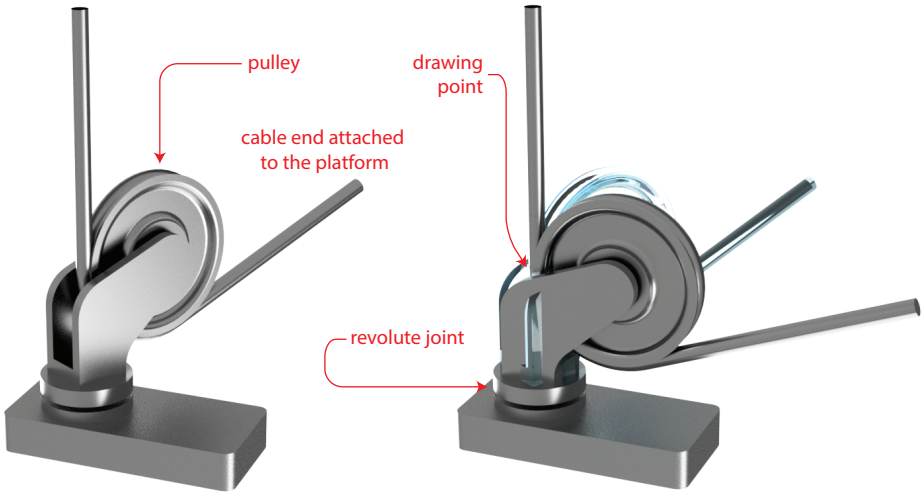

Figure 1.7 – *A typical mechanism on which the drawing pulleys are mounted, illustrating that the drawing point is not fixed.*

drawing points are not fixed. In general, the drawing pulleys rotate and the wrapping angles change in order to follow the motion of the mobile platform, as illustrated in Figure 1.7. The influence of these changes, which is often referred to as *pulley kinematics*, may be significant [35] and may be taken into account in the control of CDPRs [36,37].

Moreover, the fact that in reality the cables are elastic and do not present a straight line profile may be also studied. In cases where the cable tensions are low and the lengths are large, the influence of the cable distributed mass may be non negligible. Therefore, as an example, it is not surprising that this phenomenon was considered in the design of the CDPR FAST illustrated in Figure 1.2 (a). Yao *et al.* obtained in [2] a good compromise between precision and computing time approximating the cable profile with a parabola. A model proposed in [38] (usually referred to as Irvine model) is often used to obtain more precise results [39–43]. In contrast, if cable tensions are sufficiently high and cable specific mass is relatively low, cables may also be considered as springs [26,44,45].

At any rate, one may note that if a model considering the cable elasticity and/or distributed mass is used, the static and dynamic loads applied on the platform result in different outputs of the kinematic model. In other words, the pose does not depend only on cable lengths, but also on the loads applied on the platform. The kinematic model is, therefore, coupled to the dynamic model. Whereas cable sagging and elasticity may be very difficult to integrate in the model of the robot, the platform motion for known cable forces may be modeled with a rather simple approach. As a matter of fact, for a known set of cable forces, the platform motion can be modeled using the Euler's equations of motion [46, Section 5.5].

Apart from the modeling of the cables, pulley kinematics and platform motion, a particular attention was dedicated to the forward kinematics. As for parallel robots in general, the estimation of the platform pose for given positions of the actuators is not a trivial task. Whereas this issue may be avoided with the application of an external measurement system capturing the pose of the platform. This can be done, for instance, with the motion-capture system Bonita developed by Vicon [47], with a multi-

cameras setup [48] or with a single camera embedded on the mobile platform [49]. Nevertheless, the implementation of such measurement systems within uncontrolled environments may lead to various issues related to occlusions and lighting variations. Furthermore, these solutions also depend on additional sensors, which may be costly.

Therefore, several studies sought a reliable and accurate algorithm able to compute the CDPR platform pose for a given set of winch positions, i.e., Forward Kinematic (FK) Algorithms. The FK problem may be solved with interval analysis [50, 51]. This approach allows one to find all the possible FK problem solutions, taking into account that some cables may be slack. Even though this method is considered mathematically complex, real-time capability has been presented [52]. Additionally, iterative numerical schemes constitute an alternative typical solution [53–55]. In general, the FK problem is formulated as a nonlinear optimization problem. The iterative algorithm (usually, Levenberg-Marquadt) delivers a platform pose that locally minimizes modeling errors, taking the winch positions as input. To this end, a differential kinematic model is necessary.

1.4 Position Tracking Control

As discussed in the previous sections, this thesis is focused on advanced control techniques for CDPRs. The present section introduces the state-of-the-art of the studies addressing this topic. The main subject addressed in this thesis is the problem of *position tracking control*. The fundamental input of this problem is a desired trajectory. The control scheme is responsible for *tracking* this desired trajectory, *i.e.* generate a motion so that, at each instant of the movement, the measured (or estimated) actual pose is as close as possible to the desired one.

The approaches presented in the literature of position tracking control for CDPRs may be organized as follows. A commonly used classification separates the control schemes in two groups, namely, Joint Space Control (JSC) and Operational Space Control (OSC). In this thesis, a control strategy is classified as an OSC scheme if it measures (or estimates) the mobile platform pose error. Otherwise, the strategy is classified as JSC. Loosely speaking, an OSC strategy presents a closed loop which handles the platform pose error.

1.4.1 Joint space control

Most probably, the simpler schemes to first operate a CDPR were JSC strategies. Consider an actuation in which the motor angular positions are controlled taking the values obtained using the IK model for each pose of the desired trajectory. Such control scheme is commonly referred to as kinematic control and was the control scheme used to control the NIST ROBOCRANE [6] shown in the Figure 1.8.

As the reader may note, the JSC approach may certainly lead to a simple CDPR control strategy. Nevertheless, more elaborate strategies may be derived with the same

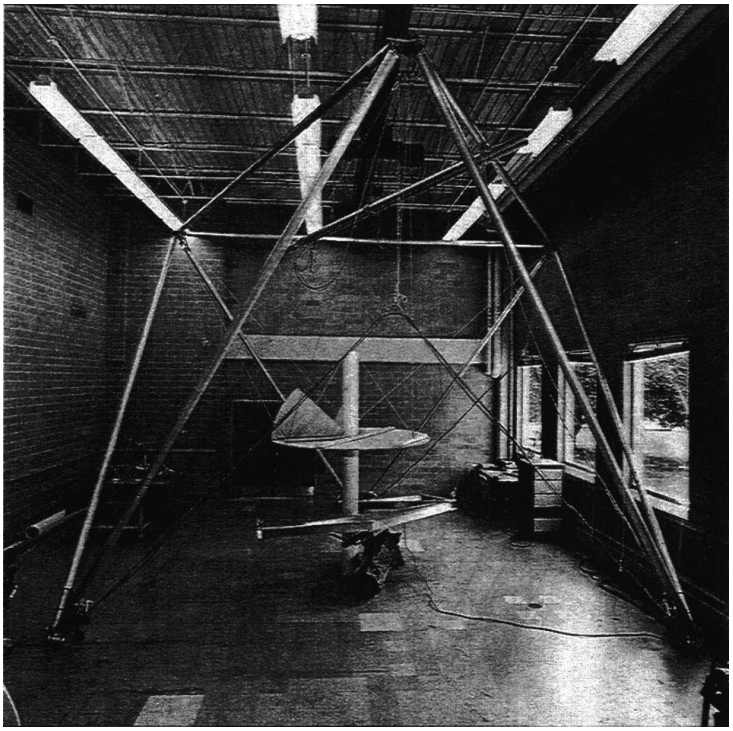

Figure 1.8 – *The NIST ROBOCRANE, controlled with a Joint Space control [6].*

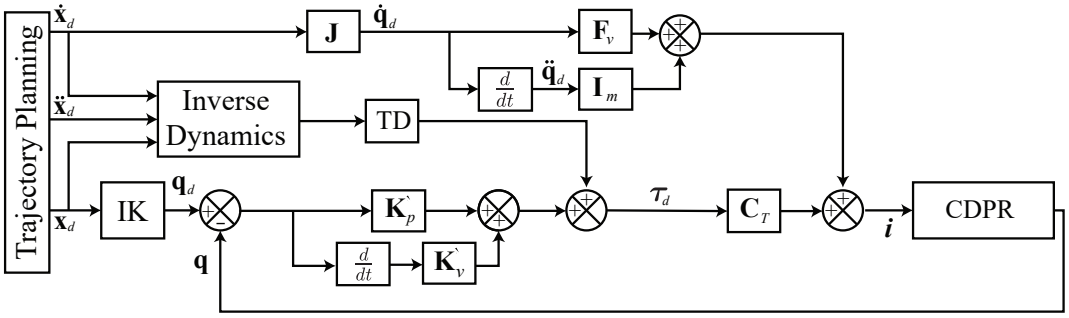

Figure 1.9 – *A Joint Space Control scheme proposed in [7]. The main variables are: The desired and actual pose* $\mathbf{x}_d$ *and* $\mathbf{x}$*, desired and measured motor positions* $\mathbf{q}_d$ *and* $\mathbf{q}$*, Jacobian matrix* $\mathbf{J}$*, desired cable tensions* $\boldsymbol{\tau}_d$*, dynamic motor friction constant* $\mathbf{F}_v$*, motor inertias* $\mathbf{I}_m$*, motor currents* $\boldsymbol{i}$*, a diagonal constant matrix* $\mathbf{C}_T$ *and PD gains* $\mathbf{K}'_p$ *and* $\mathbf{K}'_v$*. The blocks TD and IK stand for the Tension Distribution algorithm and Inverse Kinematics, respectively. Time derivatives are denoted using the dot notation.*

basis. Kawamura *et al.* propose a similar scheme adding the compensation of the gravitational effects and the management of internal forces [56]. Fang *et al.* propose in [7] the control strategy of Figure 1.9. The closed-loop in this block diagram is exclusively dedicated to the joint space and the Inverse Dynamics block delivers the *feedforward* wrench which generates the desired motion.

These control strategies implement uncoupled feedback gains, *i.e.* the feedback gain matrices are diagonal. Alternatively, the synchronous tracking control approach introduced in [57] may be used in order to improve the coordination between the cable displacements [58]. In practice, this strategy is based on gain matrices that are not diagonal.

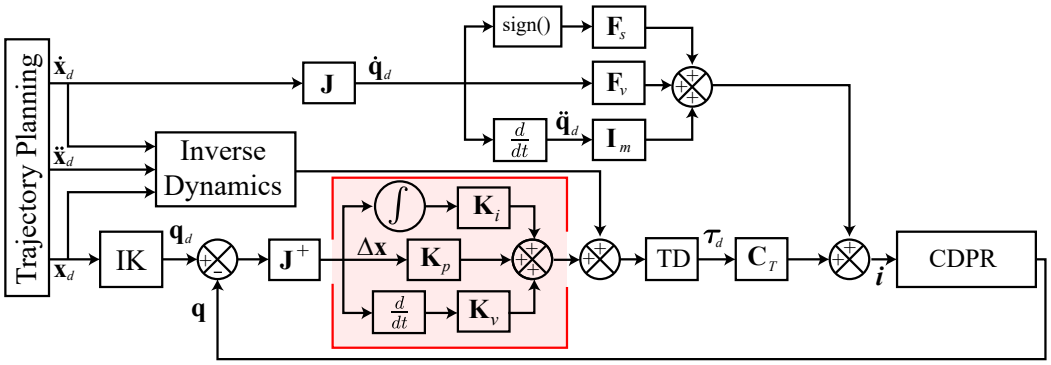

Figure 1.10 – *The Operational Space Control proposed in [8]. The variables that were not introduced in Figure 1.9 are: The pseudo-inverse jacobian matrix* $\mathbf{J}^+$*, static motor friction constant* $\mathbf{F}_s$*, tracking error of the pose* $\Delta\mathbf{x}$ *and PID gains on operational space* $\mathbf{K}_p$*,* $\mathbf{K}_i$ *and* $\mathbf{K}_v$*.*

Kraus *et al.* [59, 60] propose a control scheme in which the cable tensions are controlled using the measurement of force sensors. As in [7], the desired wrench to be applied on the platform (and the corresponding cable tensions) are computed based on the desired trajectory without the estimation of the error of the platform pose.

1.4.2 Operational space control

As a matter of fact, the majority of the advanced control techniques for the position tracking of CDPRs belongs to the class of OSC. The strategy implemented by Lamaury and Gouttefarde in [8] and presented in Figure 1.10 may be taken as an example to introduce the rationale commonly used on OSC. Comparing Figures 1.9 and 1.10, one may note that the schemes proposed in [7] and [8] are rather similar. The main difference lies on the fact that instead of applying the closed-loop strategy directly on the difference between desired and measured motor positions, the PID correction in Figure 1.10 takes as input the estimated error of the platform pose (an estimation obtained as $\Delta\mathbf{x} = \mathbf{J}^+(\mathbf{q}_d - \mathbf{q})$).

Inasmuch as a large amount of studies present a very similar approach, it is worthwhile to highlight some details involved in its motivations. One may note that the feedforward terms compensate most of the nonlinearities of the system, rendering the proposed approach closely related to the widely used computed torque control [61] or the feedback linearization method [62, Section 9.3]. Indeed, most of the control strategies discussed in this section presents a very similar rationale. The corresponding schemes are alike the one in Figure 1.10, changing notably the highlighted region, which computes the wrench representing the feedback correction for a given error of the platform pose. Note that full feedback linearization may be easily applied for non-redundant CDPRs, as proposed in [63]. Nevertheless, the presence of a number of cables greater than the number of degrees of freedom may prevent the application of such method for redundant CDPRs.

Whereas the nominal stability of these approaches can be straightforwardly proven given the simplicity of the nominal closed-loop system, interesting results were ob-

tained regarding robustness. The robust stability considering structured and unstructured uncertainties of a rather similar scheme was proved in [64] using Lyapunov theory. Similarly, Korayem *et al.* analyzed the robustness regarding the flexibility of the joints of a fully feedback linearized control scheme [63]. Alternatively to the direct Lyapunov theory, Khosravi and Taghirad addressed in [44] the stability of a similar scheme using singular perturbation theory [65]. A simple and elegant solution was proposed to extend the stability results when the elasticity of the cables are considered.

Yet in the field of linear control systems, more advanced techniques may be implemented. The Linear Quadratic Regulator (LQR) method was used in [66] considering the feedback linearized system. As for the study in [63], a non-redundant robot is necessary to obtain a linear system. The LQR method delivers the optimal gain matrices corresponding to a deterministic infinite horizon optimal control problem applied for linear systems [67, Section 9.2.1]. Due to the fact that its theoretical basis relies on the deterministic model of the system, the LQR may be sensitive to disturbances. To deal with this issue, the $\mathcal{H}_\infty$ (H-infinity) robust control [67, Section 9.3.4] method may be used. This possibility was investigated in [47, 68] using the Inca robot (as the one in Figure 1.3). Motivated by the fact that all the aforementioned studies present control schemes that are based on the theory of linear systems, such methods are denoted in this section as linear-based methods.

Clearly, methods developed with theories that are not based on the linearization of the studied system may be used. These approaches are classified in this section as nonlinear-based methods. Firstly, Sliding Mode Control (SMC) received particular attention in the control of CDPRs. The main advantages of this nonlinear control method are finite time convergence and robustness to modeling uncertainties and disturbances. Moreover, the implementation of SMCs are often simpler than other advanced nonlinear control techniques.

Oh and Agrawal [69] implemented the theory proposed in [70] in order to estimate the feasible workspace of CDPRs. Subsequently, improved versions were proposed, *e.g.* terminal sliding mode [71] and super twisting controller [72]. Nevertheless, chattering is a well-known issue commonly addressed in studies implementing SMCs. Even though numerous methods were proposed in order to reduce this difficulty, chattering is still observed in recent results [72]. Besides SMC, other nonlinear feedback control approaches were proposed, such as fuzzy control [73], neural networks [74] and the intricate method proposed in [75] involving dynamic programming with an actorcritic structure.

Inspired by the fact that the applications of CDPRs are frequently subjected to non-negligible uncertainties (in particular regarding the payload and the friction of the winches), a lot of effort has been devoted to adaptive strategies. Most of them are based on the adaptation law introduced in [76], also proposed by Slotine. An adaptive version of [8] is presented in [77], resulting in a better tracking performance than its predecessor. As a matter of fact, many of the aforementioned nonlinear control schemes are classified as adaptive strategies [71, 72, 74, 75]. Figure 1.11 summarizes the classifications discussed in the present subsection.

One may note that most of the schemes discussed in the present section define,

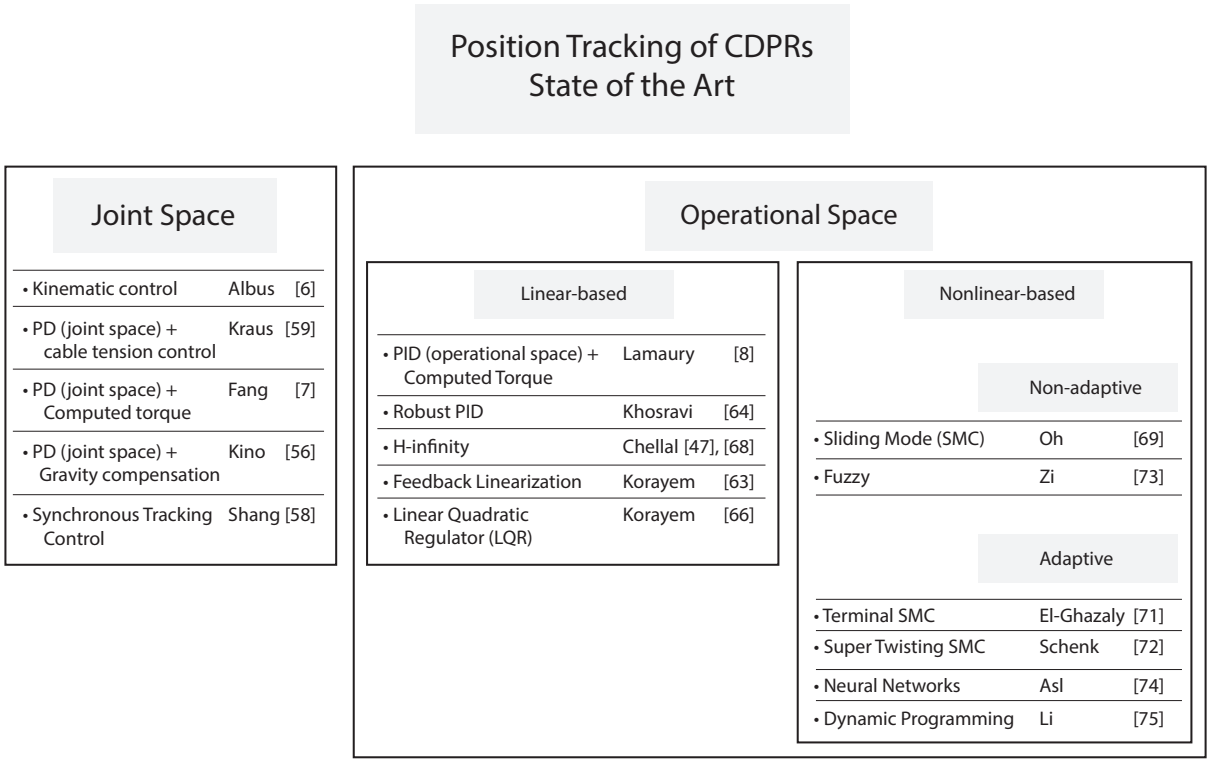

Figure 1.11 – *Summary of main studies addressing position tracking of CDPRs.*

at some point, a desired wrench to be applied on the platform. This wrench should be generated by the combination of the forces applied by the cables on the mobile platform. For a non-redundant CDPR, *i.e.* having a number of cables equal to the number of DoF, the relation between these sets of variables may be straightforwardly obtained. Nevertheless, for a redundantly actuated CDPR, this subject merits special attention. The next subsection is devoted to this issue.

1.5 Tension Distribution Algorithms

Different kinds of redundancies may be defined in robotics [78–80]. In the context of this thesis, a specific type is of particular interest, namely, actuation redundancy. Although the definition of this class of redundancy may be the subject of discussions [78], a simple definition may be synthesized for the CDPRs studied in this thesis. The classification of redundancy for reconfigurable CDPRs [81–84] or robots presenting articulated mobile platform [85, 86] may render necessary a detailed discussion. Nevertheless, the studies presented here are focused on non-reconfigurable robots with a platform consisting of a unique non-articulated body. In this case, a CDPR having a number of cables (m) greater than the number of DoF of the mobile platform (n) is classified as redundantly actuated. Most of the results presented in this thesis takes as motivation some particularities of this class of CDPRs.

For the majority of commonly used modeling approaches, for a given platform pose, one may define a constant linear mapping between the vector of cable tensions $\boldsymbol{\tau}$ and the resulting wrench $\mathbf{f}$ applied on the platform [87]. Here, each element τ_i of the

vector $\boldsymbol{\tau}$ is the euclidean norm of the (vectorial) force applied on the platform by the corresponding cable. More precisely, there exists a matrix $\mathbf{W}$, named wrench matrix, so that $\mathbf{f} = \mathbf{W}\,\boldsymbol{\tau}$.

Therefore, if $m = n$, a set of cable tensions able to generate a desired wrench $\mathbf{f}_d$ may be trivially obtained inverting such (bijective) linear mapping (if the considered pose is not singular, leading to an invertible square matrix $\mathbf{W}$). Otherwise, non-trivial problems emerges if $m \neq n$. Robots with $m < n$ are referred to as under-constrained CDPRs [29, 30] and their study is out of the scope of this thesis. In contrast, if $m > n$, infinitely many sets of cable tensions may generate the same wrench. The algorithms responsible for the definition of the most pertinent set of cable tensions for a given desired wrench are called *Tension Distribution Algorithms* (TDA).

In general, TDAs should comply with two sets of constraints. Clearly, one set of constraints result from the fact that the cable tensions should generate the desired wrench, *i.e.* vector $\boldsymbol{\tau}$ should satisfy $\mathbf{f}_d = \mathbf{W}\,\boldsymbol{\tau}$. In addition, cable tensions should be within a positive interval. This positive interval is delimited by sets of maximal and minimal cable tensions, $\boldsymbol{\tau}_{max}$ and $\boldsymbol{\tau}_{min}$, respectively. The minimal cable tensions are considered in order to avoid excessively low tensions leading to the malfunctioning of the winches and to slack cables, which may result in deteriorated precision and stiffness. Maximal cable tensions are defined according to the maximal load with under which every mechanical part operates safely.

To sum up, the goal of a TDA is to define a pertinent set of cable tensions $\boldsymbol{\tau}$ so that $\mathbf{f}_d = \mathbf{W}\,\boldsymbol{\tau}$ and $\boldsymbol{\tau}_{min} \leqslant \boldsymbol{\tau} \leqslant \boldsymbol{\tau}_{max}$, taking as input the linear mapping $\mathbf{W}$, the desired wrench $\mathbf{f}_d$ and the cable tension limits $\boldsymbol{\tau}_{min}$ and $\boldsymbol{\tau}_{max}$. Accordingly, a TDA can be formulated as an optimization problem

$$\begin{aligned} \boldsymbol{\tau}^* = \arg\min_{\boldsymbol{\tau}} \; & f(\boldsymbol{\tau}) \\ \text{s. t.} \; & \mathbf{f}_d = \mathbf{W}\,\boldsymbol{\tau} \\ & \boldsymbol{\tau}_{min} \leqslant \boldsymbol{\tau} \leqslant \boldsymbol{\tau}_{max} \end{aligned} \quad , \tag{1.1}$$

with a given cost function $f : \mathbb{R}^m \to \mathbb{R}$, so that the essential difference among them is the definition of f.

A natural choice for f is some norm of the vector $\boldsymbol{\tau}$. Using the typical definition of vector norms as in [88, Section 2.2] with $\|\boldsymbol{\tau}\|_p = (\tau_1^p + \tau_2^p + \cdots + \tau_m^p)^{1/p}$, several studies address the TDA formulated as (1.1) for $f(\boldsymbol{\tau}) = \|\boldsymbol{\tau}\|_p$ with p some positive non-null integer.

Using the 1-norm with $p = 1$, the corresponding optimization problem (1.1) is classified as a Linear Programming (LP) problem [89]. Snyman and Hay simplified the general methodology implemented to the solution of LP problems for the specific constraints of TDAs [90]. Similarly, an explicit expression of the vector of cable tensions minimizing $\|\boldsymbol{\tau}\|_1 = \sum \tau_i$ is deduced in [7]. As an alternative this, one may apply weighing costs c_i for each individual cable i according to the specific needs of the CDPR. For instance, a critical issue for the suspended CDPR in [91] is to keep the longer cables sufficiently tighten. Hence, a cost function $f(\boldsymbol{\tau}) = \sum c_i\,\tau_i$ with negative c_i may be used for the cables that need to be under increased cable tensions.

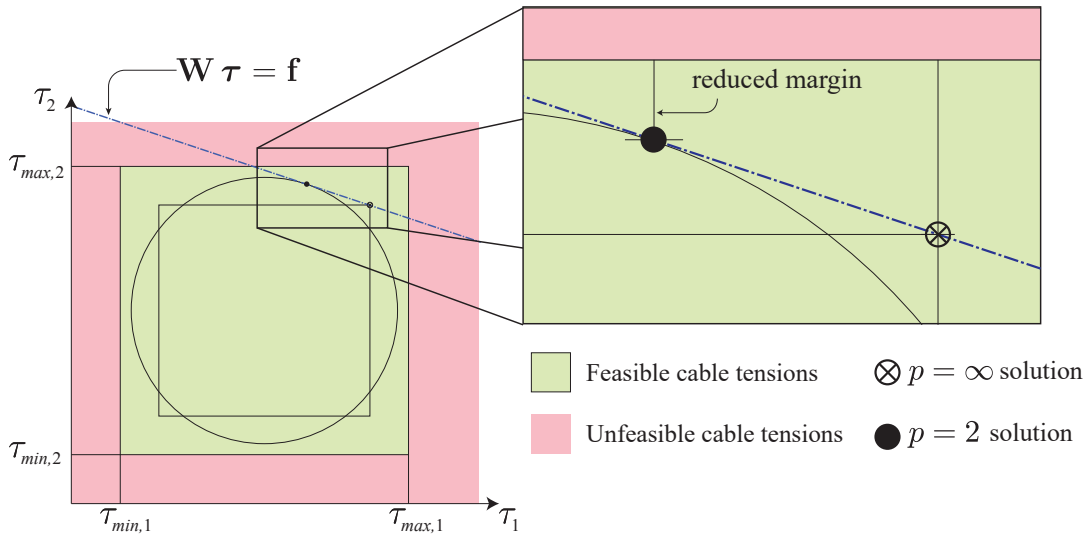

Figure 1.12 – *Illustration that, in general, a cost function $\|\boldsymbol{\tau} - \overline{\boldsymbol{\tau}}\|_2$ (i.e. $p = 2$) leads to a smaller margin between cable tensions and unfeasible tensions than its counterpart with $p = \infty$ [9].*

Besides the case of $f(\boldsymbol{\tau}) = \sum c_i \tau_i$, the LP methodology may be used to minimize the maximum element of the vector $\boldsymbol{\tau}$ (also known as minimax problem). As a matter of fact, any minimax problem with linear constraints may be solved through an equivalent LP problem [92]. Based on this approach, a versatile method is proposed in [93] to minimize $f(\boldsymbol{\tau}) = \max \tau_i$. Since $\max_i \tau_i = \lim_{p\to\infty} \|\boldsymbol{\tau}\|_p$, the notation $\|\boldsymbol{\tau}\|_\infty := \max_i \tau_i$ is commonly used. Moreover, the proposed method is versatile in the sense that it is able to maximize the minimal cable tension. Nevertheless, as highlighted in this study, TDAs based on norms with $p = 1$ or $p = \infty$ may lead to discontinuities in the profiles of cables tensions, *i.e.* infinitesimal changes in $\mathbf{f}_d$ may lead to finite changes of the $\boldsymbol{\tau}^*$ obtained with the corresponding TDAs. Clearly, additional constraints may be taken into account in the optimization problem (1.1) so that the variation of cable tension remains limited [91].

In contrast, methods taking $f(\boldsymbol{\tau}) = \|\boldsymbol{\tau}\|_p$ with $1 < p < \infty$ (with *strict* inequalities) intrinsically lead to continuous profiles of cable tensions. Gosselin presents an overview on this matter in [9] emphasizing this advantage. On the other hand, assuming that the presence of cable tensions in the neighborhood of the bounds $\boldsymbol{\tau}_{min}$ and $\boldsymbol{\tau}_{max}$ should be avoided, he points out that it may be preferable to minimize $f(\boldsymbol{\tau}) = \|\boldsymbol{\tau} - \overline{\boldsymbol{\tau}}\|_\infty$ with $\overline{\boldsymbol{\tau}} = {}^1\!/_2\,(\boldsymbol{\tau}_{min} + \boldsymbol{\tau}_{max})$. It is shown that this approach, compared to the application of other values of p, tends to maximize the difference between the desired cable tensions and the cable tension limits, preventing the incidence of values close to $\boldsymbol{\tau}_{min}$ and $\boldsymbol{\tau}_{max}$ (see Figure 1.12). Inspired by this advantage, $p = 4$ is proposed in order to ally continuity of cable tensions and increased difference between $\boldsymbol{\tau}^*$ and bounds $\boldsymbol{\tau}_{min}$ and $\boldsymbol{\tau}_{max}$. Figure 1.13 illustrates that this value of p may be used to this end. Moreover, an explicit expression for the TDA taking $m = n + 1$ and $f(\boldsymbol{\tau}) = \|\boldsymbol{\tau} - \overline{\boldsymbol{\tau}}\|_4$ is presented in [9].

In spite of the advantages highlighted in [9] led by the use of higher values of p, the majority of studies addressing TDAs with the format (1.1) is focused on values of $p \leqslant 2$. Besides the above-mentioned works addressing the case in which $p = 1$, particular attention has been devoted to the problem of TDAs based on the 2-norm of the vector of cable tensions [94–98]. In addition to the fact that such algorithms intrinsically delivers

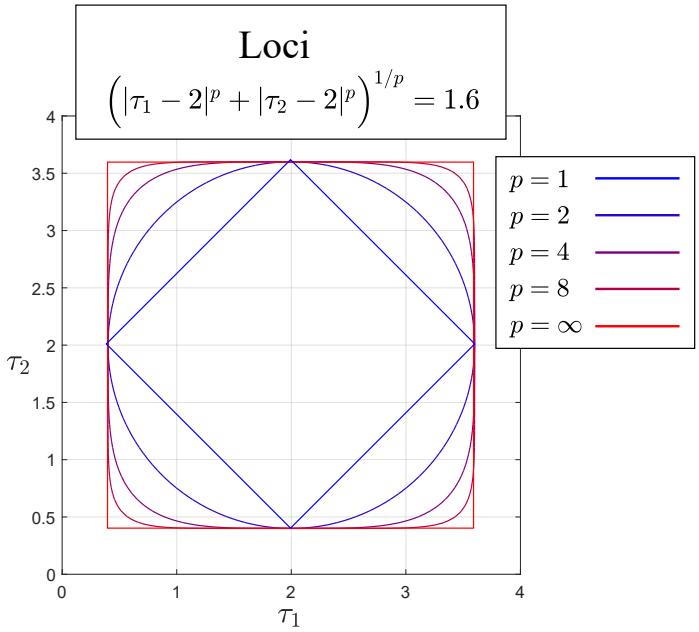

Figure 1.13 – *Comparison of Loci with constant* $\|\boldsymbol{\tau} - \overline{\boldsymbol{\tau}}\|_p$ *for different values of* p.

continuous profiles of cable tensions, the resulting optimization problems are strictly convex Quadratic Programming (QP) problems [99]. The numerical tractability of this class of optimization problem is an important advantage. Accordingly, the well-known numerical tools [100,101] able to solve QP problems were applied to TDAs minimizing the 2-norm of cable tensions [94, 95]. Alternatively, the Dykstra's Projections Algorithm [97] or direct consequences of the Karush–Kuhn–Tucker (KKT) conditions [102] may be used [98]. All these methods [94–98] are iterative and, therefore, yield a performance highly dependent on the initial guess. Fortunately, since TDAs with 2-norm cost functions deliver continuous cable tensions for continuous variations of desired wrenches, consecutive time samplings should deliver solutions close to each other. This evidence justifies the use of the solution obtained in a sampling time as initial guess for the next sampling time (the so-called *warm start*). Moreover, as discussed in [94], gradient-based algorithms such as the gradient projection method may be used with a limited number of iterations, providing feasible suboptimal solutions if real-time constraints hinder the computation of the global optimal solution.

Even though TDAs may be used in different circumstances, such as the design of CDPRs [103], it is reasonable to assert that the main application of such algorithms is the real-time control. Therefore, the computation time is a major concern. Pott *et al.* proposes in [104] a closed-form expression able to minimize $\|\boldsymbol{\tau} - \overline{\boldsymbol{\tau}}\|_2 = \|\boldsymbol{\tau} - \overline{\boldsymbol{\tau}}\|$. Besides the short computation time, this method presents an attractive simplicity of implementation. Additionally, in contrast to iterative methods, the closed-form solution demands well defined floating point operations, which may be used to bound the worst case of the computational burden. Nevertheless, even though an improved algorithm was introduced in [105], these methods do not have a clear proof that a feasible distribution may be found for any feasible desired wrench.

Alternatively to norms of the vector of cable tensions, indirect consequences of the changes in $\boldsymbol{\tau}$ may be addressed as well. In particular, changes in the internal forces

may result in variations on the stiffness of the CDPR [106–108]. The influence of the cable tensions on the robot stiffness is studied under different approaches [109, 110]. Probably, the most commonly applied approach considers cables as springs with a profile consisting of a straight line [109]. Using this model, the stiffness along a given direction may be optimized as proposed in [107].

Moreover, other methods not based on the formulation (1.1) were also proposed. Many studies are based on the characterization of the set formed by the constraints in (1.1). This is the set of feasible cable tensions that generate the desired wrench. As discussed in [111, 112], this set is a polytope which is a subset of $\mathbb{R}^m$. As any polytope, it may be described by its vertices. Aiming at the avoidance of cable tensions close to tension limits, Mikelsons *et al.* proposes the determination of the barycenter of such polytope in [113]. The method is proved to be continuous, but the computation of the polytope itself is not discussed in depth. Accordingly, Gouttefarde *et al.* introduces a method in [114] (with preliminary study published in [115]) able to compute this set for the specific case of $m = n+2$. This approach proved to be versatile in the sense that several TDAs may be adopted based on the output delivered by such algorithm. Explicit computation of the following outputs are presented: centroid, weighed barycenter, 1-norm and 2-norm optimal solutions. The algorithm finds the values of $\boldsymbol{\lambda} \in \mathbb{R}^2$ so that $\boldsymbol{\tau} = \mathbf{W}^+ \mathbf{f}_d + \mathbf{N}\,\boldsymbol{\lambda}$ are the vertices of the polytope of the TDA solutions, with $\mathbf{N} \in \mathbb{R}^{m\times 2}$ the null space matrix of $\mathbf{W}$. Therefore, the identification of the polytope vertices lying on $\mathbb{R}^8$ is simplified to computations in the 2-dimensional space of $\boldsymbol{\lambda}$.

1.6 Cable Tension Control

Inasmuch as the output of the TDAs is a set of desired cable tensions, a strategy should be implemented to convert desired cable tensions to a variable that is meaningful to the motor drivers, typically motor torques (or currents), positions or velocities. Such strategies are denoted in this thesis as cable tension control schemes. Compared to the position tracking control, this problem received less attention among the research studies related to the control of CDPRs. Notably, many works presenting exclusively simulations focused on the position tracking control neglect this subject, considering that the desired cable tensions are instantly generated [71, 72].

Maybe due to the fact that the measurement of forces are often very noisy or because force sensors to measure cable tensions were not available, the majority of published experimental results were obtained with open-loop cable tension control schemes. Figure 1.14 presents a very simple scheme using motor torques as control input to the motor drivers [1]. In spite of its simplicity, this strategy has been widely used. Clearly, this is the case for the schemes in Figures 1.9 and 1.10 of works [7] and [77], respectively. Additionally, such open-loop control is also used in other studies cited previously, *e.g.* [8, 58].

Alternatively, force sensors may be used enabling the implementation of a feedback control strategy. Kraus *et al.* used motor positions as control inputs to the motor

1. The used notations are in accordance with those in Figures 1.9 and 1.10.

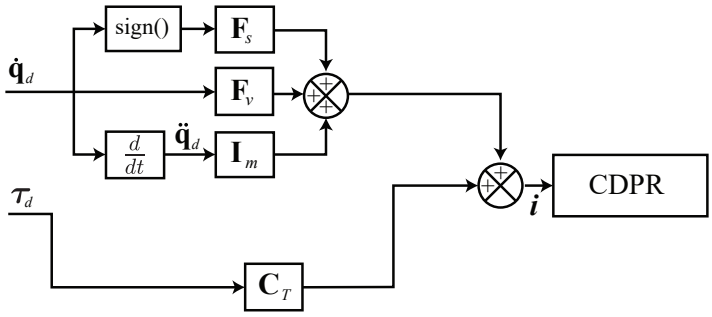

Figure 1.14 – *A simple open-loop cable tension control scheme [8].*

drivers [59]. This work presents details on the identification of the dynamic system and satisfying experimental results on position tracking control. With a very similar cable tension control scheme, the same author implemented a hybrid force-position control [116]. Reichert *et al.* attains a similar goal using torques as control inputs of the motor drivers [117]. Using a disturbance observer, a robust internal force-based impedance control is proposed [117]. A more advanced technique is proposed and tested through numerical simulations in [118] using $\mathcal{H}_\infty$ robust control in order to compensate the influence of the cable elasticity.

The use of measured cable tensions as feedback signals may be of particular interest for fully-constrained CDPRs. Consider a suspended CDPR presenting a pose and payload in accordance with the designed nominal workspace and admissible payloads. Excessively high cable tensions necessarily lead to high accelerations. Therefore, safety of suspended CDPRs regarding the limits of cable tension may be indirectly monitored by other aspects, namely, pose, payload, velocity and accelerations. For this reason, some suspended CDPRs do not present force sensors (even though most of the cranes directly monitor cable tension for safety purposes). In contrast, fully-constrained CDPRs may develop excessively high cable tensions without any other apparent variation on these variables. Therefore, for safety reasons, the use of force sensors monitoring cable tensions of a fully-constrained CDPR is highly recommended. This is the case for the robots studied in this thesis. Accordingly, the implementation of an inexpensive (regarding the cost) cable tension control scheme taking advantage of the presence of the force sensors is pertinent.

1.7 Model Predictive Control

According to the studies presented in Section 1.5, the minimal and maximal cable tensions play a major role in the control of CDPRs. However, one may note that the cable tension limits, in spite of their critical importance, are generally taken into account (in the TDA) once the desired wrench is already computed. In other words, the wrench feasibility cannot be directly considered in the corresponding control approaches. In practice, if a non-feasible wrench is computed by any of the OSC modules [8]-[77], the corresponding TDA will not be able to compute cable tensions, resulting in a sudden interruption of the robot operation. Similarly, the discussed JSC schemes handle the tension limits uniquely in the feedforward terms, as in the scheme in Figure 1.9.

Therefore, one cannot guarantee that the final cable tension is within the feasible limits.

Typically, a non-feasible desired wrench may be necessary in case of excessive disturbances or if the desired pose is out of the actual workspace. Even though the robot workspace can be theoretically computed beforehand, so that the feasible trajectories may be analyzed out of the real-time operation, the proposed workspace computation methods were proven to deliver not more than estimative results. Depending on the considered assumptions and all kind of uncertainties including approximate knowledge of the CDPR geometric parameters, the dimensions of the estimated workspace may change considerably and the computation itself may be troublesome [119, 120]. Moreover, changes on the values and location of the payload lead to variations in the workspace dimensions.

As discussed in Section 1.2, the robot designed in the Hephaestus project should operate in a construction site so that the presence of disturbances is a major concern and safety deserves particular attention. In addition, the limited available area in the neighborhood of the building results in a workspace with small margin for uncertainties, *i.e.* the difference between the workspace predicted in the robot design and the necessary to perform the tasks is very narrow. In this context, the operation of the Hephaestus prototype with a control scheme similar to state of the art approaches [8]-[77] would be probably subjected to the sudden interruptions described above, which may lead to unacceptable safety issues. In practice, a control scheme able to resolve this problem should take into account the cable tension limits within the computation of the desired wrench.

Model Predictive Control (MPC) is one of the few control methodologies able to explicitly handle system constraints [121, 122]. The main idea of MPC is well described in the early work of Lee and Markus [123]:

> *One technique for obtaining a feedback controller synthesis from knowledge of open-loop controllers is to measure the current control process state and then compute very rapidly for the open-loop control function. The first portion of this function is then used during a short time interval, after which a new measurement of the process state is made and a new open-loop control function is computed for this new measurement. The procedure is then repeated.*

In other words, an open loop optimal control problem is repeatedly solved based on updated measured states and the first portion of the optimal sequence is applied during one control cycle time. The procedure is repeated at each control cycle. As better discussed in the course of the present section, this optimal control problem may explicitly handle the system constraints.

Although the main idea of MPC is remarkably simple, its implementation relies on the solution of an optimal control problem in real-time. Clearly, considering the numerous possible combinations of system dynamics and strategies to solve the corresponding optimal control problem, the class of MPC schemes includes a variety of control strategies with characteristics remarkably different. This section tries to summarize these main characteristics and evaluate their pertinence in the context of this thesis.

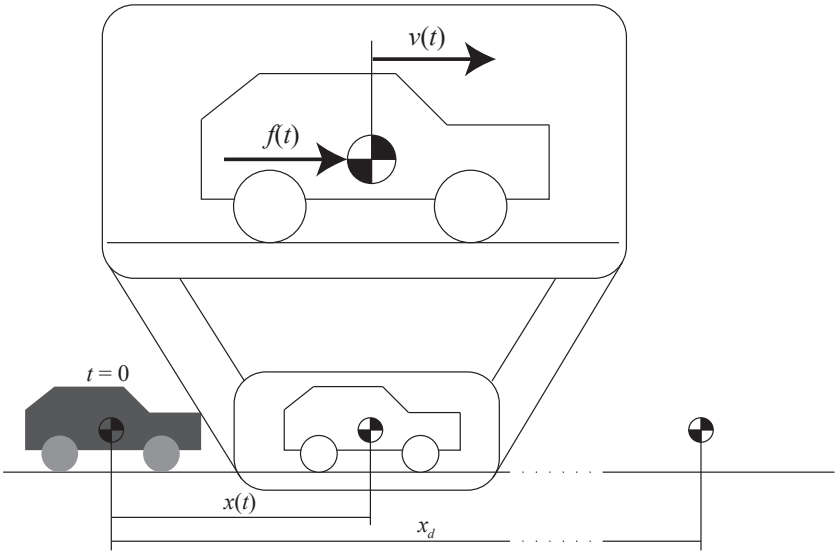

Figure 1.15 – *An illustrative system used to introduce the main elements of an OCP.*

1.7.1 Optimal control

According to Vinter [124], Optimal Control Problems (OCPs) emerged in the 1950s taking the aerospace engineering field of the American and Russian space programs as important source of motivating problems. As an illustrative example, take the very simple system of Figure 1.15. The motion of a body with unitary mass within a one-dimensional space is studied in a time span $t \in [0, t_f]$. The body departs from position $x(0) = 0$ with null initial velocity $v(0) = 0$ and is actuated by a force, namely, $f(t)$. The motion satisfies the Newton's Second Law $\dot{v}(t) = f(t)$ and the system is constrained with respect to the applied force and velocity according to $|f(t)| \leqslant f_{max}$ and $v(t) \leqslant v_{max}$ for all $t \in [0, t_f]$. A twofold goal is defined: The final position $x(t_f)$ should be as close as possible to a desired position x_d and the squared final velocity $v(t_f)^2$ should be minimized as well. Accordingly, the following OCP may be formulated:

$$\begin{aligned} &\min_{f} \ \Big(x(t_f) - x_d\Big)^2 + v(t_f)^2 \qquad . \\ &\text{s.t. } \ x(0) = 0, \ v(0) = 0 \\ &\qquad \dot{x}(t) = v(t), \ \dot{v}(t) = f(t) \\ &\qquad |v(t)| \leqslant v_{max}, \ |f(t)| \leqslant f_{max}, \ \ \forall\, t \in [0, t_f] \end{aligned} \tag{1.2}$$

In spite of its simplicity, this illustrative example introduces the main elements of a general OCP, which may be summarized as follows:

	Present example
• A dynamic system	$\dot{v}(t) = f(t)$
• A time span (bounded or unbounded)	$t \in [0, t_f]$
• System constraints	$\lvert v(t)\rvert \leqslant v_{max}, \ \lvert f(t)\rvert \leqslant f_{max}$
• A cost functional	$\Big(x(t_f) - x_d\Big)^2 + v(t_f)^2$

It is worth noting that, as presented in the list above, the analyzed time span may

be bounded or unbounded. In contrast to this illustrative example that implements a finite time span, the interval of time $t \in [t_0, \infty)$ for a given initial time t_0 is used in OCPs with an unbounded time span. The theory of infinite horizon optimal control [125] addresses this class of OCP. The (LQR) and Linear Quadratic Gaussian (LQG) controllers are well-known methods using this approach.

Even though OCPs based on continuous time systems may be categorized within the class of optimization problems, it is worth highlighting that they present a rather particular characteristic when compared to the classical problems addressed in optimization theory. Instead of taking as argument a finite set of numbers, a continuous OCP take as argument a *vector function*, namely, the control inputs within the studied time span ($f(t)$ in the example above). This is the reason why the term cost *functional* (instead of cost function) was used in the list of the main elements of an OCP. As defined in [126], a functional is a "correspondence which assigns a definite (real) number to each function (or curve) belonging to some class". Briefly speaking, the main goal of a continuous OCP is to minimize (or maximize) a cost functional, which quantifies how good is the obtained behavior. The solution of the problem represents the optimal profile of control inputs.

Once the problem is stated, a method to compute the control inputs as functions of time should be defined. To this end, the variations of the control inputs should be mapped to the variations of the cost functional. This subject is addressed by means of the Variational Calculus, which constitutes the theoretical basis to the solution of classical problems such as the derivation of the seminal Euler-Lagrange equation [126, Section 1.4]. More precisely, the Euler-Lagrange equations are differential equations that are necessary conditions to the optimality of a functional. In order to cope with the solution of a general optimal control problem (in particular, the system constraints), the results of the Variational Calculus were extended by Pontryagin [127], who introduced the maximum principle.

As an alternative to these methods, the continuous system may be studied with a discrete-time approach. In this case, continuous-time control inputs and states are converted to *sequences* of control inputs and states. This numerical discretization may be performed using the methods proposed in [13, Chapter 9] and leads to a *discrete-time* system, which is an approximation of the original continuous system.

As an illustration, considering the example introduced earlier, the functions representing the force $f(t)$ and states $\mathbf{s}(t)$ for $t \in [0, t_f]$ are converted to finite sequences $\{f_0, ..., f_{n_f}\}$ and $\{\mathbf{s}_0, \mathbf{s}_1, ..., \mathbf{s}_{n_f+1}\}$, with $\mathbf{s}_k = \begin{bmatrix} x_k & v_k \end{bmatrix}^T$ containing the actual position and velocity, x_k and v_k respectively. In this discrete-time version (in the sense of [128, Section 2.3]) of the aforementioned example, the sequence of states $\{\mathbf{s}_0, \mathbf{s}_1, ..., \mathbf{s}_{n_f+1}\}$ and the sequence of control inputs $\{f_0, f_1, ..., f_{n_f}\}$ satisfy a transition mapping

$$\mathbf{s}_{k+1} = \phi(\mathbf{s}_k, f_k), \ \forall \ k \in \{0, 1, ..., n_f\}, \tag{1.3}$$

and the OCP (1.2) may be reformulated as

$$\begin{aligned} &\min_{f_0,\dots,f_{n_f}} \left(x_{n_f+1} - x_d\right)^2 + v_{n_f+1}^2 \quad . \\ &\text{s. t. } \mathbf{s}_0 = \mathbf{0} \\ &\quad \mathbf{s}_{k+1} = \phi(\mathbf{s}_k, f_k) \; \forall \; k \in \{0, ..., n_f\} \\ &\quad |v_k| \leqslant v_{max}, \; |f_k| \leqslant f_{max}, \; \forall \; k \in \{0, ..., n_f\} \end{aligned} \tag{1.4}$$

This discrete counterpart of the OCP discussed earlier takes as argument a sequence of control inputs, instead of a continuous function. Therefore, the resulting optimization problem may be solved with standard optimization techniques. Furthermore, since the vast majority of control systems used nowadays are digital, the discrete-time approach of OCP led to relevant influences on the current state of the art of the MPC. For this reason, this subject is discussed here more in detail.

The Dynamic Programming Principle (DPP), introduced by Bellman [10], is able to break some optimization problems in several smaller subproblems in a recursive manner. As illustrated in Figure 1.16, the DPP is useful when decisions should be taken successively. This figure introduces the main elements of a problem addressed by this principle. The possible states are distributed in several layers, and successive layers present their states connected by transitions with corresponding costs. The problem of finding the sequence of transitions linking the initial state to the final state with a minimal accumulated cost (*optimal path*) may be divided into several subproblems in which a *cost-to-go* is attributed to each state of a layer. For a given state s, the cost-to-go is the minimal cost to attain the final state departing from the state s.

More precisely, the DPP may be sketched as follows. Denote s_i' the possible states that can be reached directly from s, and c_i the stage cost corresponding to the transition from s to s_i', with $i \in \{1, ..., n_s\}$ and n_s the number of possible states reachable from s. Denote also $V(\overline{s})$ the cost-to-go of any state $\overline{s}$. The DPP states that

$$V(s) = \min_i \left(c_i + V(s_i')\right). \tag{1.5}$$

The applicability of the DPP in discrete-time OCPs is evident. However, taking a continuous system such as the dynamic model of a CDPR, one may note that the application of the DPP as it is involves two instances of discretization. Firstly, the system should be *discrete-time*, *i.e.* time should be considered as a discrete sequence $t \in \{t_0, t_1, ..., t_f\}$, instead of a continuous interval. Secondly, the states and control inputs themselves should be discretized as well. That is to say, instead of considering continuous intervals of admissible cable tensions, platform poses and velocities, one should consider finite discrete sets. As a result, a mesh of possible states and control inputs need to be considered. As already pinpointed by Bellman [129, page 322], this second instance of discretization leads to a severe drawback for systems with high dimensions (this drawback is commonly referred to as the curse of dimensionality). Nevertheless, the DPP plays a very important role as an analysis tool for the performance and stability evaluation of MPC schemes [13]. Moreover, the theory introduced in [10] inspired other MPC approaches that do not require the discretization of states and control inputs, such as the Differential Dynamic Programming (DDP) [130] which has been used in recent and advanced applications of MPC [131].

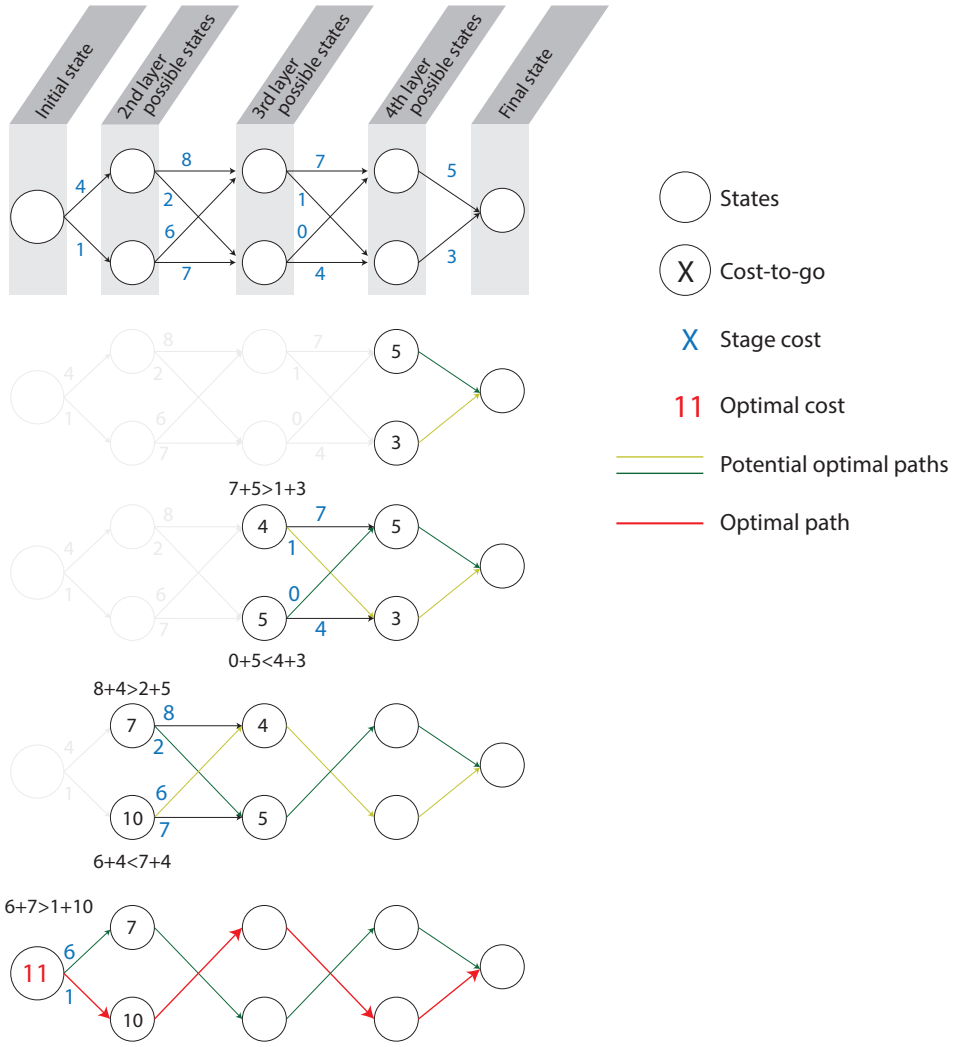

Figure 1.16 – *A simple example of an application of the Dynamic Programming principle introduced in [10].*

1.7.2 Receding horizon model predictive control

Note that the discussions related to optimal control hitherto do not consider a closed-loop strategy. Recalling the discussion at the beginning of this section, the main idea involved in MPC is to recompute the optimal control inputs as frequently as possible taking the measured states into account, leading to a closed-loop algorithm (see the illustration in Figure 1.17). Consequently, an optimal control problem should be solved in each cycle of the feedback loop.

Such as illustrated in Figure 1.17, this concept results in an interesting consequence on the analyzed time span of MPC schemes. First, consider a control scheme that solves repeatedly an OCP with a *finite* time span. In the context of MPC, the final instant of such time span is called *prediction horizon*. Obviously, the initial time of the OCP is shifted every time it is computed. Since the analyzed length of time is kept constant, the prediction horizon is also shifted. This feature presented by MPC schemes solving OCP with finite time spans inspires the commonly used term *Receding Horizon* Model Predictive Control. On the other hand, if infinite time span is used, the horizon is

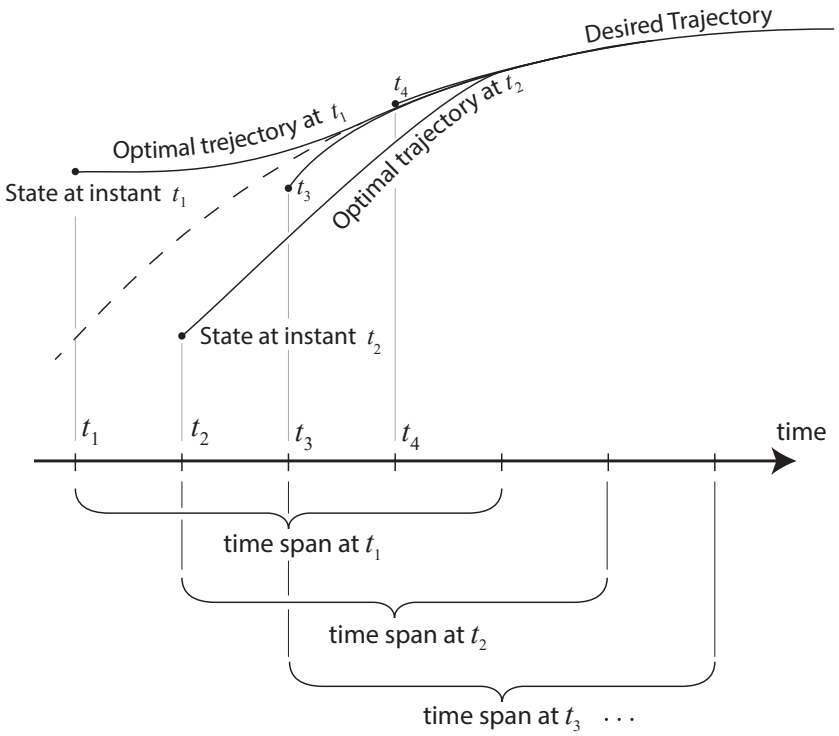

Figure 1.17 – *MPC strategies solve a finite horizon OCP at each control cycle taking updated measured states.*

constant ($t_f = \infty$). Therefore, the aforementioned examples LQR and LQG do not present receding horizons. For this reason, even though this class of feedback control repeatedly solves OCPs taking updated measures of states, they are rarely included within the set of MPC schemes. In brief, the commonly used classification indicates that a control scheme is considered as an MPC scheme only if it presents a receding horizon. Accordingly, the definition of MPC used in the present thesis is as in [132]:

> *Model predictive control (MPC) or receding horizon control (RHC) is a form of control in which the current control action is obtained by solving on-line, at each sampling instant, a finite horizon open-loop optimal control problem, using the current state of the plant as the initial state; the optimization yields an optimal control sequence and the first control in this sequence is applied.*

In spite of the fact that earlier works, such as [133], introduced strategies matching with the aforementioned rationale, the origin of MPC is commonly attributed to Richalet. The work [134] coined the name Model Predictive Heuristic Control to the method in which impulse-response methodology is used to identify and model the controlled system. Promising experimental results were obtained for the control of a steam generator, a distillation column in an oil refinery and a complete chemical plant synthesizing vinyl chloride. Richalet already highlighted the prominent advantage that constraints may be handled explicitly within the feedback loop. Handling constraints is particularly relevant since the optimal operation in industrial processes are frequently attained on the boundary of the set of feasible states and control inputs. However, some strategies which cannot be neglected on the timeline of the development of MPC schemes do not consider system constraints, *e.g.* the Generalized Predictive Control (GPC) proposed by Clarke [135].

Since Richalet's work [134], MPC has been the focus of a great deal of attention both in academia and industry. Indeed, as advocated by Maciejowski [121], MPC is sometimes known as "the only advanced control technique — that is, more advanced than standard PID — to have had a significant widespread impact on industrial process control". Here, it is interesting to highlight the scope of *industrial process control*. Besides the evident profitability obtained with a control scheme able to safely run close to constraints, industrial processes often present system dynamics much slower than other typical applications of advanced control strategies, such as robotics. Systems with slow dynamics enable the application of control running with longer cycle times. Therefore, issues related to the numerical solution of OCPs in real-time may be overcome. For these reasons, the vast majority of relevant applications of MPC in the 20th century was focused on industrial process control, or more specifically, to chemical processes [136–138].

Fortunately, thanks to the improvements on the numerical optimization methods and on the computational capability of the commercial hardware, the applicability of MPC has been extended to systems with fast dynamics. As a matter of comparison, in 1990, Hidalgo and Brosilow used a sampling time of 1 min [136], whereas numerous recent studies apply intricate MPC schemes on robots with sampling time of a few milliseconds, *e.g.* the trajectory optimization proposed in [139] that may be computed within 25 ms. Accordingly, MPC is a mature subject within the field of robotics, *e.g.* [139–150].

MPC schemes may be categorized according to several classes. The classes that are relevant to the discussions present in this thesis are discussed in the sequel. Firstly, as for the control strategies discussed in Section 1.4, several simplifications can be considered if the system to be controlled is assumed to be linear. In this case, the resulting strategy is classified as linear MPC. Even though linear systems represent a very specific class of system among the broad set of possible applications of MPC, a great part of the most well-known books dedicated to the MPC theory are focused exclusively on linear MPC [121,122,151]. For this reason, linear MPC is commonly referred to as MPC, shortly. Otherwise, if the control scheme considers a nonlinear system, it is classified as Nonlinear MPC (NMPC) [13,152].

In the scope of linear MPC, linear systems generally lead to a simple analytical expression for the cost functional optimized in the OCP. Revisiting the example of Figure 1.15, the corresponding transition mapping ϕ in (1.3) may be written as

$$\mathbf{s}_{k+1} = \overbrace{\begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}}^{\mathbf{A}} \mathbf{s}_k + \overbrace{\begin{bmatrix} \Delta t^2/2 \\ \Delta t \end{bmatrix}}^{\mathbf{B}} f_k. \tag{1.6}$$

This expression may be recursively applied so that

$$\begin{aligned} \mathbf{s}_1 &= \mathbf{B}\, f_0 \\ \mathbf{s}_2 &= \mathbf{A}\,\mathbf{B}\, f_0 + \mathbf{B}\, f_1 \\ &\vdots \\ \mathbf{s}_{n_f+1} &= \sum_{i=0}^{n_f} \left(\mathbf{A}^{n_f - i} \mathbf{B}\, f_i \right) = \overline{\mathbf{B}}\, \boldsymbol{f}, \end{aligned} \tag{1.7}$$

with $\overline{\mathbf{B}} = \begin{bmatrix} \mathbf{A}^{n_f}\mathbf{B} & \mathbf{A}^{n_f-1}\mathbf{B} & \dots & \mathbf{B} \end{bmatrix}$ and $\boldsymbol{f} = \begin{bmatrix} f_0 & \dots & f_{n_f} \end{bmatrix}^T$. As a result, the following explicit expression for the functional in (1.4) may be obtained

$$\begin{aligned} \left(x_{n_f+1} - x_d\right)^2 + v_{n_f+1}^2 &= \left(\overline{\mathbf{B}}\,\boldsymbol{f} - \begin{bmatrix} x_d \\ 0 \end{bmatrix}\right)^T \left(\overline{\mathbf{B}}\,\boldsymbol{f} - \begin{bmatrix} x_d \\ 0 \end{bmatrix}\right) = \\ &= \boldsymbol{f}^T \overline{\mathbf{B}}^T \overline{\mathbf{B}}\,\boldsymbol{f} - 2 \begin{bmatrix} x_d & 0 \end{bmatrix} \overline{\mathbf{B}}\,\boldsymbol{f} + x_d^2, \end{aligned} \tag{1.8}$$

which is a quadratic function of the vector of forces $\boldsymbol{f}$. Similarly, the constraints in (1.4) may be written as linear constraints in $\boldsymbol{f}$. As a result, the OCP (1.4) may be rewritten in the format

$$\begin{aligned} &\min_{\boldsymbol{f}} \ \boldsymbol{f}^T \mathbf{Q}\,\boldsymbol{f} + \mathbf{c}^T\,\boldsymbol{f}, \\ &\text{s.t.} \ \ \mathbf{A}_{ineq}\,\boldsymbol{f} \leqslant \mathbf{b}_{ineq} \end{aligned} \tag{1.9}$$

with constant matrices $\mathbf{A}_{ineq}$ and $\mathbf{Q}$, and constant vectors $\mathbf{b}_{ineq}$ and $\mathbf{c}$, so that the studied OCP is actually a QP problem. Moreover, the fact that the hessian matrix $\mathbf{Q} = \overline{\mathbf{B}}^T\overline{\mathbf{B}}$ is positive semidefinite demonstrates that the optimization problem is convex.

As in this illustrative example, MPC schemes very often use cost functionals that are quadratic functions of the states and control inputs. For the sake of comparison, among all the previously cited references of MPC applied to robotics [139, 142–150], the work of Worthmann *et al.* [149] is the only one that does not belong to this class of MPC.

The main characteristic illustrated in the OCP (1.9) is that a cost functional which is a quadratic function of the states and control inputs applied to linear systems may be optimized with standard QP algorithms. Recalling that, as discussed in Section 1.5, numerous software can be used to efficiently solve QP problems (see [99, Section 16.8]), the numerical solution of such OCPs may be seen as an elementary task. On the other hand, the same kind of cost functional applied to nonlinear systems leads to OCPs demanding nonlinear numerical optimization methods.

Based on this contrast between linear MPC and NMPC, the first control solution proposed in this thesis applies linear MPC (Chapter 3). Numerical simulations and experimental studies proved that the proposed solution yields to satisfying results. Nevertheless, in order to guarantee stability and good performance independent of the level of nonlinearities of the CDPR dynamics, an NMPC scheme is necessary. For this reason, additional studies presented in Chapter 4 address this type of MPC. Both approaches cope with the issue discussed at the very beginning of this section, *i.e.* the cable tension limits are explicitly handled in the OCP formulated within the MPC scheme. As a result, the tension distribution is integrated within the main feedback control. Moreover, the proposed scheme computes feasible sets of cable tensions even for reference trajectories that escape from the robot workspace.

The majority of studies addressing the stability and robustness of MPC schemes is focused on the so-called tracking MPC (or stabilizing MPC), which penalizes the deviation to a predefined *feasible* equilibrium or trajectory [153]. More precisely, sequences of desired states and desired control inputs should be computed beforehand. The cost functional should penalize the euclidean distance to these desired sequences. Therefore, the cost functional should be null for states and control inputs equal to their

desired counterparts, and strictly positive otherwise. The illustrative example of Figure 1.15 can be used to show the typical approach implemented to obtain a standard tracking MPC.

The control scheme considers sequences of desired states $\{\mathbf{s}_{d,0},\ \mathbf{s}_{d,1}, \ldots, \mathbf{s}_{d,n_f+1}\}$ and forces $\{f_{d,0}, \ldots,\ f_{d,n_f}\}$ determined beforehand. These sequences represent the desired behavior of the system. The sequence of desired states determines the reference trajectory departing from $\mathbf{s}_{d,0} = \mathbf{s}_0$ and arriving on $\mathbf{s}_{d,n_f+1} = \begin{bmatrix} x_d & 0 \end{bmatrix}^T$. The sequence $\{f_{d,0}, \ldots,\ f_{d,n_f}\}$ is the sequence of forces that generates the desired trajectory. Clearly, both sequences should satisfy the system constraints. For a given discrete instant k and actual state $\mathbf{s}_a$, a typical standard tracking MPC could apply the OCP

$$\min_{\{f_k,\ldots,f_{k+h}\}} \sum_{i=k}^{k+h} \|\mathbf{s}_i - \mathbf{s}_{d,i}\|^2 + (f_i - f_{d,i})^2 \tag{1.10a}$$

$$\text{s. t. } \mathbf{s}_k = \mathbf{s}_a \tag{1.10b}$$

$$\mathbf{s}_{i+1} = \phi(\mathbf{s}_i, f_i)\ \forall\ i \in \{k, \ldots, k+h\} \tag{1.10c}$$

$$|v_i| \leqslant v_{max},\ |f_i| \leqslant f_{max}\ \forall\ i \in \{k, \ldots, k+h\} \tag{1.10d}$$

where a positive integer h was used in order to determine the prediction horizon. As described before, the cost functional (1.10a) penalizes deviations with respect to the desired trajectory (with the term $\|\mathbf{s}_i - \mathbf{s}_{d,i}\|^2$) and also with respect to the desired forces (with $(f_i - f_{d,i})^2$). Moreover, if the actual state is equal to the desired one ($\mathbf{s}_a = \mathbf{s}_{d,k}$), the sequence of desired forces within the prediction horizon $\{f_{d,k}, \ldots,\ f_{d,k+h}\}$ leads to

$$\begin{aligned} &f_i = f_{d,i} \text{ and } \mathbf{s}_i = \mathbf{s}_{d,i}\ \forall\ i \in \{k, \ldots, k+h\}, \Rightarrow \\ &\Rightarrow \sum_{i=k}^{k+h} \|\mathbf{s}_i - \mathbf{s}_{d,i}\|^2 + (f_i - f_{d,i})^2 = 0 \end{aligned} \tag{1.11}$$

resulting in a null cost functional in (1.10). If any of the elements of $\{\mathbf{s}_k, \ldots, \mathbf{s}_{k+h}\}$ and $\{f_k, f_{k+h}\}$ is not equal to their desired counterparts, the cost functional is strictly positive. This is the desired characteristic described above.

Note, however, that such approach requires the definition beforehand of the desired sequence of control inputs ($\{f_{d,0}, \ldots,\ f_{d,n_f}\}$ in the example above). The MPC schemes proposed in this thesis perform the tension distribution as an integral part of the main controller. Therefore, the sequence of control inputs (cable tension vectors) is not known beforehand. Accordingly, an alternative approach is proposed in Chapter 4 in order to obtain a standard tracking NMPC with a similar characteristic as in (1.11). The proposed NMPC scheme allows the stability analysis of the corresponding closed-loop system. Even though the design and stability analysis of the proposed NMPC is based on the feasibility of the desired trajectory, numerical simulation results show that the controller is able to operate with unfeasible desired trajectories.

The class of tracking NMPC algorithms may be divided between control schemes (i) *with* and (ii) *without* terminal conditions. An NMPC scheme with terminal conditions (i) solves OCPs with additional *costs* and *constraints* related to the terminal states so that the cost-to-go function can be easily proved to be a Lyapunov function. Accordingly, NMPC schemes with terminal conditions typically lead to simpler stability and

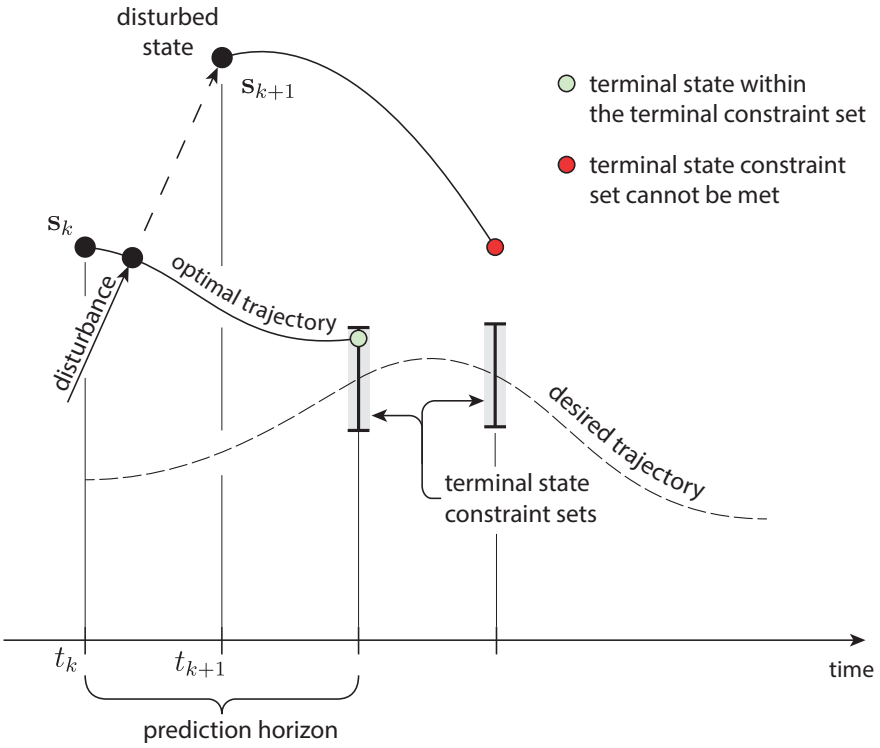

Figure 1.18 – *Illustration of feasibility issues resulted from the application of terminal constraints under the incidence of disturbances.*

performance analysis [132] compared to algorithms without terminal constraints [154]. Nevertheless, the application of terminal conditions leads to crucial drawbacks in the context of the present thesis. As a matter of fact, "*the construction of such stabilizing* (terminal) *constraints may be challenging and their use may considerably reduce the operating range of the MPC scheme*" [155]. The design of the terminal constraints may be particularly troublesome when the disturbances and modeling errors are considered. As illustrated in Figure 1.18, an NMPC of type (i) forces the terminal state in the prediction horizon to be within a terminal constraint set consisting in a neighborhood of the desired trajectory. In the incidence of significant modeling errors and disturbances, feasibility issues may be met. Recalling the considered requirements for the Hephaestus project, one may note that this limitation may lead to safety issues.

Conversely, even though the stability analysis of tracking NMPC schemes without terminal conditions also assumes feasibility of the desired trajectory, controllers of type (ii) are typically less prone to the incidence of unfeasible OCPs. This subject is discussed in detail in [154, 156] and in [13, Section 7.4]. Grüne presents an intuitive example in [154] illustrating the applicability of NMPC with unfeasible desired states.

In addition, terminal constraints may hinder the numerical solution of the OCP [13] and the implementation of NMPC schemes of type (ii) is typically easier than (i) [157]. Recalling that the numerical implementation of the proposed MPCs should be performed from scratch, these advantages also play an important role. Therefore, the control strategies proposed in this thesis are MPC schemes *without* terminal conditions. This choice is in accordance with the MPC schemes commonly used in the industry. Indeed, MPC schemes with terminal constraints "*are mostly not used in the process in-*

dustries" [156].

Apart from tracking NMPC, it is also interesting to highlight an alternative MPC architecture. Similarly to the control strategies in Chapters 3 and 4 of this thesis, the use of MPC to track unreachable setpoints was also addressed in [158]. A few years latter, the term Economic MPC (EMPC) has been coined referring to MPC algorithms that do not penalize the deviation to a feasible equilibrium (or trajectory) [159, 160]. In other words, MPC schemes that do not match with the definition of standard tracking MPC may be categorized as EMPC. The common use of cost functions evaluating the economic performance of a system inspired the expression "economic". As a matter of fact, as discussed in [13], the cost function in an EMPC can "*model all kinds of quantities, like energy consumption, yield of a substance, income of a firm, etc., which one would like to minimize or maximize*". Considering that the ability to track unfeasible desired trajectories is one of the main goal in this manuscript, the control strategies proposed in this thesis may be also classified as EMPC schemes.

As for standard tracking NMPC, the class of EMPC algorithms may be divided between control schemes *with* [160–162] and *without* [163, 164] terminal conditions. Similarly to the strategy described above, an EMPC scheme applying terminal conditions takes advantage of a terminal cost that is a Lyapunov function within the terminal state constraint set. The same disadvantages described before are valid for EMPC schemes with terminal conditions. Stability and performance estimations of this class of NMPC are closely related to the theory of dissipativity of nonlinear systems, which was introduced in [165] and firstly used within the framework of EMPC in [159]. Whereas the stability conditions of systems with a constant unfeasible reference are well-known [164, 166], the case for time-varying references is still an active field of research. Most of these studies are focused on periodical reference trajectories [163, 167, 168]. Some exceptions are the works [169, 170], which address the case for general optimal operation regimes.

Independently of the aforementioned classes, MPC schemes involve the numerical solution of OCPs in real-time. On the other hand, the vast majority of control schemes discussed in Section 1.4 defines a precomputed policy. More precisely, studies [8]-[77] define an explicit expression to the computation of the desired wrenches. The comparison between the complexity involved in these two approaches reveals an important drawback related to MPC schemes. As discussed in the end of Section 1.2, the control schemes developed in this thesis were implemented in a software commonly used in industrial environments. Since such software is not compatible with common programming libraries, the control schemes proposed here were implemented from scratch. This is a minor aspect in the scientific perspective, but crucial from a practical point of view.

Previous studies addressing the implementation of MPC in the control of CDPRs are very few and recent. Katliar *et al.* proposed a nonlinear MPC for a motion simulator in [171]. Its performance was investigated through numerical simulations. To the best of our knowledge, no experimental results were published so far. The feasibility of the MPC implementation in real time was addressed taking into account solvers such as HPMPC [172] and qpOASES [173]. However, these solvers are not compatible with common industrial real-time environments. As a motion simulator control, the focus

in [171] is a set of desired accelerations and velocities. The present thesis is focused on trajectory tracking, which prioritizes positioning accuracy.

The recent work [174] presents simulations and experimental results of a linear MPC scheme implemented in a hybrid cable-driven robot. However, the studied cable-driven system consists of a 2-DoF platform with planar movements. Only some of the cables are controlled in tension. The remaining cables are controlled in position. The actual tensions of these cables are not considered in the real-time controller and the proposed MPC strategy is used to control the vibration of the system. This thesis present results on trajectory tracking of CDPRs with six DoF, eight cables, and each cable tension is controlled by the MPC scheme. Additionally, the focus of the work [174] is vibrations attenuation. One may note that both schemes proposed in [171, 174] cannot be classified as position tracking control.

1.8 Contributions of the Thesis

Based on the state of the art discussed in Sections 1.4, 1.5, 1.6 and 1.7, MPC proved to be an appropriate strategy to the context introduced in Section 1.2. The state-of-the-art control strategies do not explicitly consider the cable tension limits in the computation of the feedback correction. Therefore, feasibility issues may be faced when the robot operates close to the system constraints. These risks present particular relevance in this thesis. Being one of the few control strategies able to explicitly handle the system constraints, MPC corresponds with the specific requirements in the Hephaestus project. Therefore, this thesis proposes different position tracking control solutions using MPC. As a matter of fact, the integration of the TDA within the main feedback control computation is a relevant achievement for the control of CDPRs, in general. Details on the design, numerical simulations, implementation, and real-time experimental results related to two MPC schemes are presented.

Chapter 3 introduces a linear MPC scheme, highlighting its main advantage: The control scheme is able to perform the tracking of unfeasible trajectories satisfying the constraints of feasible cable tensions. In other words, the performed trajectory is as close as possible to the desired one without violating the interval of feasible cable tensions. To the best of our knowledge, this controller led to the first experimental results of a CDPR control scheme with this characteristic. Additionally, characteristics related to disturbance rejection capabilities are also discussed.

Acknowledging the nonlinearities intrinsically inherent to the dynamics of a CDPR, Chapter 4 addresses an NMPC. Whereas this control scheme is inspired by the strategy of Chapter 3, it seeks guarantees of stability and performance based on the theoretical studies of tracking NMPC. Consequently, a more intricate scheme is proposed and its stability analysis is performed. Similarly to the linear MPC of Chapter 3, terminal conditions are not used. The output of both MPC schemes is a set of desired feasible cable tensions for each sampling time.

Even though the results presented in Chapters 3 and 4 represent the main contributions of this thesis, the corresponding implementation in experimental set-ups would

not have been possible without the development of some other technical solutions. Some of these technical solutions are discussed in Chapter 2.

Since the output of the proposed predictive controllers is a set of desired cable tensions, a strategy should be implemented to convert desired cable tensions to a variable that is meaningful to the motor drivers, *e.g.* motor torques, positions or velocities. As discussed in Section 1.6, the CDPRs studied in this thesis have force sensors in order to measure the cable tensions. Therefore, even though the majority of published studies address this problem with open-loop cable tension control schemes, measurements obtained with the force sensors may be used in order to guarantee that the actual tensions are as close as possible to the desired ones. Nevertheless, the CDPRs considered in this thesis have servo motors coupled to gear trains with a large reduction ratio. Real-time experiments described in Chapter 2 demonstrate that the influence of the friction introduced by these gears is non-negligible and scarcely repeatable. Inasmuch as every feedback state of the art cable tension control method discussed in Section 1.6 is highly dependent on a precise identification of the system dynamics, a non-model based scheme was developed. Chapter 2 presents the design of this controller and the corresponding experimental results.

In addition, in accordance with the priorities enumerated in Section 1.2, an efficient estimation of the pose of the platform should be used in order to improve the positioning accuracy of the CDPR. As can be seen in Table 1.1, a relatively large cable diameter (12 mm) is necessary due to the the large payload presented in the Hephaestus project. As a result, a large pulley diameter is necessary in order to satisfy the minimal curvature specific for this cable, which increases the influence of the pulley kinematics discussed in Section 1.3. Therefore, a forward kinematics algorithm considering the pulley kinematics is proposed in Chapter 2. The main contribution here consists of explicit expressions of the corresponding differential kinematics. Such expressions facilitate the implementation of an efficient algorithm using QR factorization.

One may note that implementation of the methods discussed in Chapters 2, 3 and 4 demands advanced numerical algebra, *e.g.* QP, QR factorization, null space computation, among others. Since industrial software and hardware should be used (see Section 1.2), the real-time operation of the programmable logic controller (PLC) is most often not compatible with existing libraries of advanced mathematical operations. Consequently, the necessary operations were programmed from scratch using C++ language compatible with the used industrial software (TwinCAT - Beckhoff). The Appendix A is devoted to the these implementations.

Additionally, Appendix B introduces a TDA able to minimize the infinity norm of the vector of cable tensions. In contrast to most of the algorithms compatible with $m > n + 1$, the proposed TDA consists in a closed-form expression. The optimality of the proposed algorithm is proved and experimental results are presented. The discontinuity inherent of TDAs minimizing the infinity norm is exemplified.

2

Kinematic Model and Cable Tension Control

Several variables considered in the position tracking control strategies proposed in Chapters 3 and 4 do not have a direct relation with variables that can be directly measured and controlled in the CDPRs studied in this thesis. The variables that can be directly measured and controlled in real-time are the motor currents, positions and velocities. Interestingly, the model on which the control is based presents numerous parameters and variables that are not directly related to the motors operation. The present chapter is devoted to the development of strategies and algorithms relating these variables, which are a priori abstract to the control system, with the variables that can be directly accessed through the motor drives. Section 2.1 introduces an algorithm able to estimate the platform poses and velocities based on the motor positions and velocities. Sections 2.2 and 2.3 discuss two different control strategies responsible to generate the desired cable tensions. In the first one, the control inputs are the motor currents whereas the second one is based on motor velocities.

Contents

2.1 Kinematic Model

The main goal of this section is to develop a method with which the most probable platform pose and velocity could be estimated for given measured motor positions and velocities. On the one hand, precise estimations are necessary in order to obtain an accurate position tracking control algorithm. On the other hand, since this estimation should be done in real-time, the computation time is a major concern.

Even though Pott showed in [35] that relevant errors may be obtained assuming fixed proximal anchor points, most of the previous works discussed in Section 1.3 do not address the influence of the pulley kinematics. An exception is the work of Schmidt [175], in which the FK problem considering the pulley kinematics is solved but the equations are not derived explicitly.

Accordingly, the present section explicitly derives the differential kinematics for CDPRs considering the pulley kinematics and, based on that, proposes a real-time capable Forward Kinematics (FK) algorithm. The algorithm applies iteratively QR decomposition to solve a linearized version of the least squares problem representing the FK. Differential kinematics consists here of a closed-form expression of the jacobian matrix of CDPRs considering the pulley kinematics. This jacobian matrix is used to construct the linearization of the FK problem. Experimental and numerical results address the convergence capabilities of the proposed algorithm.

Following the typical approach for the kinematic modeling of parallel robots, the kinematic constraints are firstly formulated by means of the inverse kinematics.

2.1.1 Inverse kinematic model

A 6-DoF CDPR is considered but the proposed algorithm can be applied to CDPRs with fewer DoF. The platform is driven by m cables. The main variables and parameters of the kinematic model are introduced in Figure 2.1. The position of each cable attachment point B_i on the platform is given by the vector $\mathbf{b}_i$, with $i = 1, ..., m$. Each point A_i is a fixed point which lies on the circumference of pulley i. Each pulley i is able to rotate around the line passing through A_i tangent to the pulley circumference. The constant position of A_i is given by vector $\mathbf{a}_i$ in the fixed reference frame.

The platform pose is given by the vector $\mathbf{x} = \begin{bmatrix}\mathbf{p}^T & \boldsymbol{\psi}^T\end{bmatrix}^T$. The position of the platform is $\mathbf{p} = \begin{bmatrix}p_x & p_y & p_z\end{bmatrix}^T$. Its orientation is represented by the vector $\boldsymbol{\psi} = \begin{bmatrix}\alpha & \beta & \gamma\end{bmatrix}^T$. The elements of this vector are Euler angles so that $\mathbf{b}_i = \mathbf{p} + \mathbf{R}_z(\gamma)\,\mathbf{R}_y(\beta)\,\mathbf{R}_x(\alpha)\,\mathbf{b}_{pi}$, where $\mathbf{b}_{pi}$ is the vector of the platform attachment point i written in the coordinate system $\mathcal{O}_p$, attached to the platform, and $\mathbf{R}_x$, $\mathbf{R}_y$ and $\mathbf{R}_z$ are the rotation matrices around x, y and z axis, respectively. The vector of cable lengths is $\boldsymbol{l} = \begin{bmatrix}l_i & ... & l_m\end{bmatrix}^T$, where l_i is the length of cable i and the cable elongations are neglected. The inverse kinematics model gives the motor positions for a given pose of the platform, *i.e.* compute $\boldsymbol{l}$ for a given $\mathbf{x}$. The remainder of this section recalls the inverse kinematics considering the

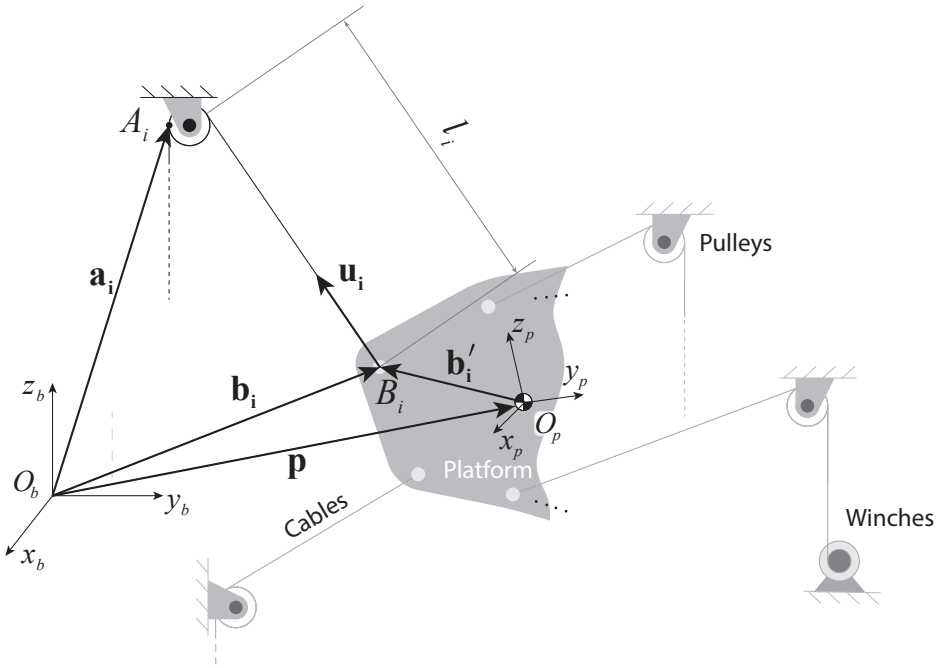

Figure 2.1 – *Illustration of the main geometric notations.*

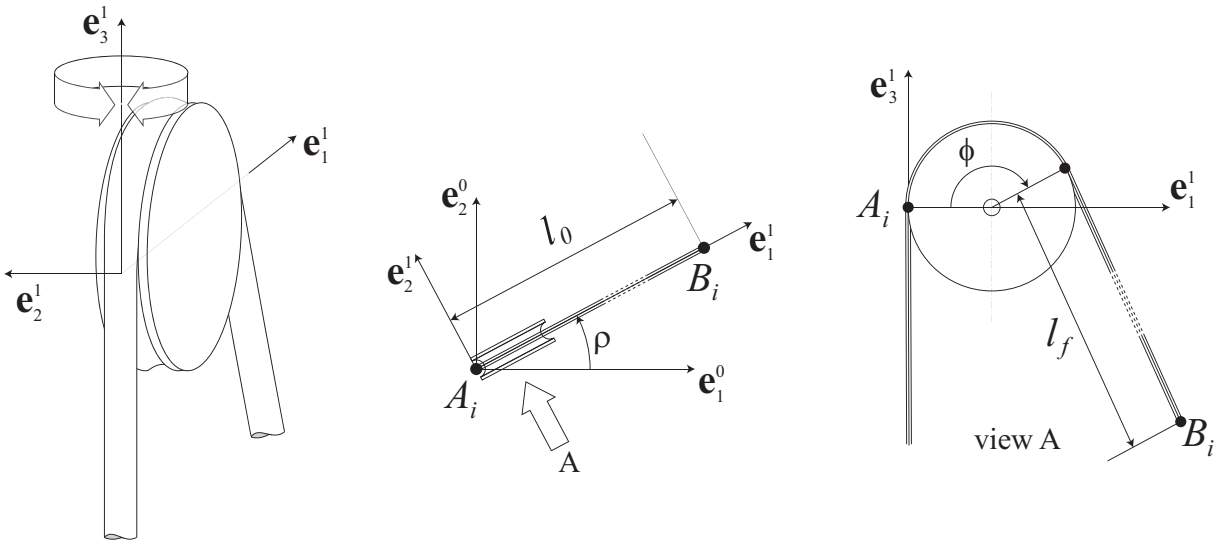

Figure 2.2 – *Pulley geometry and coordinate systems.*

influence of the pulley geometry. Similar models were notably presented in [35, 53].

The pulley geometry is illustrated in Figures 1.7 and 2.2. One drawing pulley i among the m CDPR pulleys is considered and subscript i is dropped in the remainder of this section. Consider the definition of coordinate systems $\mathbf{E}_0 = \begin{bmatrix}\mathbf{e}_1^0 & \mathbf{e}_2^0 & \mathbf{e}_3^0\end{bmatrix}$ and $\mathbf{E}_1 = \begin{bmatrix}\mathbf{e}_1^1 & \mathbf{e}_2^1 & \mathbf{e}_3^1\end{bmatrix}$. These coordinate systems are defined with respect to the inertial frame $\mathcal{O}_b$. The pulley can rotate about the fixed axis $\mathbf{e_3^0}$ (with angle ρ). $\mathbf{E}_0$ is any fixed coordinate system with $\mathbf{e}_3^0$ aligned with the rotation axis of the pulley. $\mathbf{E}_1$ rotates with the pulley and $\mathbf{e}_3^1 = \mathbf{e}_3^0$. $\mathbf{E}_1$ is thus pose dependent and its orientation is computed in the inverse kinematics.

Define $\mathbf{v} = \mathbf{b}_i - \mathbf{a}_i$, written in the CDPR fixed reference frame. Vector $\mathbf{a}_i$ is constant

for a given CDPR and $\mathbf{b}_i$ depends on the platform pose. Vector $\mathbf{v}$ can be written in the coordinate system $\mathbf{E}_0$ as $\mathbf{v}^0 = \mathbf{E}_0^T\,\mathbf{v}$. The angle ρ shown in Figure 2.2 is obtained as atan2(v_2^0, v_1^0), with v_1^0 and v_2^0 the first and second components of $\mathbf{v}^0$. Therefore, the coordinate system $\mathbf{E}_1$ is obtained as $\mathbf{E}_1 = \mathbf{E}_0\,\mathbf{R}_z(\rho)$, where $\mathbf{R}_z$ is the rotation matrix around the z axis.

The vector $\mathbf{v}^1$ written in the coordinate system $\mathbf{E}_1$ is $\mathbf{v}^1 = \mathbf{E}_1^T\,\mathbf{v} = \begin{bmatrix} v_1^1 & 0 & v_3^1 \end{bmatrix}^T$. The component $v_2^1 = 0$ because the coordinate system is aligned to the pulley orientation so that vector $\mathbf{v}$ lies in the plan defined by the vectors $\mathbf{e}_1^1$ and $\mathbf{e}_3^1$. This can be seen in Figure 2.2.

Analyzing the geometry presented in Figure 2.2, the following system of trigonometric equations is obtained

$$\mathbf{v}^{1r} = \begin{bmatrix} v_1^1 \\ v_3^1 \end{bmatrix} = l_f \begin{bmatrix} \sin\phi \\ \cos\phi \end{bmatrix} + r_p \begin{bmatrix} 1 - \cos\phi \\ \sin\phi \end{bmatrix}, \tag{2.1}$$

where r_p is the primitive radius of the pulley and l_f is the length depicted in the Figure 2.2. Eliminating the variable l_f, (2.1) leads to

$$v_3^1 \sin\phi + (r_p - v_1^1)\cos\phi - r_p = 0. \tag{2.2}$$

Trigonometric functions $\sin\phi$ and $\cos\phi$ may be written as

$$\sin\phi = \frac{2\,\tan(\phi/2)}{1 + \tan^2(\phi/2)}, \quad \cos\phi = \frac{1 - \tan^2(\phi/2)}{1 + \tan^2(\phi/2)}, \tag{2.3}$$

so that (2.2) leads to

$$\tan^2(\phi/2)\,(v_1^1 - 2\,r_p) + \tan(\phi/2)\,(2\,v_3^1) - v_1^1 = 0. \tag{2.4}$$

Equation (2.4) can be solved as a quadratic equation in $\tan(\phi/2)$. Then, using the inverse tangent function, two solutions are obtained in the interval $[-\pi, \pi]$, namely,

$$\phi_1 = 2\,\text{atan}\left(\frac{\sqrt{{v_1^1}^2 - 2v_1^1 r_p + {v_3^1}^2} - v_3^1}{v_1^1 - 2r_p}\right), \tag{2.5}$$

$$\phi_2 = 2\,\text{atan}\left(\frac{-\sqrt{{v_1^1}^2 - 2v_1^1 r_p + {v_3^1}^2} - v_3^1}{v_1^1 - 2r_p}\right). \tag{2.6}$$

If $v_1^1 > 2r_p$ (typical situation), $\phi_1 > 0$ and $\phi_2 < 0$. In this case the angle respecting the geometry of the problem is ϕ_1. Otherwise, ϕ_2 should be taken.

The length l_f may be easily obtained from (2.1) once ϕ is known. The cable length l_i is given by

$$l_i = r_p\,\phi + l_f + l_{ci}, \tag{2.7}$$

where l_{ci} is a constant additional length. This constant additional length may, for instance, take into account the total distance between fixed pulleys.

Additionally, the unitary vector $\mathbf{u}_i$ shown in Figure 2.1 can be computed as

$$\mathbf{u}_i = \mathbf{E}_1 \, \mathbf{R}_y(\phi) \, \begin{bmatrix} 0 & 0 & -1 \end{bmatrix}^T = -\mathbf{E}_1 \, \begin{bmatrix} \sin\phi & 0 & \cos\phi \end{bmatrix}^T. \tag{2.8}$$

This vector is used in the computation of the wrench matrix, as detailed in Section 3.1.1.

2.1.2 Differential kinematics

The jacobian matrix $\mathbf{J}$ is used in the solution of the FK problem since it relates $\delta\boldsymbol{l}$ to $\delta\mathbf{x}$ as

$$\delta\boldsymbol{l} = \mathbf{J} \, \delta\mathbf{x}, \tag{2.9}$$

where $\delta(\cdot)$ denotes the differential variation of $(\cdot)$. This section presents the analytical expression of the matrix $\mathbf{J}$ obtained from the equations in Section 2.1.1. As in Section 2.1.1, subscript i is dropped.

The vector $\mathbf{v} = \mathbf{b}_i - \mathbf{a}_i$ is computed as

$$\mathbf{v} = \mathbf{p} + \mathbf{R}(\boldsymbol{\psi}) \, \mathbf{b}_{pi} - \mathbf{a}_i, \tag{2.10}$$

with $\mathbf{R}(\boldsymbol{\psi}) = \mathbf{R}_z(\gamma) \, \mathbf{R}_y(\beta) \, \mathbf{R}(\alpha)$.

Differentiating this expression with respect to the platform pose, the following equation is obtained

$$\delta\mathbf{v} = \underbrace{\begin{bmatrix} \mathbf{I}_3 & \frac{\partial}{\partial\boldsymbol{\psi}} \left(\mathbf{R}(\boldsymbol{\psi}) \, \mathbf{b}_{pi} \right) \end{bmatrix}}_{\frac{\partial\mathbf{v}}{\partial\mathbf{x}}} \delta\mathbf{x}, \tag{2.11}$$

where

$$\frac{\partial}{\partial\boldsymbol{\psi}} \left(\mathbf{R}(\boldsymbol{\psi}) \, \mathbf{b}_{pi} \right) =$$

$$= \begin{bmatrix} b_{piy}\,\sigma_1 + b_{piz}\,\sigma_2 & b_{piz}\,c_\alpha\,c_\beta\,c_\gamma - b_{pix}\,c_\gamma\,s_\beta + b_{piy}\,c_\beta\,c_\gamma\,s_\alpha & b_{piz}\,\sigma_3 - b_{piy}\,\sigma_4 - b_{pix}\,c_\beta\,s_\gamma \\ -b_{piy}\,\sigma_3 - b_{piz}\,\sigma_4 & b_{piz}\,c_\alpha\,c_\beta\,s_\gamma - b_{pix}\,s_\beta\,s_\gamma + b_{piy}\,c_\beta\,s_\alpha\,s_\gamma & b_{piz}\,\sigma_3 - b_{piy}\,\sigma_2 + b_{pix}\,c_\beta\,c_\gamma \\ b_{piy}\,c_\alpha\,c_\beta - b_{piz}\,c_\beta\,s_\alpha & -b_{pix}\,c_\beta - b_{piz}\,c_\alpha\,s_\beta - b_{piy}\,s_\alpha\,s_\beta & 0 \end{bmatrix},$$

with

$$\sigma_1 = (s_\alpha\,s_\gamma + c_\alpha\,c_\gamma\,s_\beta), \quad \sigma_2 = (c_\alpha\,s_\gamma - c_\gamma\,s_\alpha\,s_\beta), \tag{2.12}$$

$$\sigma_3 = (c_\gamma\,s_\alpha - c_\alpha\,s_\beta\,s_\gamma), \quad \sigma_4 = (c_\alpha\,c_\gamma + s_\alpha\,s_\beta\,s_\gamma), \tag{2.13}$$

and $c_\theta = \cos\theta$ and $s_\theta = \sin\theta$. Defining $\mathbf{v}^{0r} = \begin{bmatrix} v_1^0 & v_2^0 \end{bmatrix}^T$ and $\mathbf{E}_{0r} = \begin{bmatrix} \mathbf{e}_1^0 & \mathbf{e}_2^0 \end{bmatrix}$, one may write

$$\mathbf{v}^{0r} = \mathbf{E}_{0r}^T \, \mathbf{v} = l_0 \, \begin{bmatrix} \cos\rho \\ \sin\rho \end{bmatrix}, \tag{2.14}$$

with l_0 the length shown in Figure 2.2. Differentiating (2.14) and isolating $\delta\rho$, this equation leads to the derivative of ρ with respect to $\mathbf{x}$

$$\delta\rho = \underbrace{\left(\frac{1}{l_0} \, \begin{bmatrix} -\sin\rho & \cos\rho \end{bmatrix} \, \mathbf{E}_{0r}^T \, \frac{\partial\mathbf{v}}{\partial\mathbf{x}} \right)}_{\frac{\partial\rho}{\partial\mathbf{x}}} \delta\mathbf{x}. \tag{2.15}$$

Defining

$$\mathbf{R}_{zr}^T(\rho) = \begin{bmatrix} \cos\rho & \sin\rho & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

equation (2.1) may be rewritten as follows

$$\mathbf{v}^{1r} = \mathbf{R}_{zr}^T(\rho)\,\mathbf{E}_0^T\,\mathbf{v} = l_f \begin{bmatrix} \sin\phi \\ \cos\phi \end{bmatrix} + r_p \begin{bmatrix} 1-\cos\phi \\ \sin\phi \end{bmatrix}. \tag{2.16}$$

Differentiating this equation, the following relation is obtained

$$\begin{aligned} &\overbrace{\left(\mathbf{D}_{Rz}(\rho)\,\mathbf{E}_0^T\mathbf{v}\frac{\partial\rho}{\partial\mathbf{x}} + \mathbf{R}_{zr}^T(\rho)\,\mathbf{E}_0^T\,\frac{\partial\mathbf{v}}{\partial\mathbf{x}}\right)}^{\frac{\partial\mathbf{v}^{1r}}{\partial\mathbf{x}}}\delta\mathbf{x} = \\ &\qquad = \left(r_p\begin{bmatrix}\sin\phi\\ \cos\phi\end{bmatrix} + l_f\begin{bmatrix}\cos\phi\\ -\sin\phi\end{bmatrix}\right)\delta\phi + \begin{bmatrix}\sin\phi\\ \cos\phi\end{bmatrix}\,\delta l_f, \end{aligned} \tag{2.17}$$

with

$$\mathbf{D}_{Rz}(\rho) = \begin{bmatrix} -\sin\rho & \cos\rho & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{2.18}$$

The differential $\delta\phi$ can be isolated in (2.17) leading to the derivative of ϕ

$$\delta\phi = \underbrace{\left(\frac{1}{l_f}\begin{bmatrix}\cos\phi & -\sin\phi\end{bmatrix}\frac{\partial\mathbf{v}^{1r}}{\partial\mathbf{x}}\right)}_{\frac{\partial\phi}{\partial\mathbf{x}}}\delta\mathbf{x}. \tag{2.19}$$

Moreover, this expression can be substituted for $\delta\phi$ in (2.17) and δl_f is thereby obtained as

$$\delta l_f = \underbrace{\left(\begin{bmatrix}\sin\phi & \cos\phi\end{bmatrix}\frac{\partial\mathbf{v}^{1r}}{\partial\mathbf{x}} - r_p\,\frac{\partial\phi}{\partial\mathbf{x}}\right)}_{\frac{\partial l_f}{\partial\mathbf{x}}}\delta\mathbf{x}. \tag{2.20}$$

Differentiation of (2.7) leads to

$$\delta l_i = \underbrace{\left(r_p\frac{\partial\phi}{\partial\mathbf{x}} + \frac{\partial l_f}{\partial\mathbf{x}}\right)}_{\frac{\partial l_i}{\partial\mathbf{x}}}\delta\mathbf{x}. \tag{2.21}$$

Finally, the computations detailed above can be applied to each kinematic chain $i = 1, ..., m$, and the derivatives are concatenated to form the expression of the Jacobian matrix

$$\delta\boldsymbol{l} = \begin{bmatrix}\delta l_1\\ \vdots\\ \delta l_m\end{bmatrix} = \underbrace{\begin{bmatrix}\frac{\partial l_1}{\partial\mathbf{x}}\\ \vdots\\ \frac{\partial l_m}{\partial\mathbf{x}}\end{bmatrix}}_{\mathbf{J}}\delta\mathbf{x}. \tag{2.22}$$

2.1.3 Forward kinematic model

The FK model consists in computing the platform pose for a given set of winch motor positions. In the case of redundant CDPRs with $m > n$, n being the number of degrees of freedom of the mobile platform, the kinematic model is overdetermined. In general, due to modeling and measurement inaccuracies, it is impossible to find a platform pose that satisfies exactly the kinematic constraints set by the m kinematic chains. Therefore, the FK may be formulated as the problem of minimizing the error between the measured cable lengths and the cable lengths computed by the IK with (2.7).

Let $\boldsymbol{l}_m$ be the set of measured cable lengths and $\hat{\boldsymbol{l}}(\mathbf{x})$ the cable lengths obtained with the IK model for a given pose $\mathbf{x}$. The error to be minimized is defined as $e(\mathbf{x}) = \|\hat{\boldsymbol{l}}(\mathbf{x}) - \boldsymbol{l}_m\|_2$, where $\|(\cdot)\|_2$ denotes the 2-norm of $(\cdot)$. Therefore, for a given $\boldsymbol{l}_m$, the FK algorithm should return the solution $\mathbf{x}^* = \arg\min_{\mathbf{x}} \|\hat{\boldsymbol{l}}(\mathbf{x}) - \boldsymbol{l}_m\|_2$.

In general, the function $e(\mathbf{x}) = \|\hat{\boldsymbol{l}}(\mathbf{x}) - \boldsymbol{l}_m\|_2$ possesses several local minima. The proposed FK algorithm takes an initial guess $\mathbf{x}_g$ and find a $\mathbf{x}^*$ that locally minimizes the function $e(\mathbf{x})$. The proposed algorithm assumes that $\mathbf{x}_g$ is sufficiently close to the current platform pose and consists of an iterative scheme. For an iteration k with platform pose $\mathbf{x}_k$, the next iteration takes $\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta\mathbf{x}_k$. Considering the approximation $\Delta\hat{\boldsymbol{l}} \approx \mathbf{J}(\mathbf{x}_k)\,\Delta\mathbf{x}$, a reasonable choice for $\Delta\mathbf{x}_k$ is

$$\Delta\mathbf{x}_k = \arg\min_{\Delta\mathbf{x}} \left\| \mathbf{J}(\mathbf{x}_k)\,\Delta\mathbf{x} - \left(\boldsymbol{l}_m - \hat{\boldsymbol{l}}(\mathbf{x}_k)\right) \right\|_2 . \tag{2.23}$$

The minimization problem (2.23) is a Linear Least Squares problem. The solution of (2.23) might be computed solving

$$\mathbf{J}(\mathbf{x}_k)^T \mathbf{J}(\mathbf{x}_k)\,\Delta\mathbf{x} = \mathbf{J}(\mathbf{x}_k)^T \left(\boldsymbol{l}_m - \hat{\boldsymbol{l}}(\mathbf{x}_k)\right), \tag{2.24}$$

for $\Delta\mathbf{x}$. However, SVD and QR factorizations may solve this linear least-square problem more efficiently [88, 176]. The SVD may be preferred because of its greater diagnostic capability in pathological cases. Nevertheless, QR factorization presents faster computing time. The latter being critical in real-time applications, the QR factorization is preferred. Therefore, the Jacobian matrix $\mathbf{J}$ is decomposed using the QR factorization such that

$$\mathbf{J} = \mathbf{Q}\,\mathbf{R},$$

with $\mathbf{Q}$ orthogonal and $\mathbf{R}$ upper triangular. The solution of (2.23) is then obtained from the back substitution of the system

$$\mathbf{R}\,\Delta\mathbf{x} = \mathbf{Q}^T \left(\hat{\boldsymbol{l}}(\mathbf{x}_k) - \boldsymbol{l}_m\right).$$

Once $\Delta\mathbf{x}_k$ is obtained, the next pose $\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta\mathbf{x}_k$ is computed. This procedure is repeated until $\|\Delta\mathbf{x}\|_2 < \epsilon$, for ϵ the desired tolerance.

Note that the proposed algorithm does not use numerical damping. This was not necessary for the studied scenarios. Numerical damping is applied in commonly used

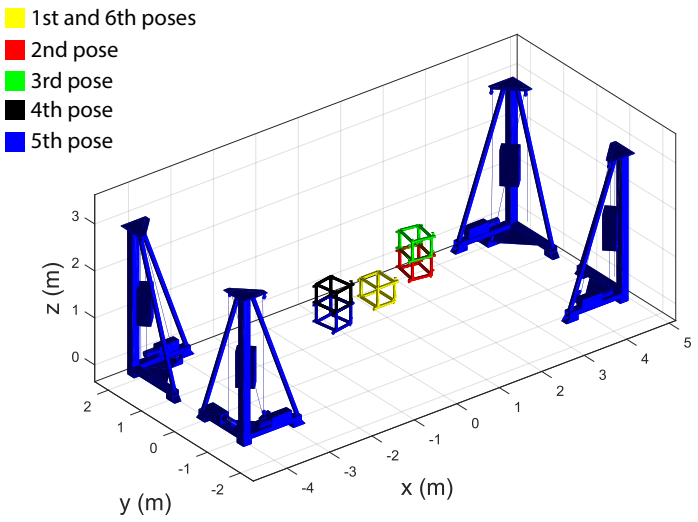

Figure 2.3 – *CAD view of the HRPCable prototype presenting the performed trajectory.*

Levenberg-Marquadt methods in order to compensate the incidence of ill-conditioned configurations. If a given iteration k reaches a pose in which $\mathbf{J}(\mathbf{x}_k)^T\,\mathbf{J}(\mathbf{x}_k)$ is close to singular, one may solve $\left(\mathbf{J}(\mathbf{x}_k)^T\mathbf{J}(\mathbf{x}_k) + \lambda\,\mathbf{I}\right)\,\Delta\mathbf{x} = \mathbf{J}(\mathbf{x}_k)^T\left(\boldsymbol{l}_m - \hat{\boldsymbol{l}}(\mathbf{x}_k)\right)$ instead of (2.24), for a positive $\lambda \in \mathbb{R}$. This procedure improves numerical stability.

2.1.4 Experimental results

The proposed FK algorithm has been implemented in the HRPCable prototype shown in Figure 1.1. As introduced in Section 1.2, this prototype has a 6-DoF platform fully constrained by 8 cables. The control control software has been developed in C++ and runs in an industrial PC Beckhoff C6920 equipped with 2.4GHz i7 core processor. The platform pose control loop runs at 125 Hz. An inner feedback loop running at 2 kHz control the cable tensions according to the control scheme proposed in Section 2.3. Cable tensions were measured by means of force sensors (load pins) placed in the drawing pulleys. Figure 2.3 depicts the performed trajectory. The threshold ϵ was defined as $\epsilon = 1 \times 10^{-6}$.

In addition to the components shown in Figure 1.1, a Metris K600 camera system is used to measure the pose (6-DoF) of the platform in real time with a precision of 70 μm. The measurements obtained with the K600 camera system were used as reference to compare with the estimations obtained with the proposed FK algorithm. These results are shown in Figure 2.4. This figure also shows the number of iterations necessary to the FK algorithm to converge for each sampling time during the trajectory.

Figure 2.4 indicates the presence of significant errors obtained with the proposed FK algorithm. Such errors are probably a consequence of the assumption that the cables are inelastic. Accordingly, an identification of the elastic behavior of the cables combined with the cable tension measurements may lead to an improved kinematic model. This is, indeed, a subject of possible future works.

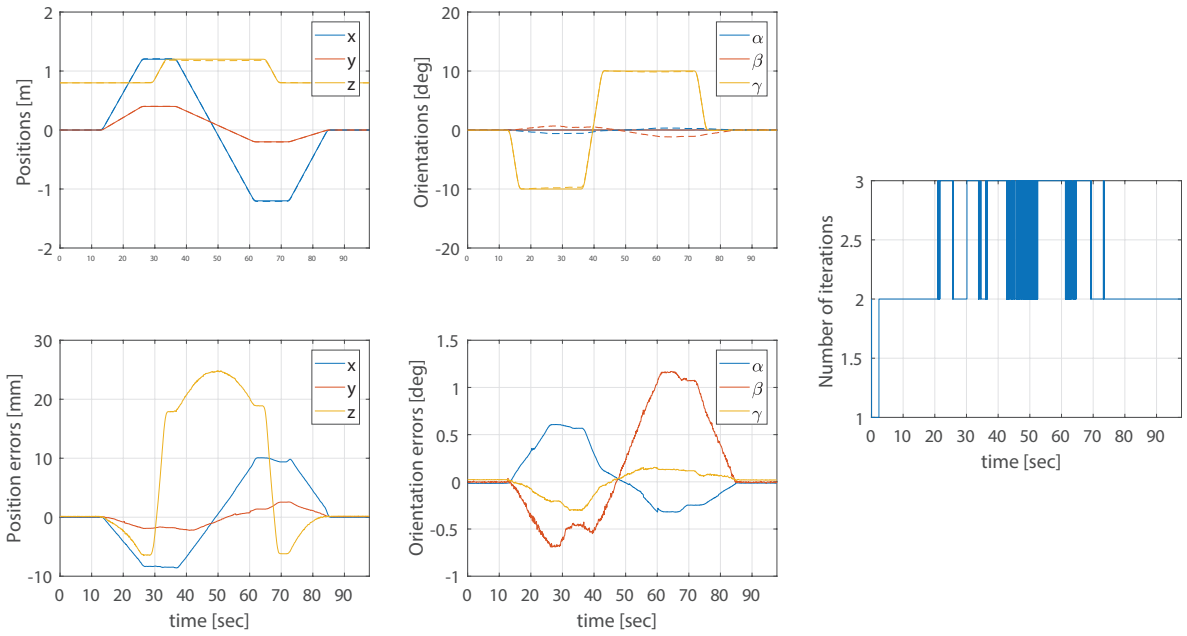

Figure 2.4 – *Experimental results: Solid lines represent the poses estimated with the proposed FK algorithm and dashed lines represent the poses measured by the Metris K600 camera system. On the right, the number of iterations necessary for the FK algorithm to converge for each sampling time during the trajectory is presented.*

For a given pair of pose and length vectors $\mathbf{x}^*$ and $\boldsymbol{l}^*$ consistent with the constraints presented in section 2.1.1, rigorously evaluating the capability (denoted here as convergence capability) of an FK iterative algorithm to quickly and reliably find $\mathbf{x}^*$ for given $\boldsymbol{l}^*$ is not a trivial task and is out of the scope of this thesis. The FK problem of a Stewart-Gough platform (topologically very similar to CDPRs) may have up to 40 solutions [177] and redundant configurations do not necessarily present a reduced number of possible solutions compared to the non-redundant ones [178]. Therefore, it is necessary to prove that the algorithm is able to converge to $\mathbf{x}^*$, and not just to one of the (potentially) many $\mathbf{x}$ that are consistent with $\boldsymbol{l}^*$. Nevertheless, it is still interesting to address this capability as follows.

The FK algorithm has been applied to IK solutions corresponding to more than 30×10^3 poses equally spaced across the workspace keeping the initial guess constant equal to $\mathbf{x}_g = \begin{bmatrix} 0 & 0 & .8 & 0 & 0 & 0 \end{bmatrix}^T$ for all the poses. The workspace of the prototype is a $4 \times 2 \times 1.4$ m^3 (x, y, z) cuboid, considering $-10° \leqslant \gamma \leqslant 10°$, $\alpha = \beta = 0$. The algorithm obtained a pose with errors smaller than the proposed tolerance within 7 iterations for all these poses. Clearly, this test was not performed in real-time. The cable length vectors $\boldsymbol{l}$ corresponding to the IK solutions were artificially fed to the FK algorithm.

2.2 A Torque-Based Cable Tension Control

The vast majority of CDPR position tracking control strategies for CDPRs define, at some point, a set of desired cable tensions. The present and next sections are dedicated to the control strategies responsible for generating these cable tensions by controlling variables that can be directly managed in the motor drives, namely, motor currents (or torques) and velocities.

As discussed in Section 1.6, the cable tension control in CDPRs is often performed by means of an open-loop scheme based on a model relating the motor torques to the cable tensions. Nevertheless, the precision obtained with such models may be deteriorated for winches presenting significant friction. Clearly, the implementation of a closed-loop strategy measuring the actual cable tensions may reduce this error.

Moreover, fully-constrained CDPRs typically have cable tension sensors for safety reasons. Hence, it is appropriate to consider the inexpensive information of measured cable tensions within these strategies. Therefore, Sections 2.2 and 2.3 focus on closed-loop strategies.

Based on the intuitive relation between cable tensions and motor torques, a torque-based scheme is firstly proposed. Several previous works presented satisfying results with torque-based cable tension control schemes operating in open-loop [8, 47]. Nevertheless, the conducted experimental tests on the HRPCable prototype indicate that a torque-based closed-loop scheme would have its implementation hindered or substantially complicated by the presence of significant friction in the gear trains. This section aims to provide theoretical and experimental data allowing to analyze the applicability of a torque-based scheme.

Before addressing the control schemes themselves, the experimental set-up used to evaluate the proposed schemes is introduced.

2.2.1 Problem formulation and experimental setup

Sections 2.2 and 2.3 address cable tension control with different approaches. Accordingly, the results presented in both sections are based on the same experimental set-up. This set-up is illustrated in Figure 2.5. Two kinematic chains of the HRPCable prototype (*cf.* Figure 1.1) are used. The mobile platform is not used. The corresponding cable ends are attached together.

The resulting system may be seen as a 1-DoF mechanism driven by two cables. The position of the attachment point is given by the variable x. The torques applied by the motors in winches 1 and 2 are denoted by γ_1 and γ_2, respectively. Similarly, motor positions are denoted by q_1 and q_2.

Consider given desired positions $x_d(t)$ and desired cable tensions $\tau_d(t)$ in function of time. The present section proposes a control scheme meant to generate measured cable tensions τ_1 as close as possible to the desired ones controlling motor torques

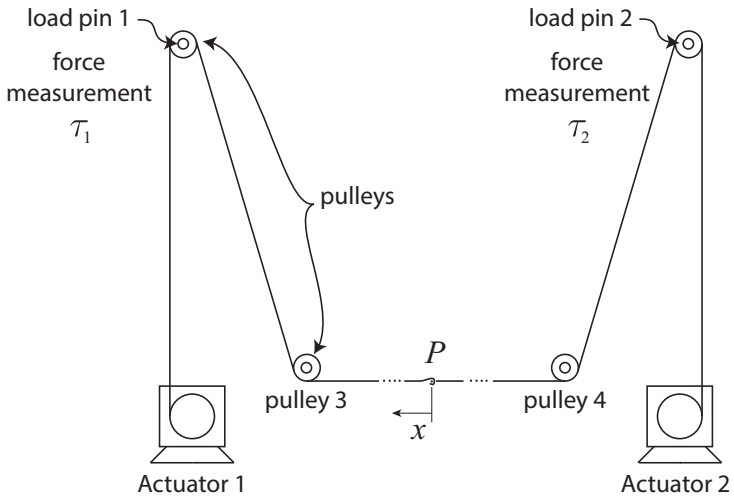

Figure 2.5 – *An illustrative schematic view of the experimental set-up.*

γ_1. The motor drive of winch 2 is controlled in position mode. The position $q_2(t)$ is controlled considering the desired position $x_d(t)$ and inelastic cables. More precisely, the motor position q_2 is controlled as

$$q_2(t) = q_2(t_0) + k_{qx}\,(x_d(t) - x_d(t_0)), \tag{2.25}$$

with t_0 the initial time and a constant k_{qx} linearly relating cable length variations to the angular displacements of the motors.

Note that, whereas the final goal is to generate the desired cable tension applied on the attachment point P, the considered goal is to track $\tau_d(t)$ based on $\tau_1(t)$. Therefore, the influence of distributed mass of the cable and the friction on the pulleys are neglected. These are the main causes of divergence between measured tension in the load-pin 1 and the force applied in point P. Even though, this subject is addressed in Section 2.3.4.

As shown in Table 1.1, the diameter of the cables is 4 mm and the diameter of the pulleys is 80 mm. The winches are composed of Beckhoff AM8061 motors coupled to AG2210 gear trains. These gear trains have a reduction ratio equal to 25. Load pins Sensy 5300-1T with integrated amplifier are positioned on the pulley axes. It is worth noting that a sufficiently small influence of noise could be obtained with this configuration of force sensing. Therefore, filtering was not necessary.

2.2.2 Dynamic Model

As stated in the previous section, the cable distributed mass is neglected so that the cable tension is considered constant all over the cable length. Under this assumption, the model of the system depicted in Figure 2.5 is reduced to the model of the winches.

A detailed model of the actuator may take into account each body of the actuator. Take as example the illustrative gear train depicted in Figure 2.6. This gear train consists of three bodies. These bodies are identified by letters a, b and c. Body b presents two gears, identified by numbers 1 and 2. Each body presents a corresponding inertia

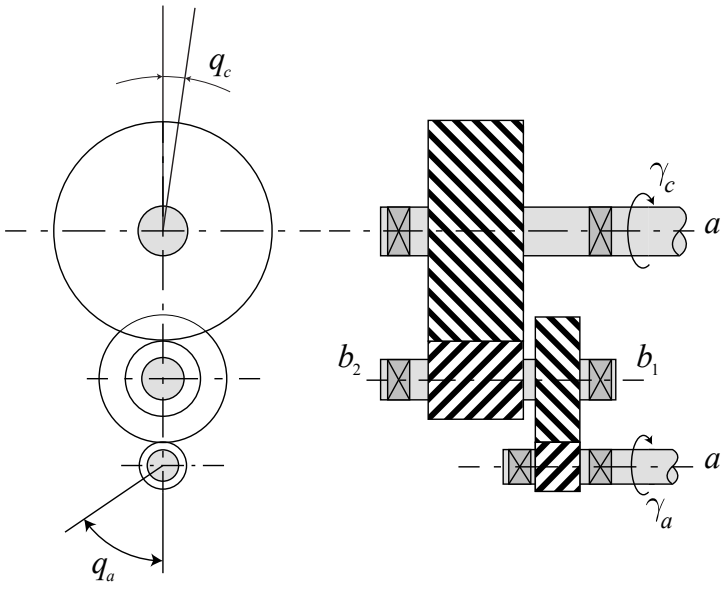

Figure 2.6 – *Drawing of a simplified gear train.*

(I_a, I_b and I_c). The angular positions are denoted by q_a, q_b and q_c. The external torques γ_a and γ_c are applied on axes a and c, respectively. The number of teeth of the gears are N_a, N_{b1}, N_{b2} and N_c. By neglecting backlash, the rigid-body dynamic model of the system can be written as

$$I_c\,\ddot{q}_c = \left((\gamma_a - I_a\,\ddot{q}_a)\frac{N_{b1}}{N_a} - I_b\,\ddot{q}_b\right)\frac{N_c}{N_{b2}} - \gamma_c - \gamma_F(\dot{q}_a, \dot{q}_b, \dot{q}_c, \gamma_a, \gamma_c). \tag{2.26}$$

The torque γ_F is a function of $(\dot{q}_a, \dot{q}_b, \dot{q}_c, \gamma_a, \gamma_c)$ that suitably models the friction of the system. Denoting $c_1 = \frac{N_c}{N_{b2}}\frac{N_{b1}}{N_a}$ and $c_2 = \frac{N_c}{N_{b2}}$, all velocities and accelerations may be written in function of the derivatives of q_c

$$\begin{cases} \dot{q}_a = c_1\,\dot{q}_c \\ \ddot{q}_a = c_1\,\ddot{q}_c \\ \dot{q}_b = c_2\,\dot{q}_c \\ \ddot{q}_b = c_2\,\ddot{q}_c \end{cases}. \tag{2.27}$$

Therefore, a new function may be written as $\gamma_f(\dot{q}_c, c_1\,\tau_1, \tau_3) = \gamma_F(c_1\dot{q}_c,\ c_2\,\dot{q}_c,\ \dot{q}_c, \tau_1, \tau_3)$. Furthermore, the model (2.26) may be simplified as

$$(I_a\,c_1^2 + I_b\,c_2^2 + I_c)\,\ddot{q}_c = -\gamma_c + c_1\,\gamma_a - \gamma_f(\dot{q}_c, c_1\,\gamma_a, \gamma_c). \tag{2.28}$$

Denoting $I = (I_a\,c_1^2 + I_b\,c_2^2 + I_c)$, $\gamma = c_1\,\gamma_a$, $\gamma_{out} = \gamma_c$ and $q = q_c$, an equivalent single body system is obtained with

$$I\,\ddot{q} = \gamma - \gamma_{out} - \gamma_f(\dot{q}, \gamma_{in}, \gamma_{out}). \tag{2.29}$$

Even if model (2.29) seems rather simple, a precise model usually involves an elaborate function γ_f [179,180]. Highly nonlinear phenomena such as stick-slip may present non-negligible influence in the studied system.

For the system illustrated in Figure 2.5, γ is the motor torque multiplied by the gear ratio. The output torque γ_{out} is the torque applied by the cables γ_t. Note that $\gamma_t = \tau\, r_w$, where τ is the cable tension and r_w is the radius of the winch drum.

Let $\mathbf{y}$ be the state vector consisting of the angular position q and velocity $\dot{q}$ of the winch drum so that

$$\mathbf{y}(t) = \begin{pmatrix} y_1(t) \\ y_2(t) \end{pmatrix} = \begin{pmatrix} q(t) \\ \dot{q}(t) \end{pmatrix}. \tag{2.30}$$

In order to take into account the stick-slip phenomenon, the following state-space model is proposed

$$\dot{\mathbf{y}}(t) = \begin{cases} \begin{pmatrix} 0 \\ 0 \end{pmatrix} & \text{, if } |\gamma| \leqslant \gamma_{sd} \text{ and } y_2 = 0; \\ \begin{pmatrix} y_2 \\ \frac{1}{I}\left(\gamma - \gamma_{kf}(y_2) - \tau\, r_w\right) \end{pmatrix} & \text{, othewise} \end{cases} \tag{2.31}$$

where γ_{sd} is the static dry friction torque. The scalar γ_{kf} is the kinetic friction torque, which is function of the angular velocity $y_2 = \dot{q}$. Different definitions of the function $\gamma_{kf}(y_2)$ were proposed in the literature. It is worth noting that the model typically considered in the control of CDPRs (such as in [7, 8, 58, 77]) is described by (2.31) with $\gamma_{sd} = 0$. The model considered in [77] is often used in this field and is given by

$$\gamma_{kf_1}(y_2) = f_{kd}\, \tanh(\mu\, y_2) + f_{v1}\, y_2, \tag{2.32}$$

where f_{kd} is the kinetic dry friction coefficient and f_{v1} is a viscous friction coefficient. The function $\tanh(\mu\, y_2)$ is a continuous approximation of the function $\mathrm{sign}(y_2)$ and the constant μ is used to tune this function. However, Section 2.2.3 shows that a better consistency with the experimental data was obtained with the following definition of the kinetic friction torque

$$\gamma_{kf}(y_2) = f_{kd}\, \tanh(\mu\, y_2) + f_{v1}\, y_2 + f_{v2}\, \mathrm{sign}(y_2)\, \sqrt{|y_2|}, \tag{2.33}$$

with a second viscous friction coefficient f_{v2}. The model presented in (2.33) is used in the remainder of this section. The obtained results presented in Section 2.2.3 show that this model is able to fit experimental and numerical data thanks to the addition of the term $f_{v2}\, \mathrm{sign}(y_2)\, \sqrt{|y_2|}$.

Next, before presenting the proposed control scheme and the corresponding results, the parameters of (2.31) and (2.33) need to be identified. The next section addresses this subject.

2.2.3 Parameters Identification

Since the dynamic parameters (inertia and friction constants) of the actuator are independent of the cable tensions, all experimental data described in this section are obtained with the winch running freely (no cable is attached to the actuator).

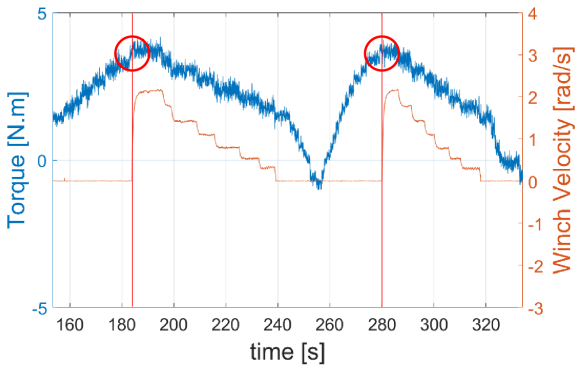

Figure 2.7 – *Identification of the static dry friction torque.*

The first identified parameter was the static dry friction torque γ_{sd}. Consider as initial state $\gamma = \dot{q} = 0$. Increasing gradually γ, velocity $\dot{q}$ will still equal to zero until $\gamma = \gamma_{sd}$. The value of γ_{sd} may be estimated using this procedure. The presented torques are given by the motor driver based on the measured motor current. The experiment depicted in Figure 2.7 was repeated several times in order to obtain a good estimation of γ_{sd}. According to model (2.31), the torque values highlighted in Figure 2.7 are equal to the static dry friction torque. The conducted experiments show that the behavior of the actuator regarding the friction is significantly different considering the direction of the torque. The analysis of the friction torque has thus been done separately for each direction ($\dot{q}$ positive and negative). The obtained results are summarized in Table 2.1.

	$\dot{q} > 0$	$\dot{q} < 0$
Mean Value	3.575	3.207
Standard Deviation	0.149	0.257

Table 2.1 – *Estimation of the static dry friction torque (measured values in Nm).*

The remaining friction coefficients have been estimated repeating the experiment depicted in Figure 2.8. Piecewise constant motor torque setpoints are sent to the drives. A constant value is set until the transient behavior ends. For each value of desired motor torque, the related angular velocity has been recorded after the transient interval. Thereby, the relationship between the recorded velocities and torques are independent of the inertia. This procedure have been repeated several times, leading to the chart depicted in Figure 2.9. The coefficient μ was set empirically equal to 10^3. A gradient based optimization (interior-point method) yielded to the coefficients of model (2.31) minimizing the difference between the predicted torque and the experimental one. The obtained results are also shown in Figure 2.9. Similarly to the estimation of γ_{sd}, different values of friction coefficients were obtained for $\dot{q} > 0$ and $\dot{q} < 0$. This figure also shows the relevance of the nonlinear term $f_{v2}\ \mathrm{sign}(y_2)\ \sqrt{|y_2|}$ in (2.33).

Finally, the inertia of the actuator has been identified by applying sinusoidal torques with different frequencies and amplitudes. The value of $I = 507.5$ kg.cm^2 minimizes the difference between the measured profile of $\dot{q}$ and the one given by the model, taking measured torques as inputs.

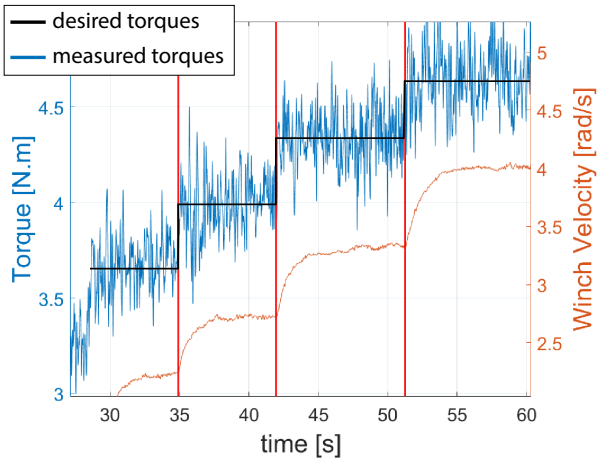

Figure 2.8 – *Estimation of the kinetic friction coefficients.*

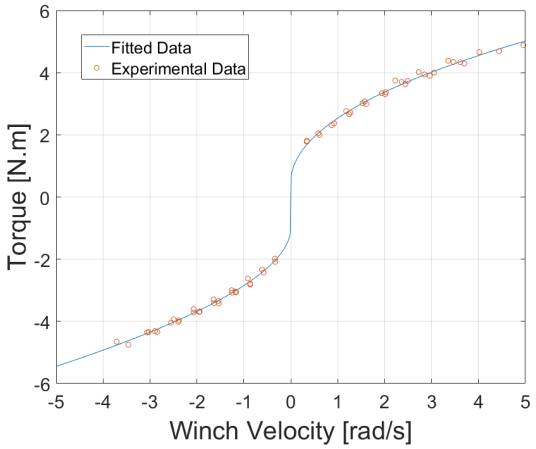

Figure 2.9 – *Estimation of the kinetic friction coefficients.*

2.2.4 Proposed control scheme

The torque applied on the winch is closely related to the cable tensions. For this reason, a unique control loop where the motor torque is the control input may seem reasonable. Anyhow, the lowest level control input applied on the motor is the torque/current. Accordingly, the present section proposes a control scheme that is based on the studies presented in [47,77].

The proposed scheme is depicted in the block diagram of Figure 2.10 and it may be expressed as

$$\gamma = k_P\,(\tau_d - \tau) + k_I \int_{t_0}^{t_a} (\tau_d - \tau)\,dt - k_D\,\dot{q} + \gamma_{kf}(\dot{q}) + \ddot{x}\,\frac{I}{r_w}, \tag{2.34}$$

with γ_{kf} given by (2.33) and gains k_P, k_I, k_D. Instants t_0 and t_a denote the initial and actual time, respectively.

The friction and inertia effects are compensated by the terms $\gamma_{kf}(\dot{q})$ and $\ddot{x}\,I/r_w$, respectively. The proportional and integral terms reduce the error between measured tension τ and desired tension τ_d. Stability is enhanced by the term $-k_D\,\dot{q}$. Note that

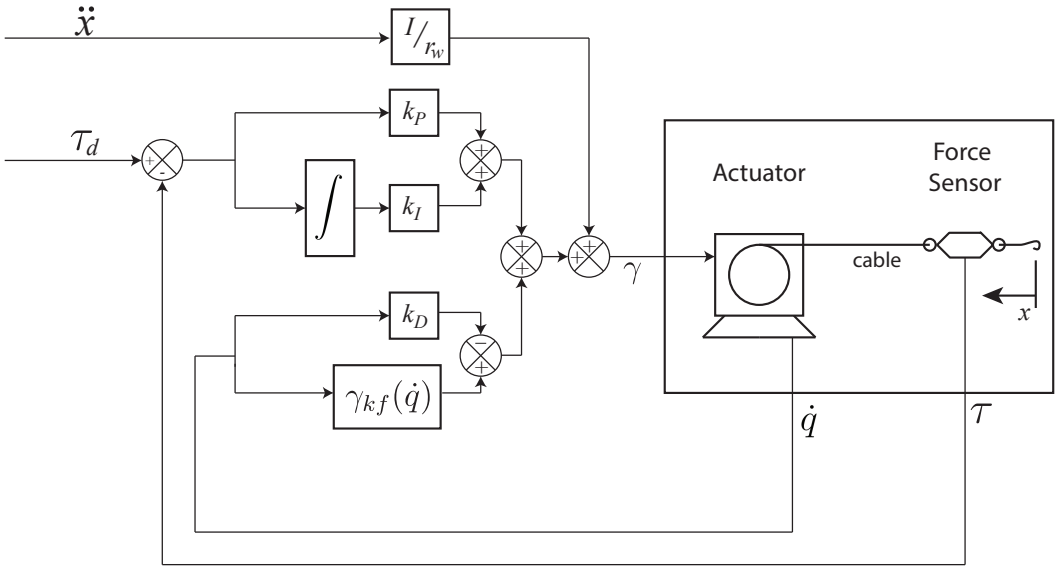

Figure 2.10 – *Block diagram of the proposed torque control scheme.*

the addition of a term proportional to the derivative of cable tensions $\dot{\tau}$ is not recommended due to the presence of noise.

2.2.5 Numerical simulation results

This section presents simulations considering the control scheme (2.34), the actuator model (2.31) and the parameters identified in Section 2.2.3. In addition to the actuator model, the cable elasticity needs to be modeled in order to obtain numerical results consistent with the experimental ones.

In contrast to the experimental set-up, the position $x(t)$ may be simulated without the presence of pulleys. Accordingly, the simulated system may be simplified as presented in Figure 2.11. It consists of an actuator (motor, gears and drum) connected to a cable. The other end of the cable is attached to a moving point P. This point moves in one direction and its displacement is given by x. A force sensor measures the cable tensions. The real stretched length of the portion of the cable that is not in contact with the winch drum is l_1. The unstretched length of this portion of the cable is equal to l_2. The angular position of the drum is q. The model (2.31) is applied including the linear elastic cable model

$$t = \max\left(\frac{EA\,(l_1 - l_2)}{l_2},\ 0\right), \tag{2.35}$$

where the term EA is the specific elastic constant of the cable. It is equal to the Young Modulus of the material multiplied by the area of the cross section of the cable. This model is based on the one proposed in [44]. Consider $E = 110$ GPa [181] the Young Modulus of stainless steel, and the fill rate equal to 0.7 (70% of the area defined by the nominal diameter of the cable is actually filled with steel). The resulting value of EA is equal to 9.68×10^5 N.

Simulations take as initial conditions $l_1 = l_2 = l_0$ and $x = q = 0$. This way, $l_2 =$

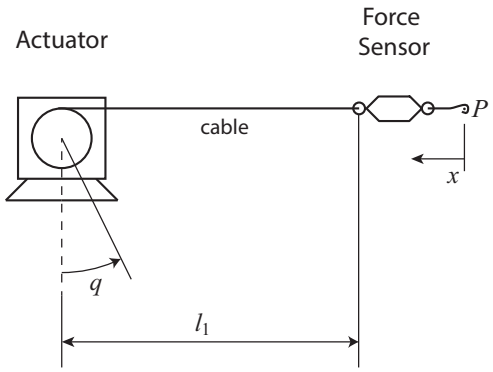

Figure 2.11 – *Illustration of the simulated simplified system.*

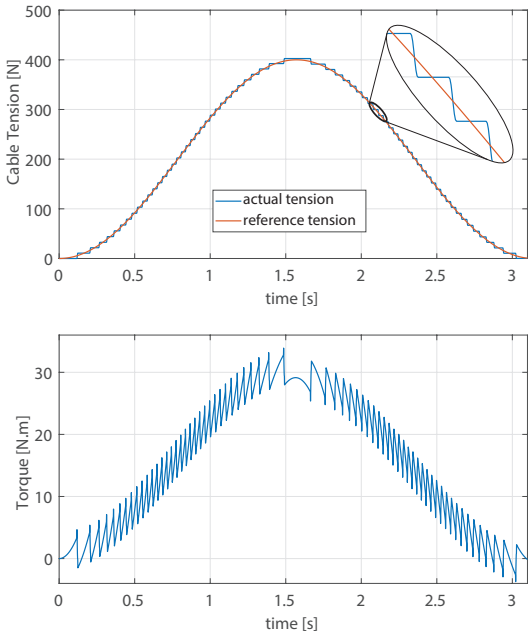

Figure 2.12 – *Simulation results with the torque control: evolution of the cable tension and the torque versus time.*

$q\, r_w + l_0$ and $l_1 = l_0 - x$. Moreover, simulations considers a cycle time of 1 ms.

Figure 2.12 presents the obtained results. The controller is not able to continuously track the desired cable tensions due to the so-called stick-slip phenomenon. This phenomenon results in a discontinuous relation between the input torque and the cable tension. This discontinuity cannot be compensated with a torque-based continuous control policy. More precisely, although γ_{kf} may be compensated in (2.34), the discontinuous system (2.31) introduces the stick-slip torque represented by γ_{sd} that is not considered in the control law. Clearly, a discontinuous torque-based control scheme compensating the stick-slip phenomenon in (2.31) is not viable.

The aforementioned issue is present for velocities $\dot{q}$ close to zero. This is the condition in which the system is susceptible to the discontinuity represented in (2.31). For sufficiently large velocities, the proposed scheme (2.34) may lead to satisfying results.

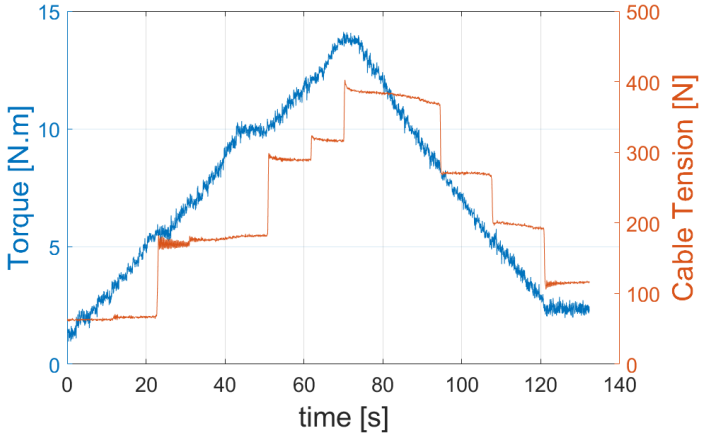

Figure 2.13 – *Simultaneous plot of the applied torque and measured cable tension (with the cable attached to a fixed support).*

2.2.6 Issues related to a torque-based control approach

Section 2.2.5 demonstrated that the application of the proposed torque-based control approach may lead to problems resulting from the presence of friction in the winches. Indeed, preliminary experimental results confirmed this issue. More specifically, the main concern related to the control of the cable tensions is the presence of static dry friction in the actuators. In some cases, this may be a minor problem. For instance, for the CDPR CoGiRo (Figure 1.2 (b)), motor torques of each winch are transmitted to the corresponding drum by a system of pulleys and belt with a reduction ratio equals to 3. Whereas each motor of HRPCable is connected to a 2 stage gear train Beckhoff AM8061 with a reduction ratio equals to 25. This leads to increased friction torques. Figure 2.13 illustrates this problem. In this experiment, one end of the cable is attached to a fixed point and the values of the measured cable tensions are obtained for different motor torques. One can see that a continuous torque variation results in abrupt changes of cable tensions.

As a matter of fact, every experiment performed with variations of the torque-based closed-loop scheme (2.34) led to technical issues. The resulting closed-loop system is not able to track cable tensions when the velocity $\dot{q}$ is small.

In addition, the estimations presented in Section 2.2.3 are time-consuming and lead to parameters identified with a bad level of precision. Considering that such estimations should be repeated for each one of the winches, a bad compromise regarding the complexity of implementation and the level of precision is obtained.

Within this context, a non-model based scheme able to minimize the influence of friction may be an appropriate alternative to the proposed torque-based scheme. This forms the rationale of the method presented in the next section.

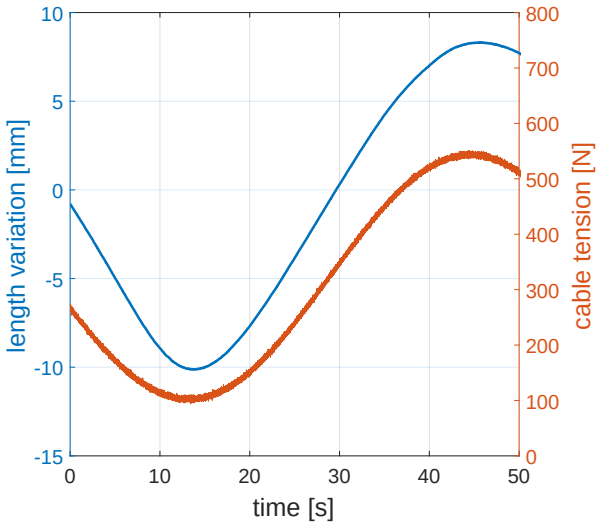

Figure 2.14 – *Simultaneous plot of variations in the unstretched length and the measured cable tension (with the cable attached to a fixed support).*

2.3 A Velocity-Based Cable Tension Control

Section 2.2.6 discussed the main limitations of torque-based cable tension control schemes when gear trains with large reduction ratio are used. Since this is the case of both HRPCable and Hephaestus prototypes, an alternative approach is proposed in this section.

As the motor torques, one may note that there is an intuitive relation between cable tensions and motor positions. For a given x, variations on q lead to variations on τ according to the cable elasticity. Figure 2.14 illustrates this idea. The presented test is very similar to the one used to obtain the results of Figure 2.13: one cable end is attached to a fixed point and the cable tension is measured while the winch is controlled. The difference between the tests performed in Figure 2.13 and Figure 2.14 is that the motor is controlled in position mode instead of torque mode.

The motor position is changed continuously while the cable tensions are measured. As a matter of fact, the results presented in Figure 2.14 are rather obvious considering the elastic behavior of the cable. Nevertheless, the comparison between Figures 2.13 and 2.14 demonstrates that a more promising scenario is obtained with the later approach.

In general, a trivial position control is able to perform a sufficiently precise position tracking of q independently of the external torques applied on the winches. This means that satisfying positioning precision of q may be obtained independently of the cable tension τ. That said, the motor position q may be controlled in order to generate the desired cable tension τ_d. Noting that the relation between the cable tension τ and the position q is continuous and well behaved, one may conclude that better results than the ones presented in Section 2.2 may be obtained controlling the motor position (instead of torque).

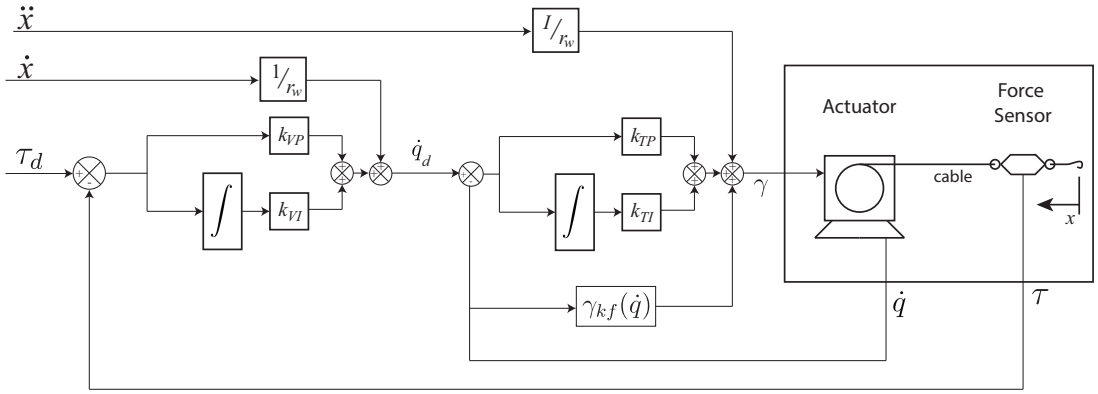

Figure 2.15 – *Block diagram of the proposed velocity-based cable tension control scheme.*

This rationale would lead to a cascade control scheme in which the outer loop takes as control input the motor position and the inner loop controls the motor torque. Thereby, the influence of the friction within the gear trains would be managed by the position (inner) control loop, which presents good robustness capabilities against model uncertainties.

This is indeed the approach proposed in [59, 60]. Nevertheless, some secondary complications come up with this method. Namely, a precise identification of the system dynamic may be necessary and the suppression of steady-state errors demands the application of an anti-windup feedback strategy (refer to [59] for further details). For this reason, the present section proposes a scheme in which the motor velocity is taken as control input. This method allies the aforementioned advantage without leading to secondary issues.

2.3.1 Control scheme

Figure 2.15 depicts the proposed velocity-based cable tension control. An outer feedback loop takes the cable tension τ as measured state and delivers the desired winch velocity $\dot{q}_d$ as intermediary control input. This outer loop applies a linear PI correction and a feedforward term to compensate the velocity corresponding to $\dot{x}$. An inner loop defines the motor torque taking the desired velocity as reference. Similarly, it applies a linear PI term and applies a feedforward term compensating the inertia and friction. In the considered experimental setup, the servo drives are responsible for this inner loop. Depending on the features available on the motor drive, an alternative feedforward term may be used. The control setpoint delivered to the motor drive is the desired motor velocity. The servo drives may present lower cycle time (between 62.5 μs and 250 μs for the servo drives used in this chapter) than the PLC cycle time. The reduced sampling time used in the torque control loop may lead to better performances.

For a given actual instant t_a, the mathematical expressions for $\dot{q}_d$ and γ in Figure

2.15 are

$$\dot{q}_d = k_{VP}\left(\tau_d - \tau\right) + k_{VI}\int_{t_0}^{t_a}\left(\tau_d - \tau\right)dt + \frac{\dot{x}}{r_w}, \tag{2.36a}$$

$$\gamma = k_{TP}\left(\dot{q}_d - \dot{q}\right) + k_{TI}\int_{t_0}^{t_a}\left(\dot{q}_d - \dot{q}\right)dt + \gamma_{kf}(\dot{q}) + \ddot{x}\frac{I}{r_w}. \tag{2.36b}$$

The values of $\dot{x}$ and $\ddot{x}$ are inputs of the block diagram in Figure 2.15. These are the time derivatives of x, which represents the desired cable length. Therefore, $\dot{x}$ and $\ddot{x}$ can be determined using the inverse kinematics proposed in Section 2.1.

Since the inner loop is performed by the servo drives in the experimental set-up, the control (2.36b) is used uniquely for the simulations. Note that the control implemented in the experimental set-up is the outer loop (2.36a) which consists of a PI feedback control with feedforward $\dot{x}/r_w$. Therefore, the implementation and tuning are rather simple.

2.3.2 Numerical simulation results

As discussed at the beginning of this section, the influence of friction should be reduced by controlling the motor velocity instead of the motor torques directly. Accordingly, in contrast to the data presented in Section 2.2.5, the numerical simulations using the proposed velocity-based control yielded to satisfying results. As depicted in Figure 2.16, this strategy is able to keep a small error. Maximum error for this simulation is equal to 0.539 N with desired cable tensions varying with an amplitude of 400 N.

The spikes in the cable tension error and torque are consequences of the stick-and-slip phenomenon. The discontinuity in (2.31) is faced when the time derivative of the desired cable tensions is close to zero. Nevertheless, in contrast to the scheme proposed in Section 2.2, the position control is sufficiently robust to reduce the influence of the friction.

2.3.3 Real-time experimental results

In accordance with the description given in Section 2.2.1, two winches of the CDPR HRPCable were used as shown in Figure 2.5. Motor 1 is controlled with the velocity control described in section 2.3.1 and cable tension obtained with load pin 1 is taken as feedback signal. Motor 2 is controlled in position mode according to (2.25). This scenario represents the conditions in which the cable tensions should be controlled while the attachment point position is variable. This application is equivalent to the cable tension control necessary to actuate a CDPR. As for the simulations presented in Section 2.3.2, the controller gains used in the present section were tuned by trial and error.

Figure 2.17 shows the results obtained with the velocity-based control with a step input. The lower level is 190 N and the upper is 440 N. The rise time to reach 440

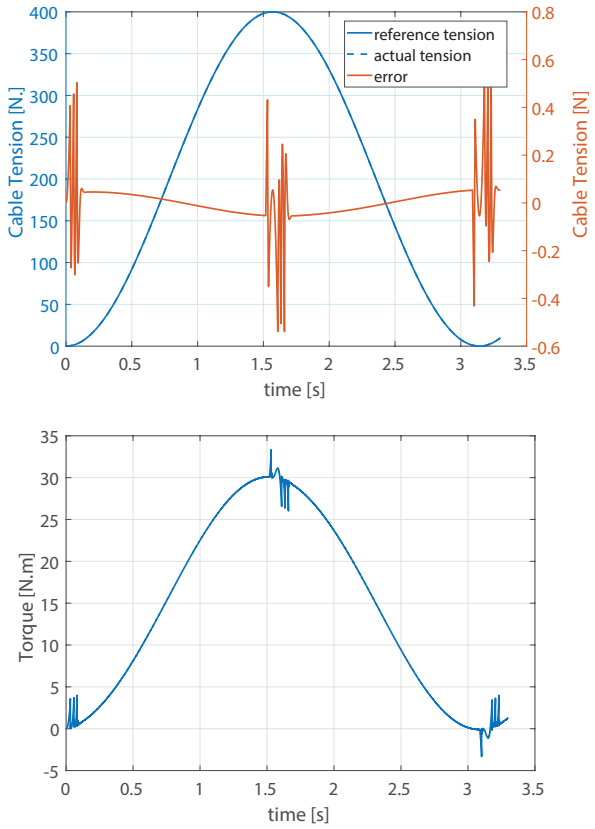

Figure 2.16 – *Simulation results with the velocity-based control: evolution of the cable tension and the torque versus time.*

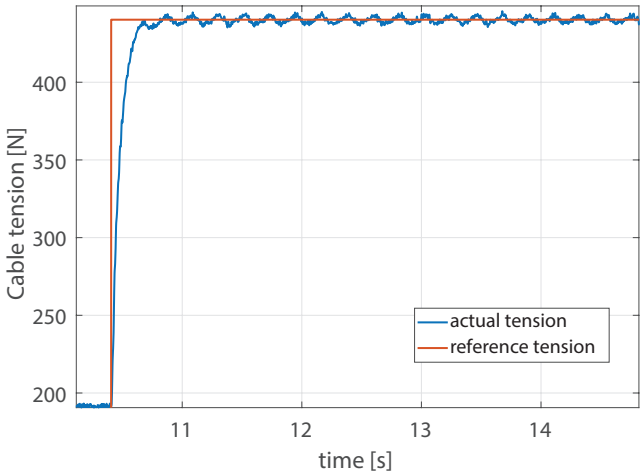

Figure 2.17 – *Measured cable tension with a step input in the desired tension.*

N is 0.4 s. The position x was kept constant. Figure 2.18 shows the results for τ_d varying with low frequency. This figure shows that the obtained error is negligible so that desired and measured cable tensions are virtually equivalent. Moreover, the small error obtained with a slow variation of τ_d shows that the proposed strategy successfully avoids the stick-slip problem.

Figure 2.19 depicts the results for a sinusoidal x and constant τ_d. This figure compares strategies with and without the feedforward term $\dot{x}/r_w$. This figure shows that a

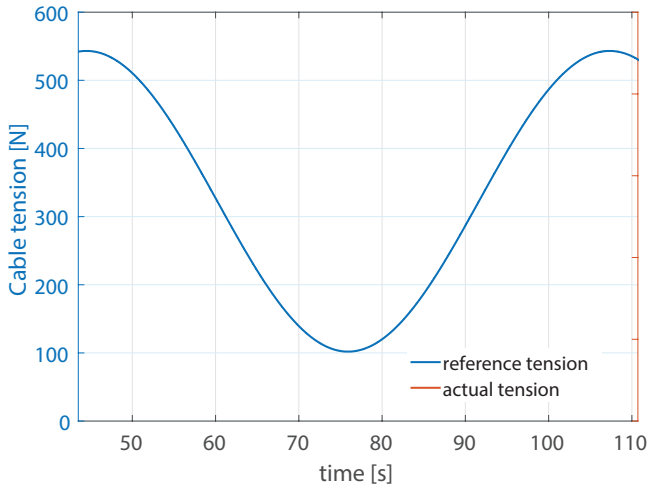

Figure 2.18 – *Measured and desired cable tensions. Displacement x is constant and desired cable tension is determined according $\tau_d = 320 + 220\ \cos(2\pi\, t/62.5)$ [N]. One may note that the obtained error is rather small.*

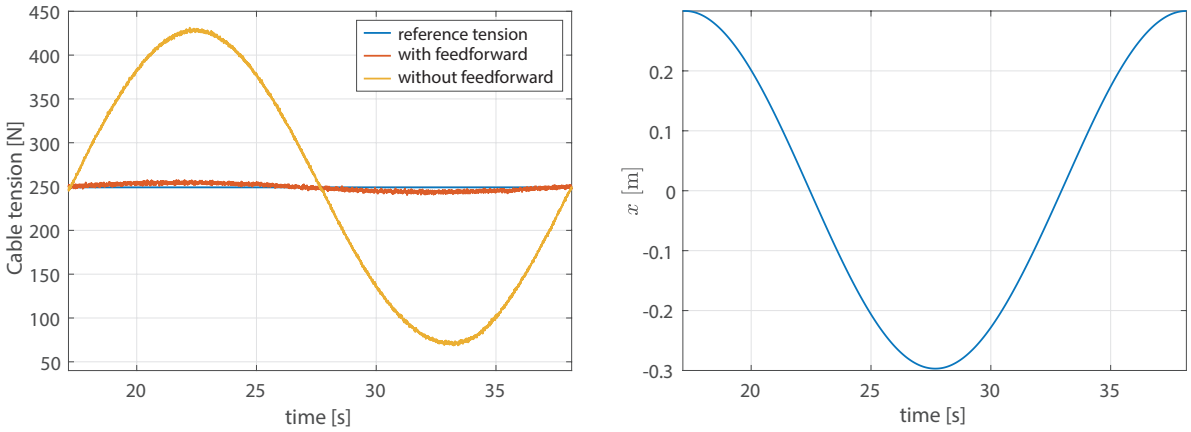

Figure 2.19 – *Experimental results for a constant desired cable tension and sinusoidal displacement x. The chart on the left presets the desired and measured tensions comparing schemes with and without the feedforward term $\frac{\dot{x}}{r_w}$. The chart on the right presents the sinusoidal displacement x .*

control scheme without the feedforward term would lead to large errors when variable x is used. The small errors obtained when the feedforward term is applied indicates that the compensation of $\dot{x}$ is satisfying.

Finally, Figure 2.20 presents the results with sinusoidal x and τ_d. A small delay may be observed. Nevertheless, the obtained results with the proposed control strategy are considered suitable. Accordingly, the velocity-based cable tension control (2.36) is experimentally validated and may be used in the position tracking discussed in the next chapters.

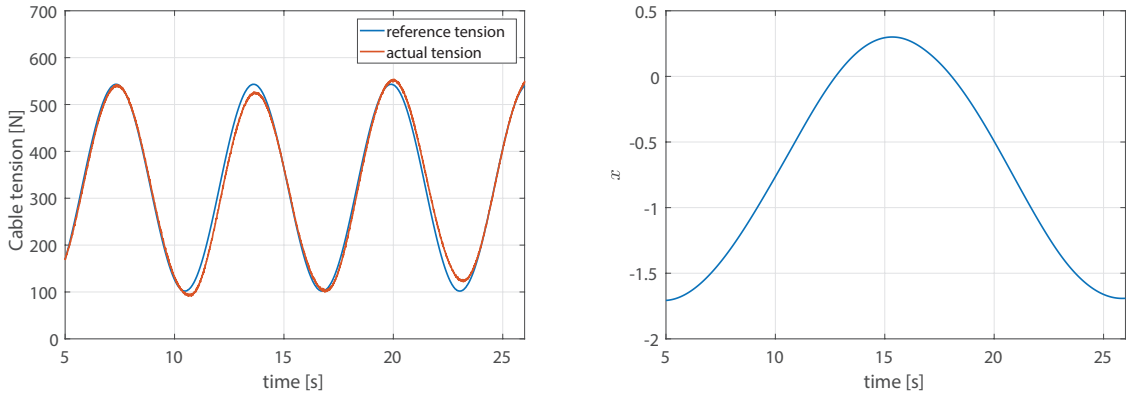

Figure 2.20 – *Measured and desired cable tensions (on the left) and displacements x (on the right).*

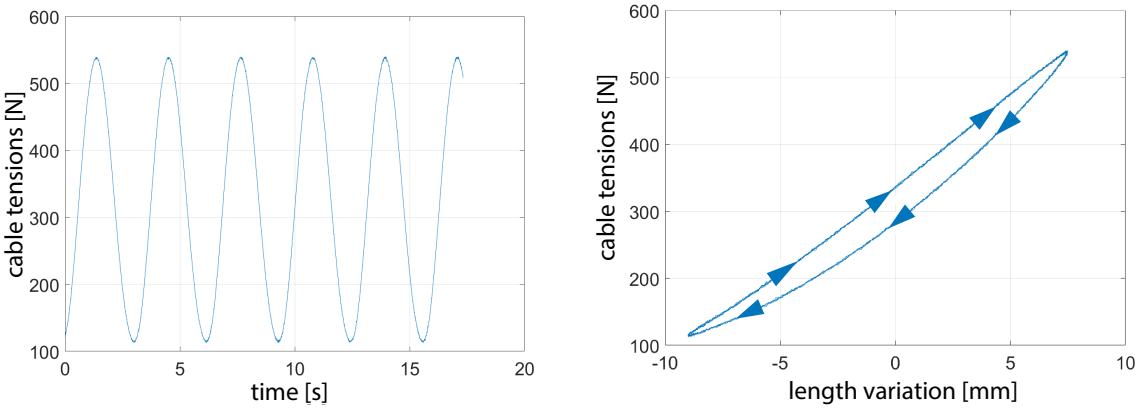

Figure 2.21 – *Experimental results highlighting the hysteresis present on the behavior of the cable elasticity.*

2.3.4 Further results

Some additional results were obtained during the experiments described above. Figure 2.21 shows several repetitions of the experiment depicted in Figure 2.18. One can see that the relation between the cable tension and the displacement of the winch ($q\, r_w$) is repeatable. The plot displaying the elongation vs. cable tension indicates that the relation between stress and strain for increasing cable tensions can be reasonably approximated with an affine function. Nevertheless, the values obtained for decreasing tensions shows that significant hysteresis is present.

As shown in Figure 2.5, there is a second load pin measuring the cable tensions in pulley 2. Figure 2.22 presents the values obtained with both sensors. One may see that the cable tensions measured by sensor 2 shift when the direction of movement changes (velocity equals to zero). This shift is probably caused by the friction in pulleys 3 and 4 shown in Figure 2.5. Since τ_1 is the value used in the control loop, it is continuous and close to the desired cable tension.

In order to clarify the influence of the friction in the pulleys, consider a simplified model

$$\tilde{\tau}_2 = \tau_1 - 2\,\tau_f\ \mathrm{sign}(\dot{x}) + \tau_k, \tag{2.37}$$

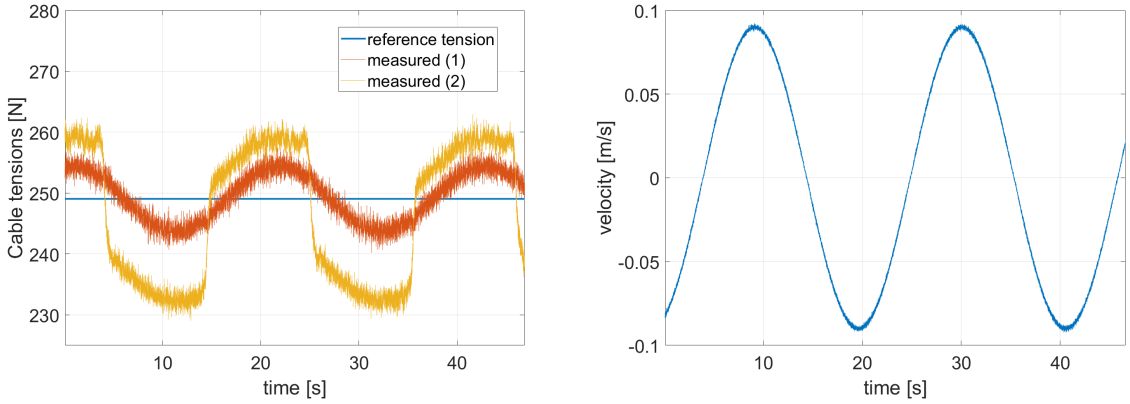

Figure 2.22 – *Difference between cable tensions measured by load pins 1 and 2 (τ_1 and τ_2).*

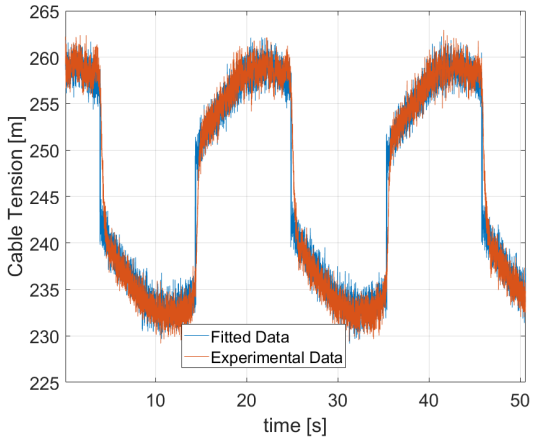

Figure 2.23 – *Identification of the pulley friction: measured cable tension τ_2 (experimental data) and predicted value $\tilde{\tau}_2$ given by* (2.37) *(fitted data).*

where τ_f is the the dry friction force due to pulleys 3 and 4, the friction forces in these pulleys being considered equal. The scalar τ_k is a constant that depends on the initialization of the system. Parameters τ_f and τ_k were identified using data from Figure 2.22. The fitted data is shown in Figure 2.23. The pulley dry friction force obtained is $\tau_f = 3.9$ N.

This correction may be easily integrated in the cable tension control. However, considering the small influence of $\tau_f = 3.9$ N in the interval of feasible cable tensions, this correction is not included in the remainder of this thesis.

3

Linear Model Predictive Control

A Linear Model Predictive Control (LMPC) for the position tracking of CDPRs is proposed and its performance is evaluated through numerical simulations and real-time experiments. Section 3.1 introduces the control algorithm and the dynamic model on which the control scheme is based. In order to simplify the formulation of the corresponding optimal control problem and its numerical solution, a linear discrete-time approximation is proposed based on the original continuous nonlinear dynamic model. Simulation results presented in Section 3.2 indicate that such approximation is pertinent for the Hephaestus prototype. Finally, Section 3.3 details the experimental validation of the proposed linear MPC. These results prove that the MPC scheme copes with the main concerns discussed in Chapter 1, namely, the cable tension limits are explicitly handled as an integral part of the main controller.

Contents

3.1 Control Design

The main motivations to use an MPC strategy for the position tracking control of CDPRs were discussed in Section 1.7. In brief, the feedback correction of the state-of-the-art schemes discussed in Section 1.4 are subjected to feasibility issues since the cable tension limits are not considered in the main computation of the feedback loop. A known advantage related to MPC schemes lies in its ability to explicitly handle the system constraints. Considering that, the present section proposes a linear MPC scheme able to handle the cable tension limits as an integral part of the main feedback control. This MPC is based on the dynamic model discussed in Section 3.1.1, while the control strategy is introduced in Section 3.1.2.

3.1.1 Dynamic model

Before addressing the model itself, some notations are introduced. Sequences of vectors are denoted with bold lowercase letters followed by $(\cdot)$ (as, for instance, $\mathbf{s}(\cdot)$). The set of sequences of vectors with dimension n_s and infinite elements is denoted as $\mathbb{S}^{n_s}$. Similarly, $\mathbb{S}^{n_{s2}}_{n_{s1}}$ denotes the set of sequences containing n_{s1} vectors with dimension n_{s2}. Accordingly, the k^{th} vector of a sequence $\mathbf{s} \in \mathbb{S}^{n_{s2}}_{n_{s1}}$ is denoted as $\mathbf{s}(k) = \mathbf{s}_k \in \mathbb{R}^{n_{s2}}$. The set of strictly positive integers is referred to as $\mathbb{N}$. The set of non-negative integers is $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$. Moreover, for $a,\ b \in \mathbb{R}$, $\mathbb{N}_{a,b} = \{i \in \mathbb{N}_0 \mid a \leqslant i \leqslant b\}$.

This subsection presents the dynamic model of a CDPR consisting of an n-DoF mobile platform driven by m cables, where $n \leq m$. The platform pose in time is given by the vectorial function $\boldsymbol{x} : \mathbb{R} \to \mathbb{R}^n$, so that $\boldsymbol{x}(t) = \begin{bmatrix}\boldsymbol{p}(t)^T & \boldsymbol{\psi}(t)^T\end{bmatrix}^T$ is the pose vector containing both the platform position $\boldsymbol{p}(t)$ and orientation $\boldsymbol{\psi}(t)$ at instant t. The dependence on time is dropped leading to the short notation $\boldsymbol{x}$ whenever there is no risk of confusion. The same practice is used for other vectorial and matricial functions in the remainder of this thesis.

Typically, $\boldsymbol{\psi}$ consists of Euler angles. The common dot notation is used to refer to the time derivatives, so that $\mathrm{d}\boldsymbol{x}/\mathrm{dt} = \dot{\boldsymbol{x}}$ and $\mathrm{d}^2\boldsymbol{x}/\mathrm{dt}^2 = \ddot{\boldsymbol{x}}$. In general, the angular velocity $\boldsymbol{\omega}$ is not equal to the time derivative orientation vector $\boldsymbol{\psi}$. For this reason, the following relations need to be introduced

$$\begin{aligned}\begin{bmatrix}\dot{\boldsymbol{p}}^T & \boldsymbol{\omega}^T\end{bmatrix}^T &= \mathbf{S}(\boldsymbol{x})\,\dot{\boldsymbol{x}} \\ \begin{bmatrix}\ddot{\boldsymbol{p}}^T & \dot{\boldsymbol{\omega}}^T\end{bmatrix}^T &= \mathbf{S}(\boldsymbol{x})\,\ddot{\boldsymbol{x}} + \dot{\mathbf{S}}(\boldsymbol{x}, \dot{\boldsymbol{x}})\,\dot{\boldsymbol{x}},\end{aligned} \tag{3.1}$$

with $\mathbf{S}(\boldsymbol{x})$ a square matrix responsible for transforming $\dot{\boldsymbol{\psi}}$ into ω. On the other hand, $\dot{\mathbf{S}}(\boldsymbol{x}, \dot{\boldsymbol{x}})$ is the component-wise time derivative of $\mathbf{S}(\boldsymbol{x})$.

Considering that the platform pose and cable forces are known, the magnitude τ_i and the direction $\mathbf{u}_i$ of the force applied by the cable $i \in \mathbb{N}_{1,m}$ on the platform are also known. Vector $\mathbf{u}_i$ is shown in Figure 2.1 and can be computed with (2.8). Accordingly, in this thesis, the vectors $\mathbf{u}_i$ are uniquely computed in function of the platform pose. In contrast, if cable sagging is considered, the direction $\mathbf{u}_i$ also varies in function of the

cable tension itself. This variation is neglected here. Therefore, we denote these vectors as vectorial functions $\mathbf{u}_i(\boldsymbol{x})$.

The vectorial function $\boldsymbol{\tau}_c : \mathbb{R} \to \mathbb{R}^m$ returns the vector of cable tensions in function of time $\boldsymbol{\tau}_c(t) = \begin{bmatrix}\tau_1(t) & \dots & \tau_m(t)\end{bmatrix}^T$. Each cable force is applied on the corresponding point B_i of the mobile platform (see Figure 2.1). Summing up the contributions of forces and moments of every cable force, the resultant wrench $\mathbf{f}(\boldsymbol{x}, \boldsymbol{\tau})$ applied on the platform is obtained with the linear mapping

$$\mathbf{f}(\boldsymbol{x}, \boldsymbol{\tau}_c) = \mathbf{W}(\boldsymbol{x})\, \boldsymbol{\tau}_c, \tag{3.2}$$

with the *wrench matrix* $\mathbf{W}(\boldsymbol{x})$ representing this linear mapping. In a general 6-DoF CDPR, as for vectors $\mathbf{u}_i$, the vectors $\mathbf{b}'_i$ in Figure 2.1 may be written as functions of the platform pose with

$$\mathbf{b}'_i(\boldsymbol{x}) = \mathbf{R}(\boldsymbol{\psi})\, \mathbf{b}_{pi}, \tag{3.3}$$

where $\mathbf{R}$ and $\mathbf{b}_{pi}$ were defined in Section 2.1.1. The wrench matrix $\mathbf{W}$ is then by

$$\mathbf{W}(\boldsymbol{x}) = \begin{bmatrix} \mathbf{u}_1(\boldsymbol{x}) & \dots & \mathbf{u}_m(\boldsymbol{x}) \\ \hat{\mathbf{b}}'_1(\boldsymbol{x})\mathbf{u}_1(\boldsymbol{x}) & \dots & \hat{\mathbf{b}}'_m(\boldsymbol{x})\mathbf{u}_m(\boldsymbol{x}) \end{bmatrix}, \tag{3.4}$$

with $\hat{\mathbf{v}}_1$ denoting a skew-symmetric matrix so that $\hat{\mathbf{v}}_1\, \mathbf{v}_2 = \mathbf{v}_1 \times \mathbf{v}_2$ is the cross product between vectors $\mathbf{v}_1,\ \mathbf{v}_2 \in \mathbb{R}^3$.

Knowing the wrench applied by the cables on the platform, its motion is modeled using Newton-Euler formalism, which leads to the following expression:

$$\begin{aligned} \mathbf{M}'(\boldsymbol{x}) \left(\dot{\mathbf{S}}(\dot{\boldsymbol{x}}, \boldsymbol{x})\, \dot{\boldsymbol{x}} + \mathbf{S}(\boldsymbol{x})\, \ddot{\boldsymbol{x}}\right) + \mathbf{C}'(\dot{\boldsymbol{x}}, \boldsymbol{x})\, \mathbf{S}(\boldsymbol{x})\dot{\boldsymbol{x}} = \\ = \mathbf{g}(\boldsymbol{x}) + \mathbf{W}(\boldsymbol{x})\, \boldsymbol{\tau}_c, \end{aligned} \tag{3.5}$$

where matrices $\mathbf{M}'$ and $\mathbf{C}'$ are given by

$$\mathbf{M}'(\boldsymbol{x}) = \begin{bmatrix} m_p\, \mathbf{I} & -m_p\, \hat{\mathbf{c}}(\boldsymbol{x}) \\ m_p\, \hat{\mathbf{c}}(\boldsymbol{x}) & \mathbf{H}(\boldsymbol{x}) \end{bmatrix} \text{ and} \tag{3.6}$$

$$\mathbf{C}'(\boldsymbol{x}, \dot{\boldsymbol{x}})\, \dot{\boldsymbol{x}} = \begin{bmatrix} m_p\, \hat{\boldsymbol{\omega}}_{\boldsymbol{x}}\, \hat{\boldsymbol{\omega}}_{\boldsymbol{x}}\, \mathbf{c}(\boldsymbol{x}) \\ \hat{\boldsymbol{\omega}}_{\boldsymbol{x}}\, \mathbf{H}(\boldsymbol{x})\, \boldsymbol{\omega}_{\boldsymbol{x}} \end{bmatrix}. \tag{3.7}$$

The scalar m_p is the platform mass and $\mathbf{I}$ is the identity matrix with dimensions suitably chosen. Denoting as $\mathbf{c}_p$ the vector going from the platform geometric center to its center of mass, expressed in the coordinate system $\mathcal{O}_p$ attached to the platform, the corresponding vector in the global coordinate system is computed as

$$\mathbf{c}(\boldsymbol{x}) = \mathbf{R}(\boldsymbol{\psi})\mathbf{c}_p = \begin{bmatrix} c_x & c_y & c_z \end{bmatrix}^T. \tag{3.8}$$

The angular velocity $\boldsymbol{\omega}_{\boldsymbol{x}}$ is obtained from $\boldsymbol{x}$ according to (3.1). The matrix $\mathbf{H}$ is defined as $\mathbf{H}(\boldsymbol{x}) = \mathbf{R}(\boldsymbol{\psi})\, \mathbf{I}_G\, \mathbf{R}(\boldsymbol{\psi})^T + m_p\, \hat{\mathbf{c}}(\boldsymbol{x})\, \hat{\mathbf{c}}(\boldsymbol{x})^T$ where $\mathbf{I}_G$ is the platform inertia matrix in respect to the coordinate system $\mathcal{O}_p$. The vector of gravitational forces is $\mathbf{g}(\boldsymbol{x}) = m_p\, g \begin{bmatrix} 0 & 0 & -1 & -c_y & c_x & 0 \end{bmatrix}^T$, with g the gravitational acceleration.

Denoting

$$\begin{aligned}\mathbf{M}(\boldsymbol{x}) &= \mathbf{M}'(\boldsymbol{x})\,\mathbf{S}(\boldsymbol{x}) \;\text{ and}\\ \mathbf{C}(\boldsymbol{x},\dot{\boldsymbol{x}}) &= \mathbf{C}'(\boldsymbol{x},\dot{\boldsymbol{x}})\,\mathbf{S}(\boldsymbol{x}) + \mathbf{M}'(\boldsymbol{x})\,\dot{\mathbf{S}}(\boldsymbol{x},\dot{\boldsymbol{x}}),\end{aligned} \tag{3.9}$$

the dynamic system (3.5) can be rewritten as

$$\mathbf{M}(\boldsymbol{x})\,\ddot{\boldsymbol{x}} + \mathbf{C}(\boldsymbol{x},\dot{\boldsymbol{x}})\,\dot{\boldsymbol{x}} = \mathbf{g}(\boldsymbol{x}) + \mathbf{W}(\boldsymbol{x})\,\boldsymbol{\tau}_c. \tag{3.10}$$

The dynamic model (3.10) is considered as the nominal system in the remainder of this thesis. This is a continuous-time nonlinear system. Nevertheless, as discussed in Section 1.7, the MPC strategy proposed in this chapter needs a linear discrete-time system. To this end, the sequences of vectors $\mathbf{x}(\cdot),\ \dot{\mathbf{x}}(\cdot),\ \ddot{\mathbf{x}}(\cdot) \in \mathbb{S}^n$ and $\mathbf{y}(\cdot) \in \mathbb{S}^{2n}$ are defined as

$$\begin{aligned}\mathbf{x}(k) &= \mathbf{x}_k = \boldsymbol{x}(t_0 + k\,\Delta t)\\ \dot{\mathbf{x}}(k) &= \dot{\mathbf{x}}_k = \dot{\boldsymbol{x}}(t_0 + k\,\Delta t)\\ \ddot{\mathbf{x}}(k) &= \ddot{\mathbf{x}}_k = \ddot{\boldsymbol{x}}(t_0 + k\,\Delta t)\\ \mathbf{y}(k) &= \mathbf{y}_k = \begin{bmatrix}\mathbf{x}_k^T & \dot{\mathbf{x}}_k^T\end{bmatrix}^T\end{aligned} \tag{3.11}$$

for all $k \in \mathbb{N}_0$, an initial time t_0 and a sampling period Δt. Without loss of generality, since the continuous system (3.10) is time-invariant (as defined in [62, Chapter 1]) the initial time is considered to be $t_0 = 0$.

Similarly, considering a digital control approach, the continuous time representation of the actual cable tensions $\boldsymbol{\tau}_c$ is considered piece-wise constant and given by its discrete counterpart $\boldsymbol{\tau}(\cdot) \in \mathbb{S}^m$, *i.e.*

$$\boldsymbol{\tau}_c(t) = \boldsymbol{\tau}(k) = \boldsymbol{\tau}_k, \;\; \text{for } t \in \Big\{ t \;\Big|\; (k-1)\Delta t \leqslant t < k\,\Delta t \Big\} \text{ and } k \in \mathbb{N}. \tag{3.12}$$

Using the Euler method of numerical integration, the state space representation of (3.10) may be approximated by

$$\begin{aligned}\mathbf{y}_{k+1} \approx \overbrace{\begin{bmatrix}\mathbf{I} & \Delta t\,\mathbf{I}\\ \mathbf{0} & \mathbf{I}\end{bmatrix}}^{\mathbf{A}} \mathbf{y}_k + \overbrace{\begin{bmatrix}\mathbf{0}\\ \Delta t\,\mathbf{M}(\mathbf{x}_{\mathbf{y}_k})^{-1}\,\mathbf{W}(\mathbf{x}_{\mathbf{y}_k})\end{bmatrix}}^{\mathbf{B}(\mathbf{y}_k)} \boldsymbol{\tau}_k +\\ + \underbrace{\begin{bmatrix}\mathbf{0}\\ \Delta t\,\mathbf{M}(\mathbf{x}_{\mathbf{y}_k})^{-1}\Big(\mathbf{g}(\mathbf{x}_{\mathbf{y}_k}) - \mathbf{C}(\mathbf{x}_{\mathbf{y}_k},\dot{\mathbf{x}}_{\mathbf{y}_k})\dot{\mathbf{x}}_{\mathbf{y}_k}\Big)\end{bmatrix}}_{\mathbf{v}(\mathbf{y}_k)}.\end{aligned} \tag{3.13}$$

with $\mathbf{x}_{\mathbf{y}},\ \dot{\mathbf{x}}_{\mathbf{y}} \in \mathbb{R}^n$ denoting the pose and velocity vectors extracted from a vector $\mathbf{y} \in \mathbb{R}^{2n}$. Additionally, $\mathbf{0}$ denotes a matrix or vector with dimensions suitably chosen and all its elements equal to zero.

In accordance with the notation introduced in the equation above, the considered approximated discrete model is

$$\mathbf{y}_{k+1} \approx \mathbf{A}\,\mathbf{y}_k + \mathbf{B}(\mathbf{y}_k)\,\boldsymbol{\tau}_k + \mathbf{v}(\mathbf{y}_k). \tag{3.14}$$

3.1.2 Control strategy

As introduced in Section 1.7, a key element of an MPC algorithm is the formulation of an OCP based on a bounded time span. This time span is determined by $h_p \in \mathbb{N}$, called *prediction horizon*. For given instant k, state $\mathbf{y}_k$ and finite sequence of cable tensions $\boldsymbol{\tau} \in \mathbb{S}^m_{h_p}$, an estimation of the sequence of future states should be defined based on the discrete-model (3.14).

3.1.2.1 State Predictions

In order to simplify the computation of the estimated future states, the variations of the matrix $\mathbf{B}$ and the vector $\mathbf{v}$ in (3.14) within the prediction horizon is assumed to be negligible. Therefore, the following approximations are considered

$$\begin{matrix} \mathbf{B}(\mathbf{y}_{k+i}) \approx \mathbf{B}(\mathbf{y}_k) \\ \mathbf{v}(\mathbf{y}_{k+i}) \approx \mathbf{v}(\mathbf{y}_k) \end{matrix} \quad \forall\, i \in \mathbb{N}_{1,h_p}. \tag{3.15}$$

In other words, for a given time instant k, these matrices are considered to be constant within the prediction horizon and are denoted shortly as $\mathbf{B} = \mathbf{B}(\mathbf{y}_k)$ and $\mathbf{v} = \mathbf{v}(\mathbf{y}_k)$.

Moreover, as Maciejowski discusses in [121], considering every possible combination of cable tensions in the prediction horizon may lead to a bad compromise between performance improvement and computational burden. Therefore, it is interesting to consider every combination within a reduced horizon $h_c < h_p$, denoted as *control horizon*, and constraint the last i^{th} components with $h_c < i \leqslant h_p$, as follows

$$\boldsymbol{\tau}_{k+i} = \boldsymbol{\tau}_{k+h_c} \; \forall\, i \in \mathbb{N}_{h_c,h_p}. \tag{3.16}$$

Using the proposed notations and assumptions, the future states along the prediction horizon are estimated applying recursively (3.14) with the identities (3.15) yielding to

$$\overbrace{\begin{bmatrix} \tilde{\mathbf{y}}_{k+1} \\ \tilde{\mathbf{y}}_{k+2} \\ \tilde{\mathbf{y}}_{k+3} \\ \vdots \\ \tilde{\mathbf{y}}_{k+h_c+1} \\ \vdots \\ \tilde{\mathbf{y}}_{k+h_p} \end{bmatrix}}^{\boldsymbol{\gamma}(\mathbf{y}_k,\mathbf{u})} = \overbrace{\begin{bmatrix} \mathbf{A} \\ \mathbf{A}^2 \\ \mathbf{A}^3 \\ \vdots \\ \mathbf{A}^{h_c+1} \\ \vdots \\ \mathbf{A}^{h_p} \end{bmatrix}}^{\mathbf{D}} \mathbf{y}_k + \overbrace{\begin{bmatrix} \mathbf{B} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{AB} & \mathbf{B} & 0 & \vdots \\ \mathbf{A}^2\mathbf{B} & \mathbf{AB} & \ddots & \mathbf{0} \\ \vdots & \vdots & & \vdots \\ \mathbf{A}^{h_c}\mathbf{B} & \mathbf{A}^{h_c-1}\mathbf{B} & \dots & \mathbf{B} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{A}^{h_p-1}\mathbf{B} & \dots & \mathbf{A}^{h_p-h_c+1}\mathbf{B} & \sum_{i=0}^{h_p-h_c} \mathbf{A}^i\,\mathbf{B} \end{bmatrix}}^{\mathbf{E}(\mathbf{y}_k)} \overbrace{\begin{bmatrix} \tau_k \\ \tau_{k+1} \\ \vdots \\ \tau_{k+h_c} \end{bmatrix}}^{\mathbf{u}} + \overbrace{\begin{bmatrix} \mathbf{v} \\ \mathbf{A}\,\mathbf{v} \\ \mathbf{A}^2\mathbf{v} \\ \vdots \\ \mathbf{A}^{h_c}\mathbf{v} \\ \vdots \\ \mathbf{A}^{h_p-1}\mathbf{v} \end{bmatrix}}^{\boldsymbol{\nu}(\mathbf{y}_k)}$$

or, with a compact notation,

$$\boldsymbol{\gamma}(\mathbf{y}, \mathbf{u}) = \mathbf{D}\mathbf{y} + \mathbf{E}(\mathbf{y})\,\mathbf{u} + \boldsymbol{\nu}(\mathbf{y}). \tag{3.17}$$

Note that, since $\mathbf{B}$ and $\mathbf{v}$ are functions of $\mathbf{y}$, $\mathbf{E}$ and $\boldsymbol{\nu}$ are also functions of $\mathbf{y}$.

3.1.2.2 Optimal Control Problem Formulation

Our goal is to define a functional cost of the MPC scheme including (i) the predicted errors, (ii) the magnitude of the cable tensions and (iii) the variation of the cable tensions. The minimization of the cable tensions variation (iii) is important mainly because large variations of cable tensions may be not physically feasible. This issue is also relevant during the initialization of the robot. If the initial cable tensions are not close to an optimal cable tension distribution, the MPC will automatically find a smooth transition between these two sets of cable tensions.

Consider sequences $\mathbf{x}_d(\cdot),\ \dot{\mathbf{x}}_d(\cdot) \in \mathbb{S}^n_{n_t}$ describing a desired trajectory, with $t_f = n_t\,\Delta t$ the trajectory final time. The vector of desired states within a prediction horizon h_p for an instant k is $\boldsymbol{\gamma}_d(k) = \begin{bmatrix}\mathbf{x}_d(k+1)^T & \dot{\mathbf{x}}_d(k+1)^T & \dots & \mathbf{x}_d(k+h_p)^T & \dot{\mathbf{x}}_d(k+h_p)^T\end{bmatrix}$. The weighted expressions of (i), (ii) and (iii) are then respectively given by

$$J_\gamma(\mathbf{y},\mathbf{u},k) = \|\boldsymbol{\gamma}_d(k) - \boldsymbol{\gamma}(\mathbf{y},\mathbf{u})\|^2_{\mathbf{K}_\gamma} \tag{3.18}$$

$$J_u(\mathbf{u}) = \|\mathbf{u}\|^2_{\mathbf{K}_u} \tag{3.19}$$

$$J_D(\mathbf{u},\boldsymbol{\tau}_c(k\,\Delta t)) = k_{\Delta u}\sum_{i=0}^{hc-1}\|\Delta\boldsymbol{\tau}_{i+k}\|^2 \tag{3.20}$$

where $\mathbf{K}_\gamma$ and $\mathbf{K}_u$ are positive definite diagonal weighting matrices and $k_{\Delta u}$ is a positive scalar. In addition, $\|\mathbf{w}\|^2_{\mathbf{K}} = \mathbf{w}^T\mathbf{K}\,\mathbf{w}$ denotes the weighted norm of $\mathbf{w} \in \mathbb{R}^{n_w}$ with a weighting symmetric positive semi-definite matrix $\mathbf{K} \in \mathbb{R}^{n_w \times n_w}$. The vector of the variation of cable tensions can be expressed as

$$\Delta\boldsymbol{\tau}_i = \begin{cases} \boldsymbol{\tau}_\mathbf{u}(i) - \boldsymbol{\tau}_\mathbf{u}(i-1), \text{ if } i > 1 \\ \boldsymbol{\tau}_\mathbf{u}(1) - \boldsymbol{\tau}_c(k\,\Delta t), \text{ if } i = 1 \end{cases}, \tag{3.21}$$

where $\boldsymbol{\tau}_\mathbf{u}(i) \in \mathbb{R}^m$ is obtained by extracting the i^{th} vector of cable tensions of $\mathbf{u}$. Alternatively, (3.21) may be written as

$$\Delta\mathbf{u} = \begin{bmatrix}\Delta\boldsymbol{\tau}_1^T & \dots & \Delta\boldsymbol{\tau}_{h_c}^T\end{bmatrix}^T = \mathbf{Q}\,\mathbf{u} - \mathbf{z}, \tag{3.22}$$

with

$$\mathbf{Q} = \begin{bmatrix} \boldsymbol{I}_m & \mathbf{0} & \dots & & \\ -\boldsymbol{I}_m & \boldsymbol{I}_m & \mathbf{0} & \dots & \\ \mathbf{0} & -\boldsymbol{I}_m & \boldsymbol{I}_m & \dots & \\ \vdots & \ddots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \dots & \dots & -\boldsymbol{I}_m & \boldsymbol{I}_m \end{bmatrix},\ \mathbf{z} = \begin{bmatrix}\boldsymbol{\tau}_c(k\,\Delta t) \\ \mathbf{0} \\ \vdots \\ \mathbf{0}\end{bmatrix}.$$

Note that $\mathbf{z}$ depends on the measured cable tensions. Alternatively, the desired cable tension obtained at $t - \Delta t$ may be used. The dimensions of $\mathbf{Q}$ and $\mathbf{z}$ are $m\,h_c \times m\,h_c$ and $m\,h_c \times 1$, respectively.

Defining $\mathbf{K}_D = k_{\Delta u}\,\mathbf{I}$, J_D can be rewritten as

$$J_D(\mathbf{u},\boldsymbol{\tau}_c(k\,\Delta t)) = \mathbf{u}^T\mathbf{Q}^T\mathbf{K}_D\,\mathbf{Q}\,\mathbf{u} - 2\,\mathbf{z}^T\mathbf{K}_D\mathbf{Q}\,\mathbf{u} + \mathbf{z}^T\mathbf{K}_D\,\mathbf{z}. \tag{3.23}$$

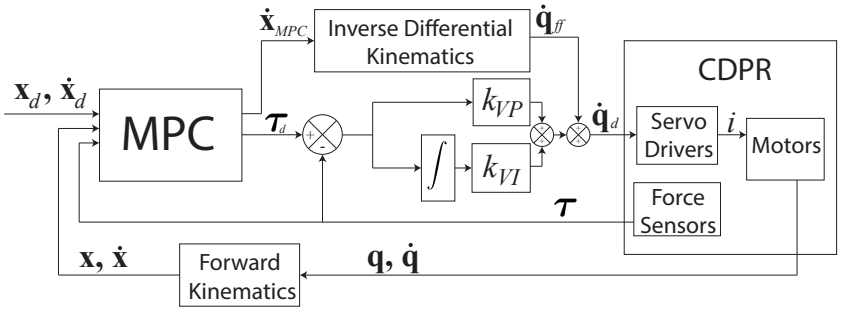

Figure 3.1 – *Block diagram of the proposed control scheme*

Using (3.17), the overall cost function $J(\mathbf{y}, \mathbf{u})$ is defined as the sum of J_γ, J_u and J_D neglecting the terms independent of $\mathbf{u}$ according to

$$\begin{aligned} J(\mathbf{y}, \mathbf{u}, k, \boldsymbol{\tau}_c(k\,\Delta t)) = \mathbf{u}^T \overbrace{\left(\mathbf{E}^T \mathbf{K}_\gamma\, \mathbf{E} + \mathbf{K}_u + \mathbf{Q}^T \mathbf{K}_D\, \mathbf{Q}\right)}^{\mathbf{H}_c} \mathbf{u} + \\ + 2 \underbrace{\left((\mathbf{D}\,\mathbf{y} + \mathbf{F} - \boldsymbol{\gamma}_d(k))^T \mathbf{K}_\gamma\, \mathbf{E} - \mathbf{z}^T \mathbf{K}_D \mathbf{Q} \right)}_{\mathbf{d}^T} \mathbf{u}. \end{aligned} \tag{3.24}$$

The proposed MPC scheme consists in computing the optimal sequence of cable tensions by solving in real-time the following QP problem

$$\begin{aligned} \mathbf{u}^* = \arg\min_{\mathbf{u}} \quad & \frac{1}{2} \mathbf{u}^T \mathbf{H}_c \mathbf{u} + \mathbf{d}^\mathbf{T} \mathbf{u} \\ \text{s.t.} \quad & \mathbf{u}_{min} \leqslant \mathbf{u} \leqslant \mathbf{u}_{max} \end{aligned} \tag{3.25}$$

where $\mathbf{u}_{min},\ \mathbf{u}_{max} \in \mathbb{R}^{n\,h_c}$ are given by

$$\begin{aligned} \mathbf{u}_{min} &= \begin{bmatrix} \boldsymbol{\tau}_{min}^T & \dots & \boldsymbol{\tau}_{min}^T \end{bmatrix}^T \text{ and} \\ \mathbf{u}_{max} &= \begin{bmatrix} \boldsymbol{\tau}_{max}^T & \dots & \boldsymbol{\tau}_{max}^T \end{bmatrix}^T. \end{aligned} \tag{3.26}$$

The optimal solution $\mathbf{u}^* = \begin{bmatrix} \boldsymbol{\tau}_k^{*T} & \dots & \boldsymbol{\tau}_{k+h_c}^{*\,T} \end{bmatrix}^T$ represents the optimal control inputs over the control horizon. The vector of desired cable tension is set as $\boldsymbol{\tau}_d(t) = \boldsymbol{\tau}_k^*$ for $(k-1)\Delta t \leqslant t < k\,\Delta t$. The solution of (3.25) is repeated at each sampling time, updating matrices $\mathbf{H}_c$ and $\mathbf{d}$ in function of $\mathbf{y}_k$ and $\boldsymbol{\tau}_c(k\,\Delta t)$.

The desired set of cable tensions is used as setpoint for the cable tension control introduced in Section 2.3.1. The cable tension control is responsible for generating actual cable tensions $\boldsymbol{\tau}_c(t)$ for $(k-1)\Delta t \leqslant t < k\,\Delta t$ as close as possible to $\boldsymbol{\tau}_k^*$.

Figure 3.1 depicts the block diagram of the implemented control scheme, showing that the kinematic model and cable tension control introduced in Chapter 2 are necessary complements to the MPC strategy. The Cartesian velocity $\dot{\mathbf{x}}_{MPC}$ predicted by the MPC is used in the cable tension control. This vector is obtained with $\mathbf{y}_{k+1} = \mathbf{A}\,\mathbf{y}_k + \mathbf{B}\,\boldsymbol{\tau}_k + \mathbf{v}$.

Alternatively to the optimization problem (3.25), the constraints may be changed using

$$\begin{aligned} \mathbf{u}^* = \arg\min_{\mathbf{U}} \quad & \frac{1}{2}\mathbf{u}^T\mathbf{H}_c\mathbf{u} + \mathbf{d}^\mathbf{T}\mathbf{u} \\ \text{s.t.} \quad & \mathbf{u}_{min} \leqslant \mathbf{u} \leqslant \mathbf{u}_{max} \\ & |\Delta\mathbf{u}| \leqslant \Delta u_{max} \end{aligned} \tag{3.27}$$

with $|\Delta\mathbf{u}|$ representing the absolute values of $\Delta\mathbf{u}$ given by (3.22) and a scalar Δu_{max} limiting the maximal cable tension variations. This way, the variation of cable tensions is explicitly constrained. In this case, the scalar $k_{\Delta u}$ may be set to 0.

3.2 Numerical Simulations

The present section compares through numerical simulations the performance obtained with the MPC scheme proposed in Section 3.1 with those obtained with state-of-the-art strategies. To this end, the control schemes with which the proposed MPC is compared should be revisited. The results presented in this section were published in [182].

3.2.1 Background on the state-of-the-art controllers

The performance obtained with the proposed MPC is compared to those led by strategies commonly used for the control of CDPRs. Namely, MPC is compared to: (i) a linear PID+ controller [183] and (ii) a Sliding Mode Control (SMC) [70]. A brief description of these two control methods is presented in the sequel.

For a given reference trajectory in time $t_0 \leqslant t \leqslant t_f$, the desired poses, velocities and accelerations are denoted by $\boldsymbol{x}_d(t)$, $\dot{\boldsymbol{x}}_d(t)$ and $\ddot{\boldsymbol{x}}_d(t)$, respectively. At a given instant t, the error in the Cartesian space is expressed as $\mathbf{e}_x(t) = \boldsymbol{x}_d(t) - \boldsymbol{x}(t)$. Similarly, the error in joint space is denoted by $\mathbf{e}_j(t) = \boldsymbol{l}_d(t) - \boldsymbol{l}(t)$, where $\boldsymbol{l}_d(t)$ is the vector of desired cable lengths obtained from $\boldsymbol{x}_d(t)$ with the inverse kinematics and $\boldsymbol{l}(t)$ is the estimated current cable length vector computed according to motor positions.

3.2.1.1 PID + computed torque

This control strategy, hereinafter referred to as PID+, applies the following wrench

$$\mathbf{f} = \mathbf{M}(\boldsymbol{x})\ddot{\boldsymbol{x}}_d + \mathbf{C}(\boldsymbol{x}, \dot{\boldsymbol{x}})\,\dot{\boldsymbol{x}}_d - \mathbf{g}(\boldsymbol{x}) + \mathbf{W}(\boldsymbol{x})\left(\mathbf{K}_p\,\mathbf{e}_j + \mathbf{K}_i\int_{t_i}^{t}\mathbf{e}_j(\tau)\,d\tau + \mathbf{K}_d\,\dot{\mathbf{e}}_j\right), \tag{3.28}$$

where $\mathbf{K}_p$, $\mathbf{K}_i$ and $\mathbf{K}_d$ are diagonal matrices containing the linear feedback PID gains.

3.2.1.2 Sliding Mode Control

The SMC strategy defines a sliding surface $\mathbf{s} = \mathbf{e}_x + \mathbf{C}_e\,\dot{\mathbf{e}}_x$, with $\mathbf{C}_e$ a positive definite diagonal matrix. The wrench control law to be applied on the platform is

$$\mathbf{f} = \mathbf{M}\left(\ddot{\boldsymbol{x}}_d + \mathbf{C}_d\,(\dot{\boldsymbol{x}}_d - \dot{\boldsymbol{x}}) + \mathbf{K}\,\mathrm{sat}(\mathbf{s}) + \mathbf{Q}\,\mathbf{s}\right) + \mathbf{C}(\boldsymbol{x}, \dot{\boldsymbol{x}})\,\dot{\boldsymbol{x}}_d - \mathbf{g}(\boldsymbol{x}), \tag{3.29}$$

where $\mathbf{K}$, $\mathbf{Q}$ and $\mathbf{C}_d$ are diagonal feedback gain matrices. The function $\mathrm{sat}(\mathbf{s})$ is a continuous approximation of the sign function. Each component of this vector-valued function is calculated as follows

$$\mathrm{sat}(s_i) = \begin{cases} 1, & \text{if} \quad s_i > \Delta \\ \frac{s_i}{\Delta}, & \text{if} \quad |s_i| \leqslant \Delta \\ -1, & \text{if} \quad s_i < \Delta \end{cases} . \tag{3.30}$$

The resulting function presents the same output than the $\mathrm{sign}(s)$ function except for the interval $-\Delta \leqslant s \leqslant \Delta$ in which a linear interpolation eliminates the discontinuity.

3.2.1.3 Redundancy Resolution

The two above control strategies define the wrench $\mathbf{f}$ to be applied on the platform. The final control output is the vector of cable tensions $\boldsymbol{\tau}$ (or motor torques). For fully-constrained CDPRs, $m > n$ and some TDA should be implemented, as discussed in Section 1.5. In the simulations described in this section, the 2-norm of the vector of cable tensions is minimized according to

$$\min_{\boldsymbol{\tau}} \|\boldsymbol{\tau}\|_2 \tag{3.31}$$

$$\text{s.t. } \mathbf{W}\,\boldsymbol{\tau} = \mathbf{f} \tag{3.32}$$

$$\boldsymbol{\tau}_{min} \leqslant \boldsymbol{\tau} \leqslant \boldsymbol{\tau}_{max} \tag{3.33}$$

Satisfying the constraints above, the tension distribution generates the desired wrench (3.32) with cable tensions in an admissible interval (3.33).

In some cases, the wrench demanded by the controller may be not feasible. More precisely, in the space of cable tensions, the intersection of the subspaces defined by the constraints (3.32) and (3.33) is empty. Another strategy should then be defined and the following optimization problem is proposed

$$\begin{aligned} &\min_{\boldsymbol{\tau}} \|\mathbf{W}\,\boldsymbol{\tau} - \mathbf{f}\|_{\mathbf{P}} \\ &\text{s.t. } \boldsymbol{\tau}_{min} \leqslant \boldsymbol{\tau} \leqslant \boldsymbol{\tau}_{max} \end{aligned} \tag{3.34}$$

where the subscript $\mathbf{P}$ indicates that the 2-norm is calculated with a weighting positive definite diagonal matrix $\mathbf{P}$, which is necessary since $\mathbf{f}$ has components with inconsistent units (forces and moments).

3.2.2 Simulations results

This section presents simulation results comparing the performances obtained with the control strategies presented in subsections 3.1 and 3.2.1. The OCP formulation constraining cable tension variations according to (3.27) is used. The context of these simulations is the project Hephaestus described in Section 1.2. A large-dimension CDPR is intended to automatize several tasks in the construction and maintenance of building facades. The main task of the CDPR is the installation of curtain wall modules. The robot workspace is a rectangular region in front of the building facade. Thereby, the CDPR mobile platform can pick up curtain wall modules on the ground and position them where needed on the building facade. Since the CDPR will operate in an outdoor environment, it will be subjected to external disturbances. One of the main concerns is the incidence of wind gusts. For this reason, the simulations presented in this section are focused on external disturbance rejection performances of the studied control strategies. An impulsive disturbance is applied and the response of the CDPR is analyzed. Note that the simulated trajectory is relatively short. However, the simulation of a longer trajectory would not affect the results which highlight the disturbance rejection capabilities of each control strategy.

The initial and final positions are depicted in Figure 3.2. The path between these two positions is a straight line segment. The trajectory is the fastest possible respecting upper bounds on linear velocities, accelerations and jerks. These bounds are 0.3 m/s, 0.3 m/s^2 and 1.0 m/s^3, respectively. The resulting trajectory has continuous derivatives up to the acceleration level. The desired orientation of the platform is constant along the trajectory. An impulsive disturbance $\mathbf{f}_d$ is applied at the instant $t = 2$ s. This impulsive wrench is $\mathbf{f}_d = \begin{bmatrix} 55 & 55 & 550 & 0 & 0 & 0 \end{bmatrix}^T$ (N and Nm) and is applied at the reference point of the platform.

The CDPR configuration (cable drawing points, cable-platform attachments, and cable arrangement) can also be seen in Figure 3.2. Moreover, the parameters of the CDPR dynamic model are the following: $\boldsymbol{\tau}_{min} = 100$ N, $\boldsymbol{\tau}_{max} = 14$ kN, $m_p = 1000$ kg, $\mathbf{c}_p = [0\ 5\ 0]^T$ m, $\mathbf{I}_G = \text{diag}([400\ 100\ 400])$ kg.m^2.

In the following, the results obtained with the three studied motion control strategies are presented and discussed. The control parameters used in the simulations are the following: $\mathbf{K}_p = 71400\,\mathbf{I}$, $\mathbf{K}_i = 71400\,\mathbf{I}$, $\mathbf{K}_d = 71400\,\mathbf{I}$, $\mathbf{C}_e = 2 \times 10^{-3}\,\mathbf{I}$, $\mathbf{C}_d = 36\,\mathbf{I}$, $\mathbf{Q} = 40\,\mathbf{I}$, $\mathbf{K} = 0.2\,\mathbf{I}$, $h_p = 20$, $\Delta_t = 6\times10^{-4}$, $\Delta_r = 5\times10^{-1}$, $K_{\gamma,p} = 6\times10^{9}$, $K_{\gamma,v} = 1\times10^{-2}$, $\mathbf{K}_u = 2 \times 10^{-5}\,\mathbf{I}$ and $k_{\Delta u} = 0$. These gains were tuned with trial-and-error method.

The components of $\mathbf{K}_\gamma$ related to the pose errors are set equal to the scalar $K_{\gamma,p}$, whereas $K_{\gamma,v}$ is used for velocity errors. Value of Δ_t is used in (3.30) as Δ for translational inputs of sat(s), and Δ_r is used for rotational inputs.

Figure 3.3 shows the evolution versus time of the norms of the translational and rotational errors. All the control strategies are able to compensate the tracking error caused by the applied external disturbance, but PID+ presents an oscillatory behavior and an increased settling time. SMC responds faster and without oscillations. MPC presents the fastest response, resulting in the smallest tracking errors along the whole

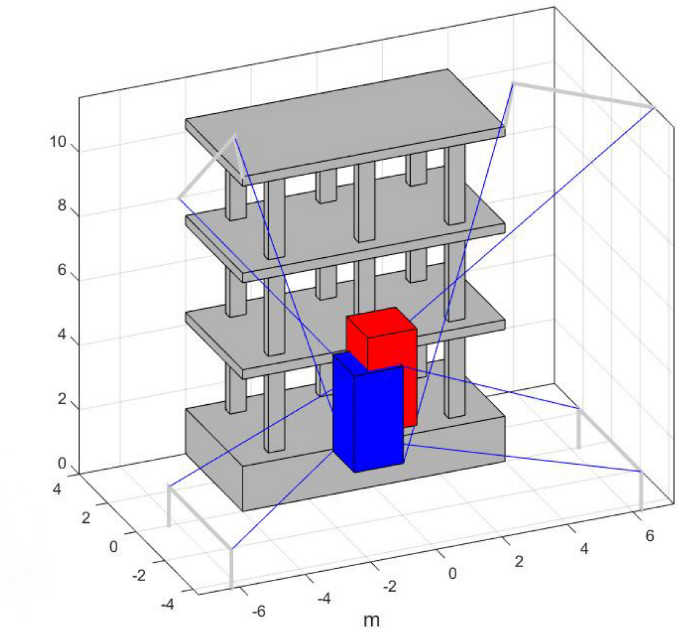

Figure 3.2 – *Illustration of the initial and final positions of the simulated trajectory.*

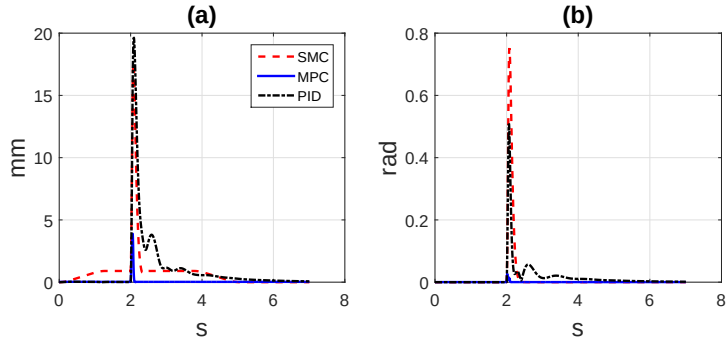

Figure 3.3 – *Norm of the (a) translational and (b) rotational errors.*

trajectory.

The histograms of Figure 3.4 present a performance comparison on different aspects. Let e_t be the 2-norm of the translational error. Histogram (a) quantifies the maximum value e_t over the trajectory. Histogram (b) compares the RMS value of e_t along the trajectory. Taking these two performance measures, MPC leads to the smallest error.

Regarding cable tension values, as shown in Figure 3.4-(c), MPC demands the maximal allowed value $\tau_{max} = 14$ kN. This is also visible in Figure 3.5, which depicts the cable tensions near the instant of application of the impulsive disturbance. Indeed, as discussed earlier, the main advantage of MPC is that the controller takes into account the constraints of the system and optimizes the control actions in order to reduce the tracking errors. Here, the maximum allowed cable tension is an active constraint just after the impulsive disturbance is applied. In the case of SMC, the maximum tension value is 13.7 kN along the trajectory, which indicates that this controller response is close to the largest admissible value $\tau_{max} = 14$ kN. If higher gains were used, the

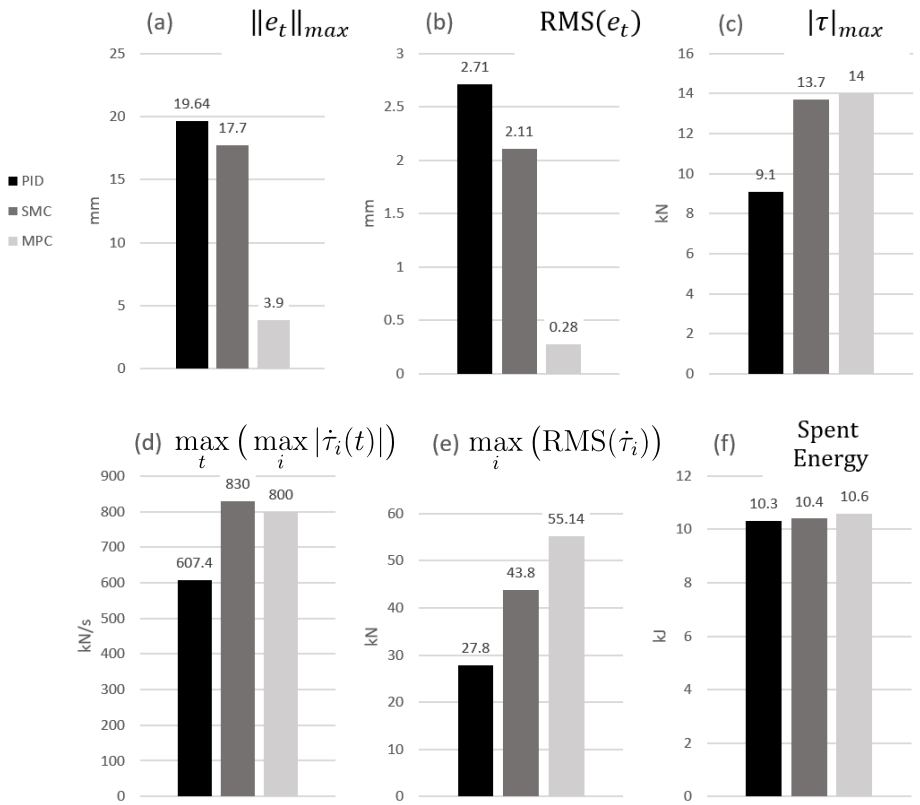

Figure 3.4 – *Comparative results: (a) maximal errors, (b) RMS errors, (c) maximal cable tensions, (d) maximal cable tension derivatives, (e) RMS of cable tension derivatives, and (f) consumed energy.*

wrench $\mathbf{f}$ would then be unfeasible. In order to not exceed the limit τ_{max}, the strategy described in Eq. (3.34) would be necessary and the resulting wrench would not be equal to $\mathbf{f}$. However, the MPC strategy does not lead to this risk. The PID+ controller has $|\tau_{max}| = 9.1$ kN. This relatively low tension is a consequence of the use of small gains $\mathbf{K}_p$. Indeed, small gains were used because larger gains lead to high frequency oscillations of cable tensions. For instance, increasing the gains of less than 1% with this strategy leads to RMS($\dot{\tau}$) equal to 86 kN.

The time derivative of the cable tensions is a measure of the degree of aggressivity of the control action. Large values of this variable may excite high frequency dynamics which are difficult to control. Figure 3.4-(d) presents the maximum derivative of cable tensions over the trajectory considering all cables. Mathematically, the values in Figure 3.4-(d) are equal to $\max_t \left(\max_i |\dot{\tau}_i(t)| \right)$. The smallest maximum cable tension derivative is obtained for the PID+, which is an advantage of this strategy. SMC is the most aggressive controller considering this performance measure. The proposed MPC strategy constraints this variable (according to (3.27)). Therefore, any value can be imposed independently of the rest of the controller parameters. In simulations, the value used is 800 kN/s and Figure 3.4-(d) shows that this value is reached. Figure 3.4-(e) presents the values of the maximum RMS value of $\dot{\tau}_i$ considering all cables $i \in \mathbb{N}_{1,m}$. More precisely, Figure 3.4 (e) presents values of $\max_i \left(\mathrm{RMS}(\dot{\tau}_i) \right)$. The MPC strategy presents the largest RMS of cable tension derivatives. Note that variables depending on the system states beyond the prediction horizon cannot be taken as constraints in

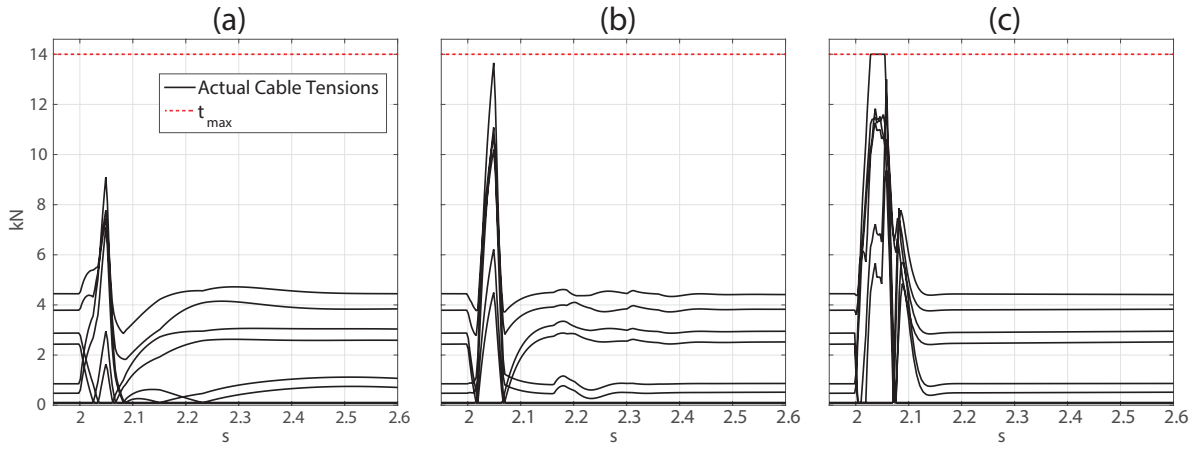

Figure 3.5 – *Simulated cable tensions for (a) PID+ Controller, (b) Sliding Mode Controller and (c) Model Predictive Control.*

the MPC optimization. The prediction horizon covers only a small part of the whole trajectory over which the RMS values are calculated.

The consumed energy over the trajectory is roughly the same for all the control schemes, *cf.* Figure3.4-(f). The consumed energy is calculated as $\int_{t_i}^{t_f} |\dot{\boldsymbol{l}}(t)^T \boldsymbol{\tau}(t)| dt$.

3.3 Real-time Experiments

As frequently concluded for MPC applications, the numerical solution of OCPs in real-time is a crucial challenge in the implementation on an experimental prototype. Moreover, since industrial software is used, this implementation could not take advantage of commonly used numerical libraries. Finally, the experiments unveil issues that may be neglected in simulations, *e.g.* the cable tension control and forward kinematics discussed in Chapter 2.

3.3.1 Experimental platform and implementation aspects

The control scheme introduced in Section 3.1 is experimented on the prototype HRPCable, a 6-DoF CDPR driven by 8 cables, installed in LIRMM facilities. Figure 1.1 shows this experimental setup and the corresponding CAD model. Details on the components of this CDPR are given in Table 1.1.

The MPC control scheme has been implemented in an industrial PC and developed in TwinCAT environment, using C++ language. As discussed before, TwinCAT is a software from Beckhoff commonly used in the industry, which turns PC-based systems into real-time controllers using Microsoft Windows kernel. In order to guarantee real-time performances, TwinCAT is not compatible with standard C++ libraries, including basic libraries such as `math.h`. This issue generally mitigates the application of MPC with constraints in industrial robotics. The control methods described in the

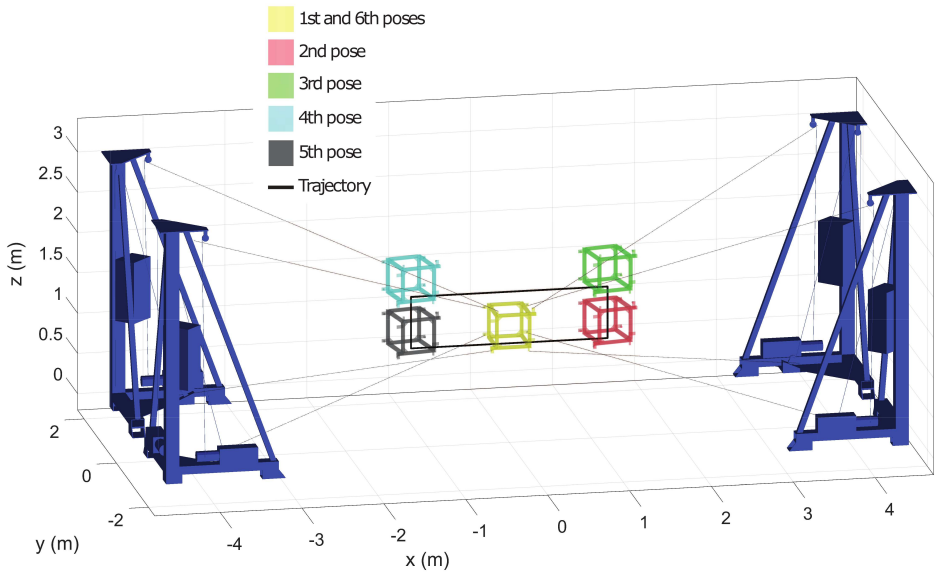

Figure 3.6 – *Illustration of a pick-and-place path in the CAD view of HRPCable.*

previous sections were thus programmed from scratch. Some notes on the numerical implementation are presented in Appendix A. The OCP formulation (3.25) is used.

In the experiments reported below, the control parameters are the following: $h_p = 6$, $h_c = 3$, $\mathbf{K}_u = 3.3 \times 10^{-3}\mathbf{I}$, $k_{\Delta u} = 0.3$. Matrix $\mathbf{K}_\gamma$ is diagonal and positive definite. Diagonal elements are repeated for each set of 12 elements (size of $\mathbf{y}$). These 12 elements are equal to

$$\begin{aligned} \mathbf{k}'_{\mathbf{y}} = & \begin{bmatrix} \mathbf{k}'^T_{\mathbf{x}} & \mathbf{k}'^T_{\dot{\mathbf{x}}} \end{bmatrix}^T = \\ = & \begin{bmatrix} 2\times10^7 & 2\times10^7 & 3.3\times10^7 & 2\times10^7 & \dots \\ 2\times10^7 & 2\times10^7 & 1.2\times10^{-3} & 0.6\times10^{-3} & \dots \\ 0.6\times10^{-3} & 0.6\times10^{-3} & 1.3\times10^{-3} & 0.6\times10^{-3} & \end{bmatrix}^T. \end{aligned} \tag{3.35}$$

These gains were tuned manually.

3.3.2 Experimental results

The present section experimentally evaluates the efficiency of the proposed MPC scheme with respect to the main goals considered in this thesis. Summarizing some of the key concerns presented in Section 1.2, the proposed method should combine satisfying positioning precision, disturbance rejection capabilities and should safely operate close to system constraints. Section 1.7 shows that this latter characteristic presents particular relevance in the context of this thesis. Indeed, state-of-the-art control strategies are not able to explicitly handle system constraints within the computation of the main feedback correction and, therefore, may be vulnerable to feasibility issues during the robot operation. The proposed MPC strategy is not prone to this issue since it handles the tension limits within the OCP formulation (3.25). In order to show the effectiveness of the proposed controller, Section 3.3.2.1 presents the results for a typical pick-and-place task performed with two different scenarios: $\boldsymbol{\tau}_{max} = 400$ N and

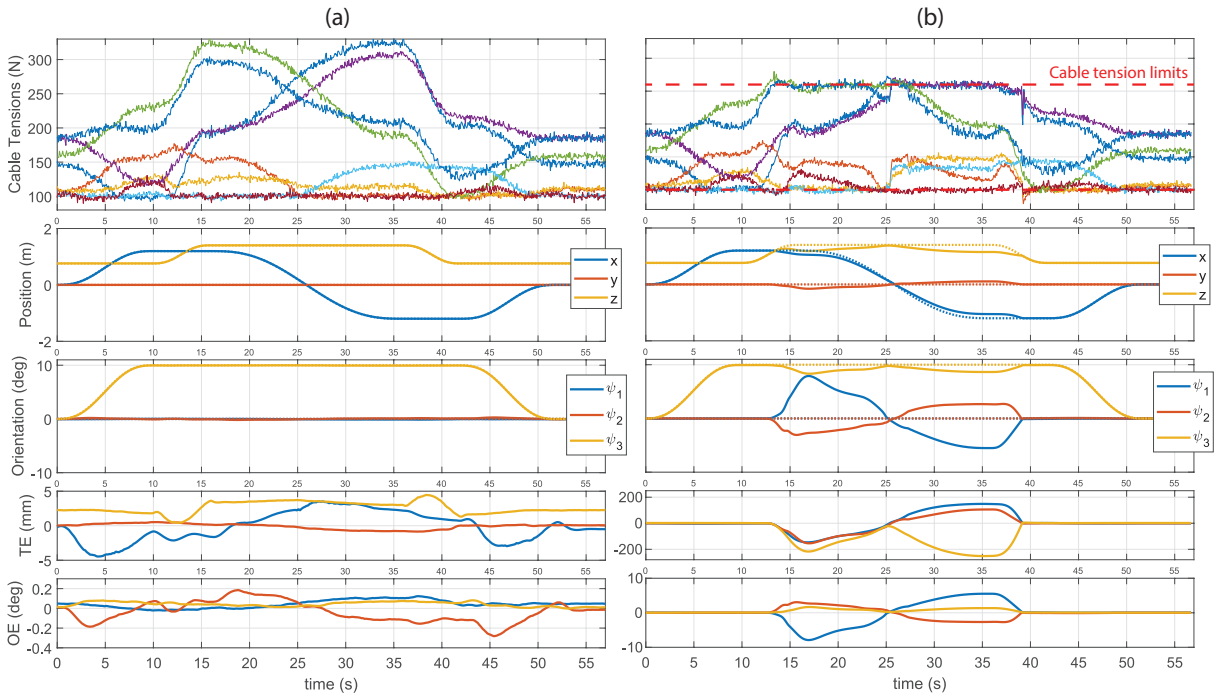

Figure 3.7 – *Pick-and-place task results with (a) $\tau_{max} = 400$ N and (b) $\boldsymbol{\tau}_{max} = 260$ N. The actual pose is shown in continuous lines and the desired one in dashed lines. Note that the axes limits for TE and OE are different for (a) and (b).*

$\boldsymbol{\tau}_{max} = 260$ N. The same cable tension limits were assumed for all the cables. The desired trajectory escapes from the robot workspace when the reduced maximum tension $\boldsymbol{\tau}_{max} = 260$ N is applied. The proposed MPC is able to comply with this reduced maximum cable tension while minimizing the Cartesian error between the actual and the desired trajectories. This is an important result related to the safety of the operation of CDPRs. Moreover, Section 3.3.2.2 evaluates the capabilities of the proposed MPC related to robustness against uncertainties.

3.3.2.1 Pick-and-Place task

The first scenario considered is a typical pick-and-place task. Figure 3.6 shows the sequence of the six desired platform poses defining the pick-and-place path. The trajectory between each subsequent pair of desired poses is defined with a 5^{th} degree polynomial function. In the following, the results obtained with two different scenarios of cable tension constraints are compared.

Nominal constraints: Figure 3.7-(a) shows the experimental results obtained with $\boldsymbol{\tau}_{max} = 400$ N and $\boldsymbol{\tau}_{min} = 100$ N. The proposed controller is able to keep the translation errors (TE) smaller than 4.5 mm and the orientation errors (OE) smaller than $0.3°$.

Reduced maximum tension: Figure 3.7-(b) shows the experimental results obtained with $\boldsymbol{\tau}_{max} = 260$ N and $\boldsymbol{\tau}_{min} = 100$ N. The proposed MPC finds poses as close as

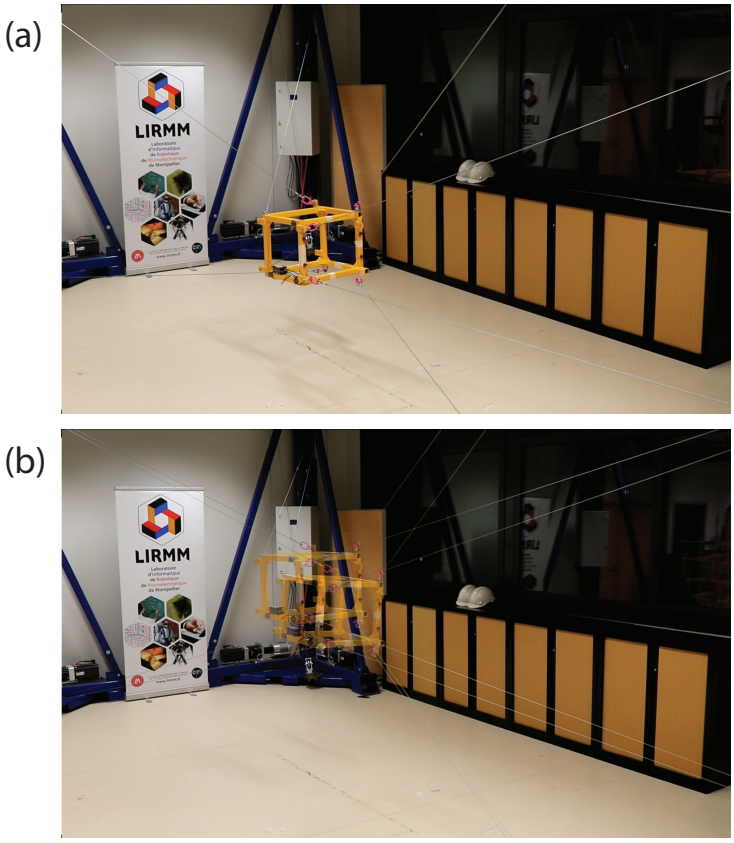

Figure 3.8 – *Overlaid views of both scenarios (nominal constraints and reduced maximum tension). The upper view (a) depicts an instant in which the desired pose is feasible for both cases. Conversely, the lower view (b) depicts an instant in which the desired pose is feasible for* $\tau_{max} = 400$ *N and unfeasible for* $\tau_{max} = 260$ *N.*

possible to desired poses complying with these cable tension limits. The translation errors reach values greater than 250 mm and orientation errors are close to $8°$.

Figure 3.8 depicts overlaid views comparing the scenarios with nominal and reduced maximum tension. This comparison is done for two different instants. The view in Figure 3.8-(a) was taken when the desired pose was feasible for both scenarios. In contrast, Figure 3.8-(b) depicts an instant in which the desired was feasible for the nominal case and unfeasible for the reduced maximum cable tension. The shift between desired and actual poses is evident in this case.

Strictly complying with the cable tension limits while minimizing the Cartesian errors in following the desired trajectory is the main advantage of the proposed controller. The control strategies discussed in Section 1.4 combined with a tension distribution schemes described in Section 1.5 would fail to fulfill this objective since the main controller would demand an unfeasible wrench. Figure 3.9 depicts the results obtained using the control scheme proposed in [8]. The robot operation is suddenly stopped when the desired trajectory reaches unfeasible poses, exemplifying the risks taken with such control strategy when operating close to system constraints.

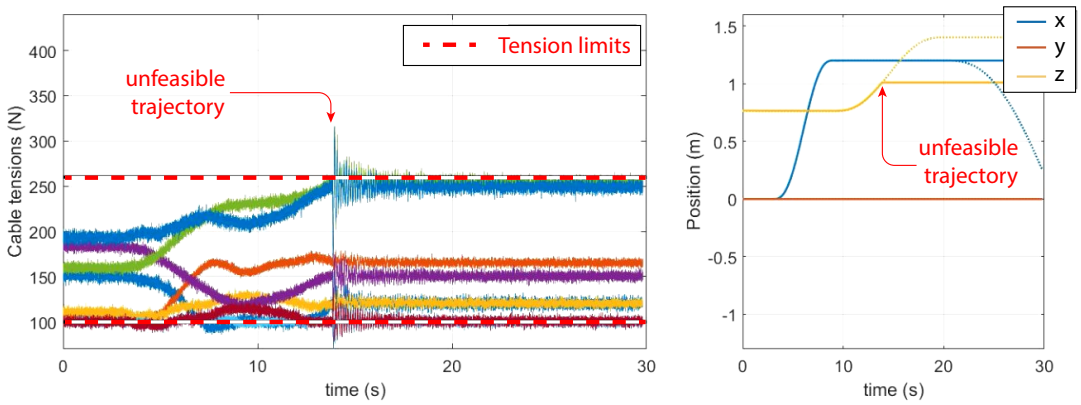

Figure 3.9 – *Experimental results obtained with the control scheme proposed in [8] for unfeasible desired an trajectory.*

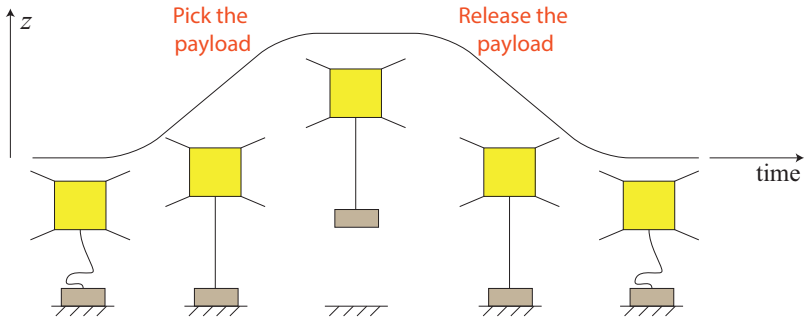

Figure 3.10 – *Illustration of the robustness test.*

3.3.2.2 Robustness against payload uncertainties

In many applications of CDPRs, the platform should pick weights from one position and release them in another position. In order to validate the applicability of the proposed control strategy in such tasks, it is important to evaluate its robustness against uncertainties on the lifted mass. To this end, the experiment illustrated in Figure 3.10 is proposed. The results are presented in Figure 3.11. An additional mass of 11.5 kg was used. This mass represents 50% of the platform mass m_p. The controller is able to keep the tracking errors smaller than 6 mm and 0.14°, despite this significant uncertainty.

3.3.3 Conclusions

The experimental results presented in this section show that the proposed control strategy is able to perform a trajectory keeping reduced errors while complying with cable tension limits. The real-time control was implemented in an industrial software and hardware environment to enable applicability of the proposed scheme in industry.

The proposed MPC scheme is able to address the cable tension limits explicitly, integrating the redundancy resolution within the main controller. As a result, cable tension limits are not violated even for reduced maximum cable tensions. Indeed, when a desired pose cannot be reached with some given tension limits, the proposed MPC is

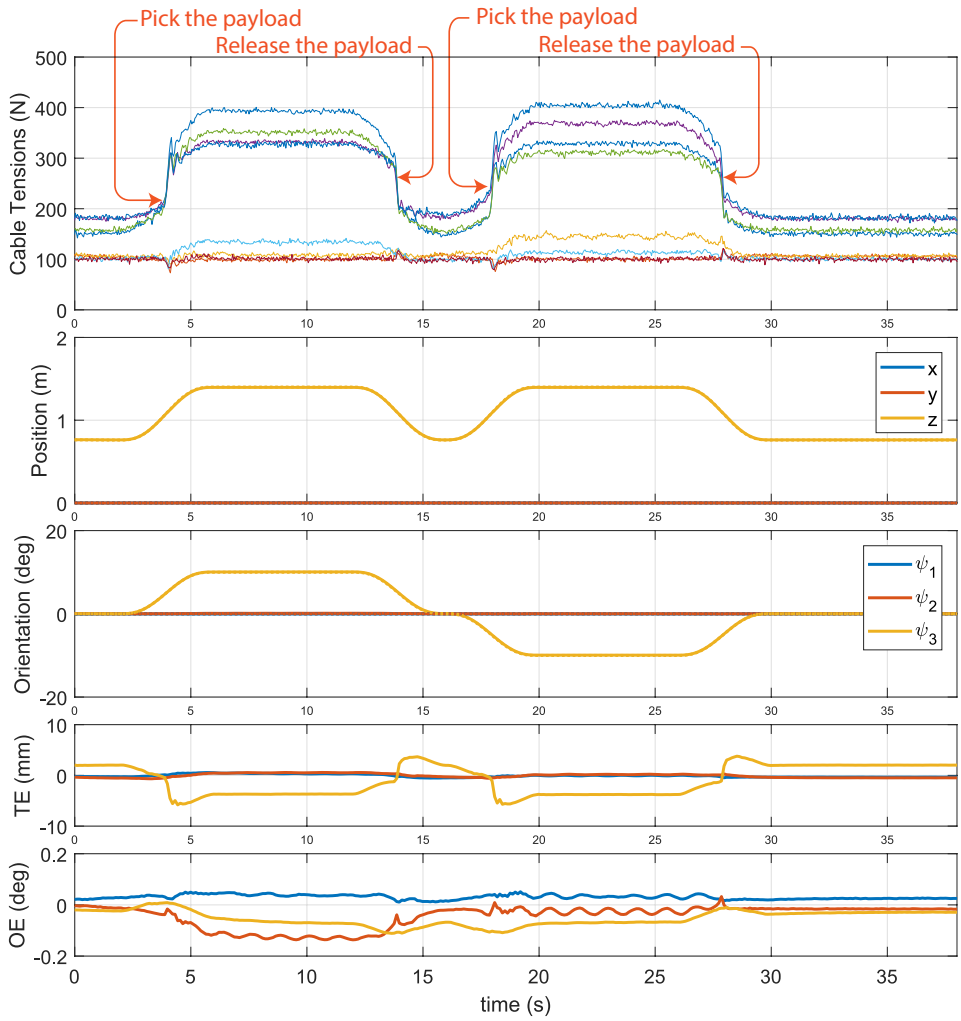

Figure 3.11 – *Robustness test results with an additional payload.*

able to find a trajectory as close as possible to the desired one while strictly respecting the tension limits. The obtained experimental results also show its robustness against payload uncertainties.

Chapter 1 showed that state-of-the-art control schemes lead to safety issues when operating close to system constraints. The results presented in this section demonstrate that the proposed linear MPC scheme overcome this issue. Therefore, the implemented control algorithm fulfills the main purpose for which it was conceived.

Whereas the necessity to safely operate close to system constraints is a specific requirement in the Hephaestus project, this characteristic is also favorable for the design of CDPRs. Considering the limitations of the existing state-of-the-art schemes, the designer must oversize the margin dedicated to model uncertainties in order to guarantee the safety of the robot operation. In contrast, a more efficient design may be used for a CDPR operating with the proposed MPC.

Nevertheless, the performance of this linear MPC may be deteriorated with the incidence of significant nonlinearities. More precisely, the vector γ obtained with the affine system (3.17) may be a bad approximation of the future states if (3.15) is not true.

In order to extend the results obtained in this chapter to system with significant nonlinearities, an NMPC scheme should be developed. The next chapter addresses this subject.

4

Nonlinear Model Predictive Control

The linear MPC presented in the previous chapter is based on a linear approximation of the CDPR dynamic model. As a result, it may present deteriorated performance in presence of significant nonlinearities. Accordingly, a nonlinear MPC is proposed in this chapter. After a preliminary discussion presented in Section 4.1, the overall NMPC algorithm and its optimal control problem are formulated in the Section 4.2. Since this strategy is based on the original nonlinear dynamic system, the stability of the corresponding closed-loop may be analyzed. This section also highlights additional aspects of the linear MPC proposed in the previous chapter that differ from the standard form of tracking MPC schemes. Notably, in order to attain this standard form, an alternative method aiming to the minimization of the cable tensions is proposed. Section 4.3 discusses issues related to the numerical discretization of the original continuous-time system and proposes numerical solutions for the optimization problems proposed in Section 4.2. The NMPC strategy is validated by numerical simulation results presented in Section 4.4.

Contents

4.1 Nonlinear Model Predictive Control

The present section outlines the main objectives and the rationale of the NMPC proposed in this thesis. Before addressing the problem itself, some definitions and notations are introduced in the following.

4.1.1 Definitions and notations

In addition to the previously used notations, some additional sets should be introduced

$$
\begin{aligned}
\mathbb{R}_+ &:= \{r \in \mathbb{R} \mid r > 0\};\\
\mathbb{R}_+^0 &:= \{r \in \mathbb{R} \mid r \geqslant 0\};\\
\mathbb{R}_+^{n_v} &:= \{\mathbf{v} \in \mathbb{R}^{n_v} \mid v_i > 0 \; \forall \, i \in \mathbb{N}_{1,n_v}\};\\
\mathbb{R}_{a,b} &:= \{r \in \mathbb{R} \mid a \leqslant r \leqslant b\};\\
\mathbb{U} &:= \{\boldsymbol{\tau} \in \mathbb{R}^m \mid \boldsymbol{\tau}_{min} \leqslant \boldsymbol{\tau} \leqslant \boldsymbol{\tau}_{max}\} \text{ (set of feasible cable tensions)};\\
\mathbb{U}^{n_u} &:= \{\boldsymbol{\tau}(\cdot) \in \mathbb{S}_{n_u}^m \mid \boldsymbol{\tau}_{min} \leqslant \boldsymbol{\tau}_i \leqslant \boldsymbol{\tau}_{max} \; \forall \, i \in \mathbb{N}_{1,n_u}\};\\
\mathbb{Y} &:= \left\{\mathbf{y} \in \mathbb{R}^{2n} \;\middle|\; \mathbf{y} = \begin{bmatrix}\mathbf{x}^T & \dot{\mathbf{x}}^T\end{bmatrix}^T \text{ with } \mathbf{x} \text{ and } \dot{\mathbf{x}} \text{ admissible}\right\};\\
\mathbb{X} &:= \left\{\mathbf{x} \in \mathbb{R}^n \;\middle|\; \exists \, \dot{\mathbf{x}} \in \mathbb{R}^n, \; \begin{bmatrix}\mathbf{x}^T & \dot{\mathbf{x}}^T\end{bmatrix}^T \in \mathbb{Y}\right\}.
\end{aligned}
$$

The set $\mathbb{Y}$ should restrict the set of poses and velocities within which the control is able to operate. This definition should use the tools related to the computation of the workspace of CDPRs, *e.g.* [28, 119, 184, 185]. More details on this matter are discussed in Sections 4.2 and 4.4.

The set $\mathbb{X}$ is the projection of $\mathbb{Y}$ in the space of platform poses. This set represents the admissible poses (poses $\mathbf{x} \in \mathbb{R}^n$ for which there exists a velocity vector $\dot{\mathbf{x}} \in \mathbb{R}^n$ such that $\begin{bmatrix}\mathbf{x}^T & \dot{\mathbf{x}}^T\end{bmatrix}^T \in \mathbb{Y}$).

Additionally, some comparison functions are used in this chapter. In accordance with the commonly used notations [13, 62], the following classes of functions are considered

$$
\begin{aligned}
\mathcal{K} &:= \{\alpha \in \mathbb{R}_0^+ \to \mathbb{R}_0^+ \mid \alpha \text{ continuous, strictly increasing and } \alpha(0) = 0\};\\
\mathcal{K}_\infty &:= \{\alpha \in \mathcal{K} \mid \alpha \text{ unbounded }\};\\
\mathcal{L} &:= \{\delta \in \mathbb{R}_0^+ \to \mathbb{R}_0^+ \mid \delta \text{ continuous, strictly decreasing and } \lim_{t\to\infty} \delta(t) = 0\};\\
\mathcal{KL} &:= \{\beta \in \mathbb{R}_0^+ \times \mathbb{R}_0^+ \to \mathbb{R}_0^+ \mid \beta(r,\cdot) \in \mathcal{L} \text{ and } \beta(\cdot,t) \in \mathcal{K}\}.
\end{aligned}
\tag{4.1}
$$

As in Chapter 3, any vectorial function representing the cable tensions in time $\boldsymbol{\tau}_c : \mathbb{R} \to \mathbb{R}^m$ is considered piece-wise constant. Accordingly, a sequence of cable tensions $\boldsymbol{\tau}(\cdot) \in \mathbb{S}^m$ is related to $\boldsymbol{\tau}_c$ by

$$
\boldsymbol{\tau}(k) = \boldsymbol{\tau}_k = \boldsymbol{\tau}_c(t), \;\; \text{for } t \in \left\{t \;\middle|\; (k-1)\Delta t \leqslant t < k\,\Delta t\right\} \text{ and } k \in \mathbb{N}, \tag{4.2}
$$

for a given sampling period Δt. The initial time is considered $t_0 = 0$. Note that, as in Chapter 3, a sequence of vector is denoted with bold letters followed by $(\cdot)$, such that the k^{th} vector of a sequence $\mathbf{s}(\cdot) \in \mathbb{S}^{n_s}$ is denoted by $\mathbf{s}(k) = \mathbf{s}_k$.

Moreover, the euclidean distance between two vectors $\mathbf{v}_1,\ \mathbf{v}_2 \in \mathbb{R}^{n_v}$, for any $n_v \in \mathbb{N}$, is denoted as $\|\mathbf{v}_1\|_{\mathbf{v}_2} = \|\mathbf{v}_2\|_{\mathbf{v}_1} = \sqrt{(\mathbf{v}_1 - \mathbf{v}_2)^T(\mathbf{v}_1 - \mathbf{v}_2)}$.

The reader is invited to refer to the Glossary at the end of this manuscript, which is a summary of the used notations.

4.1.2 Problem formulation

Since the cost functional (3.24) considers a linear approximation (3.17) of the continuous model (3.10), the satisfying results presented in Section 3.2 and 3.3 are obtained only for dynamic systems with limited nonlinearities. Therefore, a rigorous stability analysis of a general CDPR nonlinear dynamic model and its closed-loop is hindered. In this context, the goal of the present chapter is to obtain an NMPC scheme based on the original nonlinear model leading to satisfying performance and stability independently of the influence of nonlinearities.

Thanks to the simplicity resulting from the linear model, several issues in the design and implementation of a general NMPC could be addressed in Chapter 3 without a clear separation between themselves. Due to the increased complexity involved in the NMPC controller proposed in the present chapter, it is important to decouple these issues and address them separately. Accordingly, the design and implementation of an NMPC scheme may be divided into four aspects, namely:

(i) **Numerical discretization:** Definition of a transition-mapping representing a discrete-time approximation of the continuous-time state space model;

(ii) **Optimal Control Problem (OCP) formulation**: Definition of the cost functional to be minimized, as well as the control input and state constraints;

(iii) **Formulation of the OCP as a Nonlinear Programming (NLP) problem**: In general, the OCP obtained in (ii) is not formulated as a standard NLP. Typically, the transition mapping defined in (i) is not explicitly considered in the OCP. Therefore, (i) and (ii) should be integrated, leading to a standard NLP problem;

(iv) **Numerical solution of the NLP**: The output of the controller is obtained by solving numerically the NLP formulated in (iii). Interior-point and Sequential Quadratic Programming (SQP) methods are numerical algorithms commonly used to this end.

Numerical discretization is necessary since most of NMPC schemes assume discrete-time systems and the CDPR dynamic model (3.10) is continuous in time. More in detail, consider constants $t_a \in \mathbb{R}$, $\mathbf{x}_a,\ \dot{\mathbf{x}}_a \in \mathbb{R}^n$, $\boldsymbol{\tau}_a \in \mathbb{R}^m$ representing the actual time, pose, velocity and cable tensions, respectively. Considering that the cable tensions $\boldsymbol{\tau}_a$ are kept constant during a controller cycle period Δt, the solutions $\boldsymbol{x} : \mathbb{R} \to \mathbb{R}^n$ and $\dot{\boldsymbol{x}} : \mathbb{R} \to \mathbb{R}^n$

of the Initial Value Problem (IVP)

$$\ddot{\boldsymbol{x}} = \mathbf{M}(\boldsymbol{x})^{-1}\Big(\mathbf{g}(\boldsymbol{x}) + \mathbf{W}(\boldsymbol{x})\,\boldsymbol{\tau}_a - \mathbf{C}(\boldsymbol{x}, \dot{\boldsymbol{x}})\,\dot{\boldsymbol{x}}\Big), \tag{4.3a}$$

$$\boldsymbol{x}(t_a) = \mathbf{x}_a \tag{4.3b}$$

$$\dot{\boldsymbol{x}}(t_a) = \dot{\mathbf{x}}_a, \tag{4.3c}$$

lead to a transition mapping $\boldsymbol{\phi}_{\mathbf{x}} : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ such that

$$\begin{bmatrix} \boldsymbol{x}(t_a + \Delta t) \\ \dot{\boldsymbol{x}}(t_a + \Delta t) \end{bmatrix} = \boldsymbol{\phi}_{\mathbf{x}}\Big(\mathbf{x}_a, \dot{\mathbf{x}}_a, \boldsymbol{\tau}_a\Big). \tag{4.4}$$

Using a more compact notation, the vector of the next pose and velocity $\mathbf{y}^+ = \begin{bmatrix} \boldsymbol{x}(t_a + \Delta t)^T & \dot{\boldsymbol{x}}(t_a + \Delta t)^T \end{bmatrix}^T$ obtained with initial conditions $\begin{bmatrix} \boldsymbol{x}(t_a)^T & \dot{\boldsymbol{x}}(t_a)^T \end{bmatrix}^T = \mathbf{y} = \begin{bmatrix} \mathbf{x}_a^T & \dot{\mathbf{x}}_a^T \end{bmatrix}^T$ and application of constant cable tensions $\boldsymbol{\tau} \in \mathbb{R}^m$ is described with the transition mapping $\boldsymbol{\phi}_{\mathbf{y}} : \mathbb{R}^{2n} \times \mathbb{R}^m \to \mathbb{R}^{2n}$ according to

$$\mathbf{y}^+ = \boldsymbol{\phi}_{\mathbf{y}}(\mathbf{y}, \boldsymbol{\tau}). \tag{4.5}$$

Typically, the OCP formulation of an NMPC scheme and the subsequent stability analysis consider that the transition mapping representing the dynamic system is known. Accordingly, Section 4.2 considers that the expression for $\boldsymbol{\phi}_{\mathbf{y}}$ is known. The definition of the numerical discretization method is closely related to the definition of the NLP corresponding to an OCP. A deeper discussion on this matter is postponed to the Section 4.3. This section also discusses the items (iii) and (iv). Before that, the reader should be introduced to the proposed NMPC algorithm.

Section 4.2 introduces an NMPC taking the same state vector as the one introduced in Chapter 3. This state vector includes the CDPR platform pose and velocity. Since this set constitutes the minimal set of states with which the dynamic system may be modeled with a state space formulation, the corresponding NMPC scheme is called *minimal state* NMPC.

4.2 Minimal State NMPC

The present section introduces a minimal state NMPC scheme and is organized as follows: Section 4.2.1 details the control algorithm, Section 4.2.2 introduces the optimal control problem. The stability of the proposed scheme is discussed in the Section 4.2.3.

4.2.1 NMPC algorithm

The operation of the NMPC scheme proposed in this chapter may be presented with a block diagram equivalent to the used in Chapter 3 (*i.e.* Figure 3.1). The difference here consists in the MPC formulation which is present in the corresponding block

in Figure 3.1. Instead of a linear MPC, the present chapter proposes an NMPC scheme. In accordance with the definition of receding horizon MPC schemes discussed in Section 1.7.2, an optimal control problem is solved every controller cycle such that a cost functional is minimized satisfying system constraints.

As for the control scheme proposed in Chapter 3, no terminal condition is used (in accordance with the reasons discussed in Section 1.7). Accordingly, the cost functional (denoted by $J_{h_p} : \mathbb{N} \times \mathbb{Y} \times \mathbb{S}^m_{h_p} \to \mathbb{R}$) consists of a sum of individual costs associated to each sampling time within the prediction horizon $h_p \in \mathbb{N}$. The function responsible to compute these individual costs is called *stage cost* function and is denoted by $\ell : \mathbb{N} \times \mathbb{Y} \times \mathbb{U} \to \mathbb{R}$. Therefore, for an instant $k \in \mathbb{N}$, actual state $\mathbf{y}_a \in \mathbb{Y}$ and sequence of cable tensions $\boldsymbol{\tau}(\cdot) \in \mathbb{S}^m_{h_p}$, the cost functional is given by

$$J_{h_p}(k, \mathbf{y}_a, \boldsymbol{\tau}(\cdot)) = \sum_{j=1}^{h_p} \ell\left(k, \mathbf{y}_{\boldsymbol{\tau}(\cdot)}(j, \mathbf{y}_a), \boldsymbol{\tau}_j\right), \tag{4.6}$$

where the state $\mathbf{y}_{\boldsymbol{\tau}(\cdot)}(j, \mathbf{y}_a)$ represents the j^{th} term of the sequence of states $\mathbf{y}_{pa}(\cdot) \in \mathbb{S}^{2n}_{h_p}$ resulting from the application of the cable tensions $\boldsymbol{\tau}(\cdot)$ taking as initial state $\mathbf{y}_a$. This sequence of states is obtained applying recursively the transition mapping (4.5) j times, such that

$$\begin{aligned}
\mathbf{y}_{pa}(1) &= \mathbf{y}_{\boldsymbol{\tau}(\cdot)}(1, \mathbf{y}_a) = \boldsymbol{\phi}_{\mathbf{y}}(\mathbf{y}_a, \boldsymbol{\tau}_1) \\
\mathbf{y}_{pa}(2) &= \mathbf{y}_{\boldsymbol{\tau}(\cdot)}(2, \mathbf{y}_a) = \boldsymbol{\phi}_{\mathbf{y}}(\mathbf{y}_{pa}(1), \boldsymbol{\tau}_2) \\
\mathbf{y}_{pa}(3) &= \mathbf{y}_{\boldsymbol{\tau}(\cdot)}(3, \mathbf{y}_a) = \boldsymbol{\phi}_{\mathbf{y}}(\mathbf{y}_{pa}(2), \boldsymbol{\tau}_3) \\
&\vdots
\end{aligned} \tag{4.7}$$

The implemented NMPC algorithm is in accordance with the standard formulation of NMPC schemes without terminal conditions. The Algorithm 1 recalls the common implementation of NMPC schemes with the notations used in this chapter. This algorithm is also used to define the control policy $\boldsymbol{\tau}_{fb} : \mathbb{N} \times \mathbb{Y} \to \mathbb{U}$, which computes the vector of desired cable tensions $\boldsymbol{\tau}_{fb}(k, \mathbf{y}_a)$ for given instant $k \in \mathbb{N}$ and actual state $\mathbf{y}_a \in \mathbb{Y}$.

Considering a stage cost $\ell : \mathbb{N} \times \mathbb{Y} \times \mathbb{U}$ (which will be defined in Section 4.2.2), this vector is computed based on the solution of the OCP (4.8). Algorithm 1 integrates this control policy within the feedback loop such that the OCP (4.8) is solved for each controller cycle with updated estimations of the states. Accordingly, the NMPC scheme defines a set of desired cable tensions $\boldsymbol{\tau}_d(t)$ to be applied in the time interval $t \in [k\,\Delta t,\ (k+1)\,\Delta t)$. These outputs are used as setpoints for a cable tension control scheme.

The estimation of the states $\mathbf{y}_k$ in step 3 is performed with the tools introduced in Chapter 2. The detailed definition of the OCP (4.8) (mainly the considered stage cost) is presented in the Section 4.2.2.

Algorithm 1 NMPC Algorithm

Main control loop

1: Set $k \leftarrow 0$;
2: **loop**
3: Estimate the actual states $\mathbf{y}_k$;
4: Set desired cable tensions $\boldsymbol{\tau}_d(t) \leftarrow \boldsymbol{\tau}_{fb}(k, \mathbf{y}_k)$ for $t \in [k\,\Delta t,\ (k+1)\,\Delta t)$;
5: Set $k \leftarrow k + 1$;
6: **end loop**

Control policy

Inputs: $\mathbf{y}_a \in \mathbb{Y}$ and $k \in \mathbb{N}$;
Output: $\boldsymbol{\tau}_{fb}^v \in \mathbb{U}$;

1: **function** $\boldsymbol{\tau}_{fb}(k, \mathbf{y}_a)$
2: Solve the OCP

$$\begin{aligned} \boldsymbol{\tau}^* \leftarrow \underset{\boldsymbol{\tau}(\cdot)\in\mathbb{U}^{h_p}}{\arg\min} \ & \sum_{j=1}^{h_p} \ell\Big(j+k, \mathbf{y}_{\boldsymbol{\tau}(\cdot)}(j, \mathbf{y}_a), \boldsymbol{\tau}_j\Big) \\ \text{s.t.} \ & \mathbf{y}_{\boldsymbol{\tau}(\cdot)}(j, \mathbf{y}_a) \in \mathbb{Y} \ \forall\, j \in \mathbb{N}_{1,h_p} \end{aligned} \tag{4.8}$$

3: Set $\boldsymbol{\tau}_{fb}^v \leftarrow \boldsymbol{\tau}_1^*$;
4: **return** $\boldsymbol{\tau}_{fb}^v$.
5: **end function**

4.2.2 OCP formulation

The definition of a pertinent stage cost is a crucial step for the design of a stable and effective NMPC scheme. In order to meet the theoretical requirements necessary to guarantee stability and satisfying performance of an NMPC scheme, one should seek some particular properties of the stage cost. Postponing the formal discussion on this matter to Section 4.2.3, some key properties may be sketched to explain the rationale used in the design of the proposed OCP. The next paragraphs are dedicated to this end.

Typically, NMPC schemes penalize states and control inputs with respect to the *desired behavior of the system*. In order to better formulate this desired behavior, the following assumption defines the sequence of desired cable tensions and presents the conditions under which they are feasible.

Assumption 4.1. *Consider the sequences $\mathbf{x}_d(\cdot)$, $\dot{\mathbf{x}}_d(\cdot) \in \mathbb{S}^n$ of desired poses and velocities, respectively, and $\mathbf{y}_d(\cdot) \in \mathbb{S}^{2n}$ such that $\mathbf{y}_d(k) = \begin{bmatrix}\mathbf{x}_d(k)^T & \dot{\mathbf{x}}_d(k)^T\end{bmatrix}^T$ represents the desired trajectory. The sequence $\mathbf{y}_d(\cdot)$ is considered feasible, in the sense that there exists a sequence of desired cable tensions $\boldsymbol{\tau}'_d(\cdot) \in \mathbb{S}^m$ such that, for $\mathbf{y}_0 = \mathbf{y}_d(1)$,*

$$\mathbf{y}_{\boldsymbol{\tau}'_d(\cdot)}(k, \mathbf{y}_0) = \mathbf{y}_d(k) \in \mathbb{Y} \tag{4.9a}$$

$$\boldsymbol{\tau}'_d(k) \in \mathbb{U} \tag{4.9b}$$

for all $k \in \mathbb{N}$*. Moreover, each vector of* $\boldsymbol{\tau}'_d(\cdot)$ *is considered to have minimal 2-norm such that, for every* $k \in \mathbb{N}$*,*

$$\nexists\, \boldsymbol{\tau} \in \mathbb{U} \mid \mathbf{W}\left(\mathbf{x}_d(k)\right) \boldsymbol{\tau}'_d(k) = \mathbf{W}\left(\mathbf{x}_d(k)\right) \boldsymbol{\tau} \text{ and } \|\boldsymbol{\tau}\| < \|\boldsymbol{\tau}'_d(k)\|. \tag{4.10}$$

The cost functional of a tracking MPC (or classical stabilizing MPC) scheme should be null for states and control inputs equal to the desired trajectory and desired control inputs, and strictly positive if the states are not equal to the desired ones. More precisely, the stage cost function should satisfy

$$\begin{cases} \ell(k, \mathbf{y}, \boldsymbol{\tau}) = 0 \text{ if } \mathbf{y} = \mathbf{y}_d(k) \text{ and } \boldsymbol{\tau} = \boldsymbol{\tau}'_d(k), \\ \ell(k, \mathbf{y}, \boldsymbol{\tau}) > 0 \text{ if } \mathbf{y} \neq \mathbf{y}_d(k). \end{cases} \tag{4.11}$$

NMPC schemes satisfying this property are called hereinafter in this thesis as *Standard Tracking NMPC* (ST-NMPC). As a matter of fact, the formulation proposed in Section 3.1 is not in accordance with the standard MPC form. Since the stage cost used in Chapter 3 presents a term proportional to the squared norm of the vector of cable tensions (the term (3.19)), which is always strictly positive due to the minimal cable tension bounds, the corresponding OCP does not have null cost for trajectories coinciding with the desired one. Therefore, this section proposes a different approach aiming to minimize the 2-norm of the cable tension vector while satisfying (4.11).

As for the desired states, ST-NMPC schemes commonly penalize a weighted norm of the difference between control inputs taken as argument of the stage cost and the desired control inputs. For the stage cost $\ell(k, \mathbf{y}, \boldsymbol{\tau})$ studied here, a weighted norm of $\boldsymbol{\tau} - \boldsymbol{\tau}'_d(k)$ could be used. Nevertheless, in order to obtain $\boldsymbol{\tau}'_d(\cdot)$, one would need to solve the tension distribution problem of the whole trajectory beforehand, using a TDA coupled to the inverse dynamics. Although this is a reasonable solution, the OCPs proposed in this thesis exempt the control scheme of this previous computation. To this end, the following stage cost is proposed

$$\ell(k, \mathbf{y}, \boldsymbol{\tau}) = \|\mathbf{y} - \mathbf{y}_d(k)\|^2_{\mathbf{K}_\mathbf{y}} + \|\boldsymbol{\tau} - \tilde{\boldsymbol{\tau}}_r(\mathbf{x}_\mathbf{y}, \boldsymbol{\tau})\|^2_{\mathbf{K}_{\boldsymbol{\tau}}}, \tag{4.12}$$

where $\mathbf{K}_\mathbf{y} = \mathrm{diag}(\mathbf{k}_\mathbf{y})$ and $\mathbf{K}_{\boldsymbol{\tau}} = \mathrm{diag}(\mathbf{k}_{\boldsymbol{\tau}})$, with constant vectors $\mathbf{k}_\mathbf{y} \in \mathbb{R}^{2n}_+$ and $\mathbf{k}_{\boldsymbol{\tau}} \in \mathbb{R}^m_+$. $\|(\cdot)\|_\mathbf{K}$ is the norm of $(\cdot)$ weighted with a symmetric positive definite matrix $\mathbf{K}$. The vector $\mathbf{x}_\mathbf{y}$ represents the pose vector obtained by extracting the first n elements of the $\mathbf{y}$. The vectorial function $\tilde{\boldsymbol{\tau}}_r : \mathbb{X} \times \mathbb{U} \to \mathbb{U}$ denotes the estimation of desired cable tensions, such that, for given $\mathbf{x} \in \mathbb{X}$ and $\boldsymbol{\tau}_{in} \in \mathbb{U}$,

$$\tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau}_{in}) = \arg\min_{\boldsymbol{\tau}'} \|\boldsymbol{\tau}'\|_2^2 \tag{4.13a}$$

$$\text{s.t. } \mathbf{W}(\mathbf{x})\, \boldsymbol{\tau}_{in} = \mathbf{W}(\mathbf{x})\, \boldsymbol{\tau}' \tag{4.13b}$$

$$\boldsymbol{\tau}_{min} \leqslant \boldsymbol{\tau}' \leqslant \boldsymbol{\tau}_{max} \tag{4.13c}$$

In words, $\tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau}_{in})$ is the vector of cable tensions with minimal 2-norm that is able to generate the same wrench than $\boldsymbol{\tau}_{in}$ at the pose $\mathbf{x}$. Therefore, for given $k \in \mathbb{N}$, $\mathbf{y} \in \mathbb{Y}$ and $\boldsymbol{\tau}_{in} \in \mathbb{U}$, the term $\|\boldsymbol{\tau} - \tilde{\boldsymbol{\tau}}_r(\mathbf{x}_\mathbf{y}, \boldsymbol{\tau}_{in})\|^2_{\mathbf{K}_{\boldsymbol{\tau}}} = 0$ iff $\boldsymbol{\tau}$ presents minimal 2-norm ($\boldsymbol{\tau} = \tilde{\boldsymbol{\tau}}_r(\mathbf{x}_\mathbf{y}, \boldsymbol{\tau}_{in})$). Otherwise, $\|\boldsymbol{\tau} - \tilde{\boldsymbol{\tau}}_r(\mathbf{x}_\mathbf{y}, \boldsymbol{\tau}_{in})\|^2_{\mathbf{K}_{\boldsymbol{\tau}}} > 0$, indicating that there exists a

$\tilde{\boldsymbol{\tau}}_r(\mathbf{x}_\mathbf{y}, \boldsymbol{\tau}_{in}) \in \mathbb{U}$ generating the same wrench (thanks to (4.13b)) with $\|\tilde{\boldsymbol{\tau}}_r(\mathbf{x}_\mathbf{y}, \boldsymbol{\tau}_{in})\|_2 < \|\boldsymbol{\tau}\|_2$.

The Algorithm 2 presented in Section 4.3 is able to solve the optimization problem (4.13). As discussed in Section 1.5, one crucial part of a TDA is to compute a *feasible* cable tension vector able to generate the desired wrench. Since $\boldsymbol{\tau}_{in} \in \mathbb{U}$, this vector is necessarily feasible and $\boldsymbol{\tau}_{min} \leqslant \boldsymbol{\tau}_{in} \leqslant \boldsymbol{\tau}_{max}$. Moreover, the desired wrench in (4.13b) is, in fact, the one generated by $\boldsymbol{\tau}_{in}$. Therefore, the cable vector of tensions $\boldsymbol{\tau}_{in}$ may be used as a feasible initial guess in an iterative optimization, which may be much simpler than a general TDA.

For a given actual state $\mathbf{y}_a$ at time instant k the proposed optimal control problem is summarized as follows

$$\begin{aligned} \min_{\boldsymbol{\tau}(\cdot)\in\mathbb{U}^{h_p}} \; & \sum_{j=1}^{h_p} \ell\Big(j+k, \mathbf{y}_{\boldsymbol{\tau}(\cdot)}\left(j, \mathbf{y}_a\right), \boldsymbol{\tau}_i\Big) \quad . \\ \text{s. t. } & \mathbf{y}_{\boldsymbol{\tau}(\cdot)}\Big(j, \mathbf{y}_a\Big) \in \mathbb{Y} \; \forall \; j \in \mathbb{N}_{1,h_p} \end{aligned} \tag{4.14}$$

Note that, as a consequence of (4.10) and (4.13),

$$\|\tilde{\boldsymbol{\tau}}_r(\mathbf{x}_d(k), \boldsymbol{\tau}'_d(k)) - \boldsymbol{\tau}'_d(k)\| = \ell\Big(k, \mathbf{y}_d(k), \boldsymbol{\tau}'_d(k)\Big) = 0 \; \forall \; k \in \mathbb{N} \tag{4.15}$$

and conditions (4.11) are satisfied with the stage cost (4.12). Thereby, the tension distribution is implicitly performed when solving the OCP (4.14).

4.2.3 Stability analysis

This section presents the stability analysis of the closed-loop system obtained with the NMPC proposed in Sections 4.2.1 and 4.2.2. More precisely, the main goal is to study the conditions under which the corresponding closed-loop system is *uniformly asymptotically stable* according to the following definition.

Definition 4.2. *Consider the NMPC Algorithm 1 with prediction horizon $h_p \in \mathbb{N}$ and a feasible sequence of desired states $\mathbf{y}_d(\cdot) \in \mathbb{S}^{2n}$. The resulting closed-loop system*

$$\mathbf{y}_{j+1} = \boldsymbol{\phi}_\mathbf{y}\left(\mathbf{y}_j, \boldsymbol{\tau}_{fb}(j, \mathbf{y}_j)\right), \; \forall \; j \in \mathbb{N} \tag{4.16}$$

is called uniformly asymptotically stable to $\mathbf{y}_d(\cdot)$ on $\mathbb{Y}$ if, for each $\mathbf{y}_a \in \mathbb{Y}$ and $k \in \mathbb{N}$, there exists a $\beta \in \mathcal{KL}$ such that $\mathbf{y}^{fb}(\cdot) \in \mathbb{S}^{2n}$ defined according

$$\begin{aligned} \mathbf{y}_1^{fb} &= \boldsymbol{\phi}_\mathbf{y}\left(\mathbf{y}_a, \boldsymbol{\tau}_{fb}(k, \mathbf{y}_a)\right) \\ \mathbf{y}_{j+1}^{fb} &= \boldsymbol{\phi}_\mathbf{y}\left(\mathbf{y}_j^{fb}, \boldsymbol{\tau}_{fb}(j+k, \mathbf{y}_j^{fb})\right), \; \forall \; j \in \mathbb{N} \end{aligned} \tag{4.17}$$

satisfies the following relation

$$\|\mathbf{y}_j^{fb}\|_{\mathbf{y}_d(k+j)} \leqslant \beta(\|\mathbf{y}_a\|_{\mathbf{y}_d(k)}, j), \; \forall \; j \in \mathbb{N}. \tag{4.18}$$

In words, (4.17) constructs the sequence of states $\mathbf{y}^{fb}(\cdot)$ that is obtained considering the closed-loop (4.16) "departing" from $\mathbf{y}_a$ at sampling time k. The closed-loop (4.16) takes as feedback control policy $\boldsymbol{\tau}_{fb}$ defined in Algorithm 1. The definition of uniform asymptotic stability presented above is based on the upper bound of the error $\|\mathbf{y}_j^{fb} - \mathbf{y}_d(k+j)\| = \|\mathbf{y}_j^{fb}\|_{\mathbf{y}_d(k+j)}$.

Note that, since the sequence $\mathbf{y}^{fb}(\cdot)$ starts shifted from k sampling periods from the initial time, the error between $\mathbf{y}^{fb}(\cdot)$ and $\mathbf{y}_d(\cdot)$ should be shifted in time as well. In other words, k represents the initial discrete time of the studied trajectory and j represents its evolution in time starting from sampling time k. The same notations are used in the remainder of this section for trajectories others than $\mathbf{y}^{fb}(\cdot)$.

The upper bound used to limit the tracking error is established in terms of the function $\beta \in \mathcal{KL}$. An illustrative example of a function belonging to this class is presented in Figure 4.1. In accordance to the definition presented in Section 4.1.1, the illustrative function $\beta_{ex}(r', t)$ tends to zero for $t \to \infty$ and is strictly decreasing with respect to the second argument (for a constant $r' \in \mathbb{R}_+$). Since the second argument in (4.18) represents the time, the tracking error asymptotically converges to zero. In contrast, β is strictly increasing with respect to the first argument. Therefore, for increased initial errors $\|\mathbf{y}_a\|_{\mathbf{y}_d(k)}$, the function of time $\beta(\|\mathbf{y}_a\|_{\mathbf{y}_d(k)}, \cdot) \in \mathcal{L}$ may be increased as well.

Definition 4.2 is used in [13, Definition 2.16] for the analysis of MPC schemes and is closely related to [62, Definition 4.2], which is more often used in general nonlinear control. As a matter of fact, the analysis presented in this section is based on [13] and the used notations are similar to those proposed in this reference.

Since the OCP (4.14) does not use stabilizing terminal conditions, tools designed to the analysis of NMPC schemes without terminal conditions should be used in the stability analysis of the closed-loop (4.16). The *minimal time-varying stage cost* denoted by $\ell^* : \mathbb{N} \times \mathbb{Y} \to \mathbb{R}$ is fundamental in this context. This function is defined as

$$\ell^*(k, \mathbf{y}) = \min_{\boldsymbol{\tau} \in \mathbb{U}} \ell(k, \mathbf{y}, \boldsymbol{\tau}). \tag{4.19}$$

The asymptotic stability of ST-NMPCs without terminal conditions is addressed in [13, Chapter 6]. Summarizing some of the results presented in this chapter, the following theorem can be stated.

Theorem 4.3. *Consider the NMPC Algorithm 1 with prediction horizon $h_p \in \mathbb{N}$ and minimal stage cost satisfying*

$$\alpha_1(\|\mathbf{y} - \mathbf{y}_d(k)\|) \leqslant \ell^*(k, \mathbf{y}) \leqslant \alpha_2(\|\mathbf{y} - \mathbf{y}_d(k)\|) \; \forall \, k \in \mathbb{N}, \mathbf{y} \in \mathbb{Y} \tag{4.20}$$

with suitable α_1, $\alpha_2 \in \mathcal{K}_\infty$. Suppose that, for all $\mathbf{y}_a \in \mathbb{Y}$, there exist a feasible $\boldsymbol{\tau}^e(\cdot) \in \mathbb{S}^m$, real $C < \infty$ and $\sigma \in (0, 1)$ satisfying

$$\ell\left(k + j, \mathbf{y}_{\boldsymbol{\tau}^e(\cdot)}(j, \mathbf{y}_a), \boldsymbol{\tau}_j^e\right) \leqslant C\, \sigma^j\, \ell^*\left(k, \mathbf{y}_a\right), \tag{4.21}$$

for all k, $j \in \mathbb{N}$. Then, the nominal closed-loop (4.16) is uniformly asymptotically stable on $\mathbb{Y}$ provided that h_p is sufficiently large. □

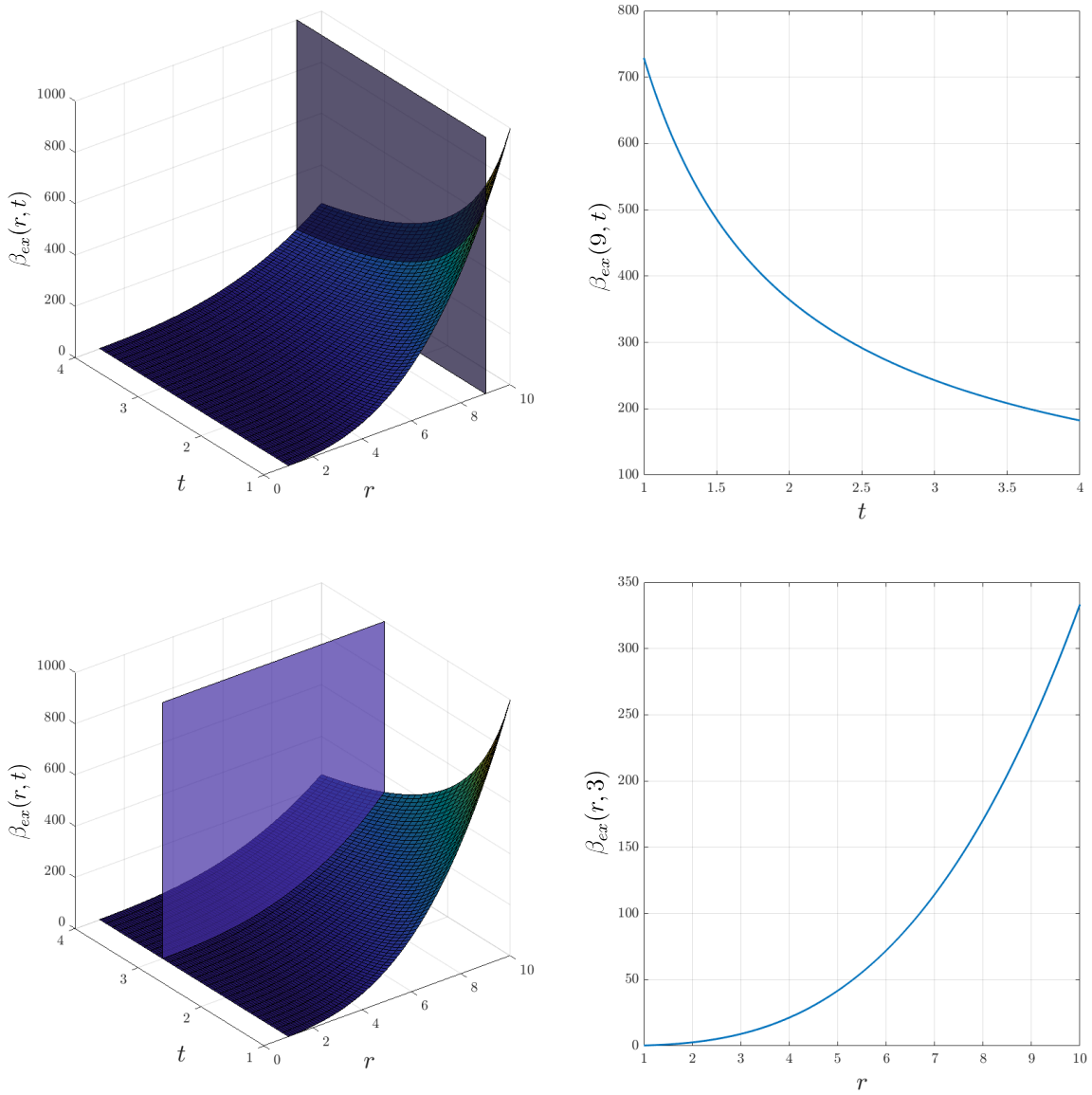

Figure 4.1 – *An illustrative example of a function* $\beta_{ex} \in \mathcal{KL}$.

Theorem 4.3 describes sufficient conditions to obtain a stable closed-loop system with the proposed NMPC for h_p sufficiently large. Inequalities (4.20) are closely related to (4.11). These conditions will be quite straightforwardly deduced at the end of this section.

Nevertheless, inequality (4.21) represents an important restriction to the properties of the stage cost. This inequality imposes, for each discrete instant $k \in \mathbb{N}$ and actual state $\mathbf{y}_a \in \mathbb{Y}$, the existence of a feasible sequence $\boldsymbol{\tau}^e(\cdot)$ that leads to a stage cost converging exponentially to zero in time. The term $\ell\left(k+j, \mathbf{y}_{\boldsymbol{\tau}^e(\cdot)}(j, \mathbf{y}_a), \boldsymbol{\tau}_j^e\right)$ represents this stage cost. The exponential convergence is a result of the term $C\,\sigma^j$ with $\sigma \in (0,\,1)$. Note that the upper bound function $C\,\sigma^j\,\ell^*\,(k, \mathbf{y}_a)$ is proportional to the initial minimal stage cost $\ell^*\,(k, \mathbf{y}_a)$.

It is important to highlight that Theorem 4.3 is based on the existence of *any* feasi-

ble sequence $\boldsymbol{\tau}^e(\cdot)$ that satisfies (4.21). The control policy $\boldsymbol{\tau}_{fb}$ is not considered explicitly in this theorem. In contrast to most of the control architectures, constrained MPC schemes generally do not have an analytical expression for the control policy. For this reason, the closed-loop cannot be studied explicitly. This is done by means of suitable properties of the stage cost. In this chapter, this suitable property is described by (4.21).

Since one of the terms in the stage cost (4.12) is the error $\|\mathbf{y} - \mathbf{y}_d\|^2_{\mathbf{K_y}}$, this error need to be bounded as well. Based on that, the following assumption establishes important properties of the controlled system in order to attain (4.21).

Assumption 4.4. *Consider the error $e_{\mathbf{y}} : \mathbb{N} \times \mathbb{Y} \to \mathbb{R}$ given by*

$$e_{\mathbf{y}}(k, \mathbf{y}) = \|\mathbf{y} - \mathbf{y}_d(k)\|^2_{\mathbf{K_y}}. \tag{4.22}$$

For each $\mathbf{y}_a \in \mathbb{Y}$ and $k \in \mathbb{N}$, there exist $\overline{\boldsymbol{\tau}}(\cdot) \in \mathbb{S}^m$, real $C_1, < \infty$ and $\sigma_1 < 1$ such that

$$e_{\mathbf{y}}\Big(k + j, \mathbf{y}_{\overline{\boldsymbol{\tau}}(\cdot)}(j, \mathbf{y}_a)\Big) \leqslant C_1\, \sigma_1^j\, e_{\mathbf{y}}(k, \mathbf{y}_a) \tag{4.23a}$$

$$\overline{\boldsymbol{\tau}}_j \in \mathbb{U} \tag{4.23b}$$

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

If Assumption 4.4 is true, then, for each state $\mathbf{y}_a \in \mathbb{Y}$ and instant $k \in \mathbb{N}$, there exists a feasible sequence of cable tensions $\overline{\boldsymbol{\tau}}(\cdot)$ that generates a trajectory in which the error $\|\mathbf{y}_{\overline{\boldsymbol{\tau}}(\cdot)}(k + j, \mathbf{y}_a) - \mathbf{y}_d(k + j)\|_{\mathbf{K_y}}$ exponentially converges to zero. Moreover, the upper bound function in (4.23a) is proportional to the initial errors $e_{\mathbf{y}}(k, \mathbf{y}_a)$. Note that, for given k and $\mathbf{y}_a$, the error $e_{\mathbf{y}}(k, \mathbf{y}_a)$ is a constant with respect to the discrete time j.

It is easy to show that this assumption may be satisfied with a typical control scheme. Consider, for instance, a continuous-time OSC scheme that applies the wrench

$$\mathbf{f} = \mathbf{M}(\boldsymbol{x})\Big(\ddot{\boldsymbol{x}}_d + \mathbf{K}_P(\boldsymbol{x}_d - \boldsymbol{x}) + \mathbf{K}_D(\dot{\boldsymbol{x}}_d - \boldsymbol{x})\Big) + \mathbf{C}(\boldsymbol{x}, \dot{\boldsymbol{x}})\, \dot{\boldsymbol{x}} - \mathbf{g}(\boldsymbol{x}) \tag{4.24}$$

with diagonal gain matrices $\mathbf{K}_P$ and $\mathbf{K}_D$. The nominal closed-loop system dynamics is given by

$$\ddot{\boldsymbol{x}} = \ddot{\boldsymbol{x}}_d + \mathbf{K}_P(\boldsymbol{x}_d - \boldsymbol{x}) + \mathbf{K}_D(\dot{\boldsymbol{x}}_d - \boldsymbol{x}) \tag{4.25}$$

or, written in terms of the error $\boldsymbol{e}(t) = \boldsymbol{x}_d(t) - \boldsymbol{x}(t)$,

$$\ddot{\boldsymbol{e}} = -\mathbf{K}_D\, \dot{\boldsymbol{e}} - \mathbf{K}_P\, \boldsymbol{e}. \tag{4.26}$$

Choosing, for instance, $\mathbf{K}_P = \mathbf{K}_D = \mathbf{I}$, the solution to the differential equation (4.26) can be written as

$$\boldsymbol{e}(t) = e^{-t/2}\, \left(\mathbf{a}_1 \sin(\omega\, t) + \mathbf{a}_2 \cos(\omega\, t)\right), \tag{4.27}$$

with $\omega = \sqrt{3}/2$ and constant vectors $\mathbf{a}_1,\ \mathbf{a}_2 \in \mathbb{R}^n$ given by

$$\begin{aligned} \mathbf{a}_1 &= \boldsymbol{e}_0 \\ \mathbf{a}_2 &= \frac{1}{\omega}\left(\dot{\boldsymbol{e}}_0 - \frac{1}{2}\boldsymbol{e}_0\right) \end{aligned} \tag{4.28}$$

where $\boldsymbol{e}_0 = \boldsymbol{e}(0)$, $\dot{\boldsymbol{e}}_0 = \dot{\boldsymbol{e}}(0)$. The expressions of the error $\boldsymbol{e}$ and its time derivative satisfy

$$\begin{aligned}
\|\boldsymbol{e}(t)\| &= \|e^{-t/2}(\mathbf{a}_1 \sin(\omega\, t) + \mathbf{a}_2 \cos(\omega\, t))\| &&\leqslant \left(e^{-1/2}\right)^t (\|\mathbf{a}_1\| + \|\mathbf{a}_2\|) \\
\|\dot{\boldsymbol{e}}(t)\| &= \left\| e^{-t/2} \left\{ \underbrace{\left(\frac{1}{2}\,\mathbf{a}_1 - \omega\,\mathbf{a}_2\right)}_{\mathbf{a}_3} \sin(\omega\, t) + \underbrace{\left(\frac{1}{2}\,\mathbf{a}_2 + \omega\,\mathbf{a}_1\right)}_{\mathbf{a}_4} \cos(\omega\, t) \right\} \right\| &&\leqslant \left(e^{-1/2}\right)^t (\|\mathbf{a}_3\| + \|\mathbf{a}_4\|)
\end{aligned}$$

A tedious but straightforward computation using (4.28) leads to the bound $\|\boldsymbol{e}(t)\|^2 + \|\dot{\boldsymbol{e}}(t)\|^2 \leqslant c_{ey}\, e^{-t}\, (\|\boldsymbol{e}_0\|^2 + \|\dot{\boldsymbol{e}}_0\|^2)$, with $c_{ey} = (16 + 2\sqrt{3})/3$.

Define $\mathbf{k}_{\mathbf{x}},\ \mathbf{k}_{\dot{\mathbf{x}}} \in \mathbb{R}^n$ subvectors of the previously defined $\mathbf{k}_{\mathbf{y}}$ such that $\mathbf{k}_{\mathbf{y}} = \begin{bmatrix} \mathbf{k}_{\mathbf{x}}^T & \mathbf{k}_{\dot{\mathbf{x}}}^T \end{bmatrix}^T$ and matrices $\mathbf{K}_{\mathbf{x}},\ \mathbf{K}_{\dot{\mathbf{x}}} \in \mathbb{R}^{n\times n}$ diagonal matrices with $\mathbf{K}_{\mathbf{x}} = \operatorname{diag}(\mathbf{k}_{\mathbf{x}})$, $\mathbf{K}_{\dot{\mathbf{x}}} = \operatorname{diag}(\mathbf{k}_{\dot{\mathbf{x}}})$. Denoting also

$$\begin{aligned}
K_{\mathbf{y}}^{max} &= \max_i \left(k_{\mathbf{y},i}\right), \\
K_{\mathbf{y}}^{min} &= \min_i \left(k_{\mathbf{y},i}\right),
\end{aligned} \tag{4.29}$$

the following inequalities hold

$$\begin{aligned}
\|\boldsymbol{e}(t)\|_{\mathbf{K}_{\mathbf{x}}}^2 + \|\dot{\boldsymbol{e}}(t)\|_{\mathbf{K}_{\dot{\mathbf{x}}}}^2 &\leqslant K_{\mathbf{y}}^{max}(\|\boldsymbol{e}(t)\|^2 + \|\dot{\boldsymbol{e}}(t)\|^2) \\
&\leqslant \left(e^{-1}\right)^t\, K_{\mathbf{y}}^{max}\, c_{ey}\, (\|\boldsymbol{e}_0\|^2 + \|\dot{\boldsymbol{e}}_0^2\|) \\
&\leqslant \left(e^{-1}\right)^t\, \frac{K_{\mathbf{y}}^{max}}{K_{\mathbf{y}}^{min}}\, c_{ey}\, (\|\boldsymbol{e}_0\|_{\mathbf{K}_{\mathbf{x}}}^2 + \|\dot{\boldsymbol{e}}_0\|_{\mathbf{K}_{\dot{\mathbf{x}}}}^2)
\end{aligned}$$

which is the continuous counterpart of (4.23a) with $C_1 = c_{ey}\, K_{\mathbf{y}}^{max}/K_{\mathbf{y}}^{min} < \infty$ and $\sigma_1 = e^{-1} < 1$.

As important as (4.23a) imposing the exponential convergence of error $e_{\mathbf{y}}(k,\cdot)$, (4.23b) requires that the sequence $\overline{\boldsymbol{\tau}}(\cdot)$ is feasible. Considering an OSC strategy, this means that the wrench defined in the feedback control policy is feasible for any pose and velocity within the state constraint set $\mathbb{Y}$. Clearly, the definition of the robot workspace and the set $\mathbb{Y} \subset \mathbb{R}^{2n}$ play crucial roles in Assumption 4.4. The study of the feasibility of a given wrench considering different poses and velocities is addressed in several works (*e.g.* [28, 119, 184, 185]) and it is out of the scope of this thesis. Moreover, it is important to highlight that assumption (4.23b) is often implicitly taken. Studies such as [26, 58, 71] analyze the corresponding closed-loop systems considering that the feedback loop does not lead to unfeasible cable tensions.

Since the term $e_{\mathbf{y}}(k,\mathbf{y}) = \|\mathbf{y} - \mathbf{y}_d(k)\|_{\mathbf{K}_{\mathbf{y}}}^2$ in the expression (4.12) of the stage cost $\ell(k,\mathbf{y},\boldsymbol{\tau})$ is independent of $\boldsymbol{\tau}$, the minimal time-varying stage cost is given by

$$\ell^*(k,\mathbf{y}) = e_{\mathbf{y}}(k,\mathbf{y}) + \min_{\boldsymbol{\tau}\in\mathbb{U}} (\|\boldsymbol{\tau} - \tilde{\boldsymbol{\tau}}_r(\mathbf{x}_{\mathbf{y}},\boldsymbol{\tau})\|_{\mathbf{K}_{\boldsymbol{\tau}}}^2). \tag{4.30}$$

Lemma 4.7 will show that the minimum $\min_{\boldsymbol{\tau}\in\mathbb{U}}(\|\boldsymbol{\tau} - \tilde{\boldsymbol{\tau}}_r(\mathbf{x}_{\mathbf{y}},\boldsymbol{\tau})\|_{\mathbf{K}_{\boldsymbol{\tau}}})$ is known and (4.30) can thus be simplified. Before stating and proving Lemma 4.7, a preparatory lemma is necessary.

Lemma 4.5. *Consider* $\boldsymbol{\tau}^a$, $\boldsymbol{\tau}^b \in \mathbb{U}$ *and* $\mathbf{x} \in \mathbb{X}$. *If*

$$\mathbf{W}(\mathbf{x})\,\boldsymbol{\tau}^a = \mathbf{W}(\mathbf{x})\,\boldsymbol{\tau}^b, \tag{4.31}$$

then,

$$\tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau}^a) = \tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau}^b). \tag{4.32}$$

Proof. First, the existence of a $\tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau}_{in})$ for all $\mathbf{x} \in \mathbb{R}^n$, $\boldsymbol{\tau}_{in} \in \mathbb{U}$ should be proved. The objective function $\|\boldsymbol{\tau}'\|_2^2$ in (4.13a) is quadratic and strictly convex. The substitution of $\boldsymbol{\tau}' = \boldsymbol{\tau}_{in} \in \mathbb{U}$ in the constraints of (4.13b)-(4.13c) shows that $\boldsymbol{\tau}_{in}$ itself is an element of the set defined by these constraints. Hence, this is a convex non-empty set. Therefore, (4.13) is a feasible strictly convex inequality constrained QP problem, which assumes a global minimum.

Denoting $\mathbf{f}' \in \mathbb{R}^n$ the wrench such that $\mathbf{f}' = \mathbf{W}(\mathbf{x})\,\boldsymbol{\tau}^a$, the definition (4.13) indicates that both $\tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau}^a)$ and $\tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau}^b)$ are obtained with

$$\tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau}^a) = \tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau}^b) = \arg\min_{\boldsymbol{\tau}'} \|\boldsymbol{\tau}'\|_2^2 \quad , \tag{4.33}$$

$$\text{s.t. } \mathbf{W}(\mathbf{x})\,\boldsymbol{\tau}' = \mathbf{f}' \tag{4.34}$$

$$\boldsymbol{\tau}_{min} \leqslant \boldsymbol{\tau}' \leqslant \boldsymbol{\tau}_{max} \tag{4.35}$$

and, therefore, (4.32). □

Lemma 4.9 shows that if two vectors of cable tensions generate the same wrench in a given pose, they lead to the same estimation of desired cable tensions. The following corollary is based on this conclusion.

Corollary 4.6. *For every* $\boldsymbol{\tau}^a$, $\boldsymbol{\tau}^b \in \mathbb{U}$ *and* $\mathbf{x} \in \mathbb{R}$, *if*

$$\boldsymbol{\tau}^b = \tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau}^a), \tag{4.36}$$

then

$$\boldsymbol{\tau}^b = \tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau}^b). \tag{4.37}$$

Proof. Taking $\boldsymbol{\tau}^a$, $\boldsymbol{\tau}^b$ and $\mathbf{x}$ from the corollary statement such that $\boldsymbol{\tau}^b = \tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau}^a)$, constraints (4.13b)-(4.13c) in the definition of $\tilde{\boldsymbol{\tau}}_r$ implies that $\mathbf{W}(\mathbf{x})\,\boldsymbol{\tau}^a = \mathbf{W}(\mathbf{x})\,\boldsymbol{\tau}^b$ and $\boldsymbol{\tau}^b \in \mathbb{U}$. Therefore, applying Lemma 4.5, $\tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau}^a) = \tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau}^b)$. Thanks to (4.36), this leads to $\boldsymbol{\tau}^b = \tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau}^b)$. □

Lemma 4.7. *For every* $\mathbf{y} \in \mathbb{Y}$, $\boldsymbol{\tau}_{in} \in \mathbb{U}$ *and* $k \in \mathbb{N}$,

$$\min_{\boldsymbol{\tau} \in \mathbb{U}} (\|\boldsymbol{\tau} - \tilde{\boldsymbol{\tau}}_r(\mathbf{x}_\mathbf{y}, \boldsymbol{\tau}_{in})\|_{\mathbf{K}_\tau}) = 0, \tag{4.38}$$

and the minimal time-varying stage cost is given by

$$\ell^*(k, \mathbf{y}) = \|\mathbf{y} - \mathbf{y}_d(k)\|_{\mathbf{K}_\mathbf{y}}^2 = e_\mathbf{y}(k, \mathbf{y}). \tag{4.39}$$

Proof. For every $\mathbf{y} \in \mathbb{Y}$ and $\boldsymbol{\tau}_{in} \in \mathbb{U}$, one may set $\boldsymbol{\tau}_{out} = \tilde{\boldsymbol{\tau}}_r(\mathbf{x}_\mathbf{y}, \boldsymbol{\tau}_{in})$. Thanks to the constraints (4.13c), $\boldsymbol{\tau}_{out} \in \mathbb{U}$. Moreover, $\|\boldsymbol{\tau}_{out} - \tilde{\boldsymbol{\tau}}_r(\mathbf{x}_\mathbf{y}, \boldsymbol{\tau}_{in})\|_{\mathbf{K}_\tau} = 0$. Since the expression $\|\boldsymbol{\tau}_{out} - \tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau}_{in})\|_{\mathbf{K}_\tau} = 0$ is non-negative, this proves (4.38). Substituting (4.38) in (4.30), equation (4.39) is obtained. □

As discussed in Section 4.2.2, $\tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau})$ is the vector of cable tensions with minimal 2-norm that generates the same wrench than $\boldsymbol{\tau}$ in pose $\mathbf{x}$. Loosely speaking, the influence of $\tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau})$ and $\boldsymbol{\tau}$ on the system dynamics should be equivalent. This equivalence is described in a more precise manner in the following assumption.

Assumption 4.8. *Consider* $\mathbf{y}$, $\mathbf{y}^+ \in \mathbb{Y}$ *and* $\boldsymbol{\tau}^a$, $\boldsymbol{\tau}^b \in \mathbb{U}$. *If these vectors satisfy*

$$\mathbf{y}^+ = \boldsymbol{\phi}_\mathbf{y}(\mathbf{y}, \boldsymbol{\tau}^a) \text{ and} \tag{4.40a}$$

$$\mathbf{W}(\mathbf{x}_\mathbf{y})\, \boldsymbol{\tau}^b = \mathbf{W}(\mathbf{x}_\mathbf{y})\, \boldsymbol{\tau}^a \tag{4.40b}$$

then, the following relation also holds

$$\mathbf{y}^+ = \boldsymbol{\phi}_\mathbf{y}(\mathbf{y}, \boldsymbol{\tau}^b) = \boldsymbol{\phi}_\mathbf{y}(\mathbf{y}, \boldsymbol{\tau}^a). \tag{4.41}$$

Limitations resulting from this Assumption 4.8 are better discussed in Section 4.3. This assumption is also commonly used in the state-of-the-art control schemes presented in Section 1.4.

Based on Assumption 4.8, for a given initial state vector, two sequences of cable tensions that generate identical wrenches on the corresponding poses result in identical trajectories. Therefore, any trajectory performed with an arbitrary sequence of feasible cable tensions may be also generated with an alternative sequence consisting of minimal 2-norm vectors. In this case, the stage cost related to each instant along the trajectory is equal to the minimal time-varying cost. This assertion is rigorously formulated in Lemma 4.9.

Lemma 4.9. *For each* $\mathbf{y}_a \in \mathbb{Y}$, $\mathbf{y}(\cdot) \in \mathbb{S}^{2n}$ *and* $\boldsymbol{\tau}(\cdot) \in \mathbb{S}^m$ *such that, for all* $k \in \mathbb{N}$,

$$\mathbf{y}_k = \mathbf{y}_{\boldsymbol{\tau}(\cdot)}(k, \mathbf{y}_a) \in \mathbb{Y} \tag{4.42a}$$

$$\boldsymbol{\tau}_k \in \mathbb{U}, \tag{4.42b}$$

there exists a $\boldsymbol{\tau}^*(\cdot) \in \mathbb{S}^m$ *that satisfies*

$$\mathbf{y}_{\boldsymbol{\tau}(\cdot)}(k, \mathbf{y}_a) = \mathbf{y}_{\boldsymbol{\tau}^*(\cdot)}(k, \mathbf{y}_a) = \mathbf{y}_k, \tag{4.43a}$$

$$\tilde{\boldsymbol{\tau}}_r(\mathbf{x}_{\mathbf{y}_k}, \boldsymbol{\tau}_k^*) = \boldsymbol{\tau}_k^* \in \mathbb{U} \tag{4.43b}$$

$$\ell(k, \mathbf{y}_k, \boldsymbol{\tau}_k^*) = e_\mathbf{y}(k, \mathbf{y}_k) = \ell^*(k, \mathbf{y}_k) \tag{4.43c}$$

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

Proof. For $\mathbf{y}_a$, $\mathbf{y}(\cdot)$ and $\boldsymbol{\tau}(\cdot)$ stated in the lemma, define $\boldsymbol{\tau}^*(\cdot) \in \mathbb{S}^m$ and $\mathbf{y}^*(\cdot) \in \mathbb{S}^{2n}$ according to

$$\begin{aligned} \boldsymbol{\tau}_k^* &= \tilde{\boldsymbol{\tau}}_r(\mathbf{x}_{\mathbf{y}_k}, \boldsymbol{\tau}_k), \\ \mathbf{y}_k^* &= \mathbf{y}_{\boldsymbol{\tau}^*(\cdot)}(k, \mathbf{y}_a) \end{aligned} \tag{4.44}$$

for all $k \in \mathbb{N}$. Due to the constraint (4.13b),

$$\mathbf{W}(\mathbf{x}_{\mathbf{y}_a})\,\boldsymbol{\tau}_1 = \mathbf{W}(\mathbf{x}_{\mathbf{y}_a})\,\boldsymbol{\tau}_1^* \tag{4.45}$$

and, according to Assumption 4.8,

$$\mathbf{y}_1^* = \boldsymbol{\phi}_{\mathbf{y}}(\mathbf{y}_a, \boldsymbol{\tau}_1^*) = \mathbf{y}_{\boldsymbol{\tau}^*(\cdot)}(1, \mathbf{y}_a) = \mathbf{y}_1. \tag{4.46}$$

Moreover, (4.44)-(4.45) and Lemma 4.5 imply that

$$\tilde{\boldsymbol{\tau}}_r(\mathbf{x}_{\mathbf{y_1}}, \boldsymbol{\tau}_1^*) = \tilde{\boldsymbol{\tau}}_r(\mathbf{x}_{\mathbf{y_1}}, \boldsymbol{\tau}_1) = \boldsymbol{\tau}_1^*. \tag{4.47}$$

Replacing recursively $\mathbf{y}_a$ by $\mathbf{y}_k$, $\forall\, k \in \mathbb{N}$ in (4.45)-(4.47), equations (4.43a) and (4.43b) are obtained.

Equation (4.43b) and Lemma 4.7 imply that

$$\ell(k, \mathbf{y}_k, \boldsymbol{\tau}_k^*) = e_{\mathbf{y}}(k, \mathbf{y}_k) + \|\boldsymbol{\tau}_k^* - \tilde{\boldsymbol{\tau}}_r(\mathbf{x}_{\mathbf{y}_k}, \boldsymbol{\tau}_k^*)\|_{\mathbf{K}_{\boldsymbol{\tau}}} = e_{\mathbf{y}}(k, \mathbf{y}_k) = \ell^*(k, \mathbf{y}_k), \tag{4.48}$$

as in (4.43c). □

Finally, the main results regarding the stability of the ST-NMPC scheme proposed in Section 4.2.1 may be presented as follows.

Theorem 4.10. *Consider the ST-NMPC Algorithm 1 with prediction horizon $h_p \in \mathbb{N}$ and feasible desired trajectory $\mathbf{y}_d(\cdot) \in \mathbb{S}^{2n}$. Consider also that Assumptions 4.4 and 4.8 hold. Then, the nominal closed-loop system* (4.16) *is uniformly asymptotically stable on $\mathbb{Y}$ provided that h_p is sufficiently large.*

Proof. The proof consists of the analysis of the conditions presented in Theorem 4.3. First, it is necessary to prove that there exist $\alpha_1,\, \alpha_2 \in \mathcal{K}_\infty$ such that $\alpha_1(\|\mathbf{y} - \mathbf{y}_d(k)\|) \leqslant \ell^*(k, \mathbf{y}) \leqslant \alpha_2(\|\mathbf{y} - \mathbf{y}_d(k)\|)\ \forall\, k \in \mathbb{N}$. According to Lemma 4.7, $\ell^*(k, \mathbf{y}) = \|\mathbf{y} - \mathbf{y}_d(k)\|^2_{\mathbf{K}_{\mathbf{y}}}$ such that the following inequalities hold for all $k \in \mathbb{N}$

$$K_{\mathbf{y}}^{min}\,\|\mathbf{y} - \mathbf{y}_d(k)\|^2 \leqslant \ell^*(k, \mathbf{y}) = \|\mathbf{y} - \mathbf{y}_d(k)\|^2_{\mathbf{K}_{\mathbf{y}}} \leqslant K_{\mathbf{y}}^{max}\,\|\mathbf{y} - \mathbf{y}_d(k)\|^2. \tag{4.49}$$

where $K_{\mathbf{y}}^{min}$ and $K_{\mathbf{y}}^{max}$ were defined in (4.29).

Defining $\alpha_1,\, \alpha_2 \in \mathcal{K}_\infty$ by $\alpha_1(r) = K_{\mathbf{y}}^{min}\, r^2$ and $\alpha_2(r) = K_{\mathbf{y}}^{max}\, r^2$, (4.49) can be written as

$$\alpha_1(\|\mathbf{y}\|_{\mathbf{y}_d(k)}) \leqslant \ell^*(k, \mathbf{y}) \leqslant \alpha_2(\|\mathbf{y}\|_{\mathbf{y}_d(k)})\ \forall\, k \in \mathbb{N}. \tag{4.50}$$

In addition, (4.21) should be proved. If Assumption 4.4 holds, then for each $\mathbf{y}_a \in \mathbb{Y}$ and $k \in \mathbb{N}$, there exist $\overline{\boldsymbol{\tau}}(\cdot) \in \mathbb{S}^m$, real $C_1, < \infty$ and $\sigma_1 \in (0, 1)$ such that

$$e_{\mathbf{y}}\left(k + j, \mathbf{y}_{\overline{\boldsymbol{\tau}}(\cdot)}(j, \mathbf{y}_a)\right) \leqslant C_1\, \sigma_1^j\, e_{\mathbf{y}}(k, \mathbf{y}_a) \tag{4.51a}$$

$$\overline{\boldsymbol{\tau}}_j \in \mathbb{U} \tag{4.51b}$$

for all $j \in \mathbb{N}$. Applying Lemma 4.9 with $\boldsymbol{\tau}(\cdot) = \overline{\boldsymbol{\tau}}(\cdot)$, the obtained sequence $\boldsymbol{\tau}^*(\cdot)$ leads to the same trajectory (as implied by (4.43a)) and, therefore, (4.51a) remains valid. In addition, (4.43c) implies that

$$\ell\Big(j+k, \mathbf{y}_{\boldsymbol{\tau}^*(\cdot)}(j, \mathbf{y}_a), \boldsymbol{\tau}_j^*\Big) = e_{\mathbf{y}}(j+k, \mathbf{y}_{\boldsymbol{\tau}^*(\cdot)}(j, \mathbf{y}_a)) \tag{4.52a}$$

$$\leqslant C_1\, \sigma_1^j\, e_{\mathbf{y}}(k, \mathbf{y}_a) \tag{4.52b}$$

$$= C_1\, \sigma_1^j\, \ell^*(k, \mathbf{y}_a), \tag{4.52c}$$

where (4.51a) and (4.39) were used in order to obtain (4.52b) and (4.52c), respectively. Inequality (4.52) proves (4.21), finalizing the stability analysis. □

4.3 Numerical Implementation of the Control Algorithms

The present section introduces the main numerical algorithms necessary to the implementation of the control scheme discussed in Section 4.2. Section 4.3.1 proposes a numerical discretization method, Section 4.3.2 introduces the algorithm to compute $\tilde{\boldsymbol{\tau}}_r$ and the numerical solution of the optimal control problem is discussed in Section 4.3.3.

4.3.1 Numerical discretization

As discussed in Section 4.1, a discrete-time system approximating the continuous-time model (3.10) is necessary in order to implement the proposed NMPC.

As for general nonlinear systems, it is not possible to compute the exact solution of the initial value problem (4.3) (represented by the transition mapping $\boldsymbol{\phi}_{\mathbf{y}}$ in (4.5)). Therefore, one may define an estimation $\tilde{\boldsymbol{\phi}}_{\mathbf{y}}$ of the exact transition mapping $\boldsymbol{\phi}_{\mathbf{y}}$. In practice, a discrete-time system is obtained using some kind of numerical integration method such as the Euler's or Runge-Kutta's methods [186]. The control scheme proposed in Section 4.2 may be implemented with any of these integration algorithms.

Recalling the IVP (4.3), explicit numerical integration methods return $\tilde{\mathbf{y}}^+ = \tilde{\boldsymbol{\phi}}(\mathbf{y}_a, \boldsymbol{\tau}_a)$ based on the computation of $\ddot{\boldsymbol{x}}$ according to (4.3a) for suitably chosen combinations of $\boldsymbol{x}$ and $\dot{\boldsymbol{x}}$ in the neighborhood of $\mathbf{x}_a$ and $\mathbf{x}_a$. Calling n_ϕ the number of considered combinations of $\boldsymbol{x}$ and $\dot{\boldsymbol{x}}$, the integration error may be reduced increasing n_ϕ. This is illustrated in Figure 4.2 comparing the approximation given by the Euler's method for $n_{\phi,1} = 3$ and $n_{\phi,2} = 6$. The resulting error for $n_{\phi,2}$ is approximately the half of the one obtained for $n_{\phi,1}$.

The maximal error obtained with the approximate transition mapping is denoted by $\epsilon \in \mathbb{R}_+^0$, such that

$$\|\boldsymbol{\phi}_{\mathbf{y}}(\mathbf{y}, \boldsymbol{\tau}) - \tilde{\boldsymbol{\phi}}_{\mathbf{y}}(\mathbf{y}, \boldsymbol{\tau})\| \leqslant \epsilon \;\forall\, \mathbf{y} \in \mathbb{Y},\; \boldsymbol{\tau} \in \mathbb{U}. \tag{4.53}$$

It should be noted that the stability analysis presented in the previous sections neglects the error ϵ. In order to prove the stability of the closed-loop system considering

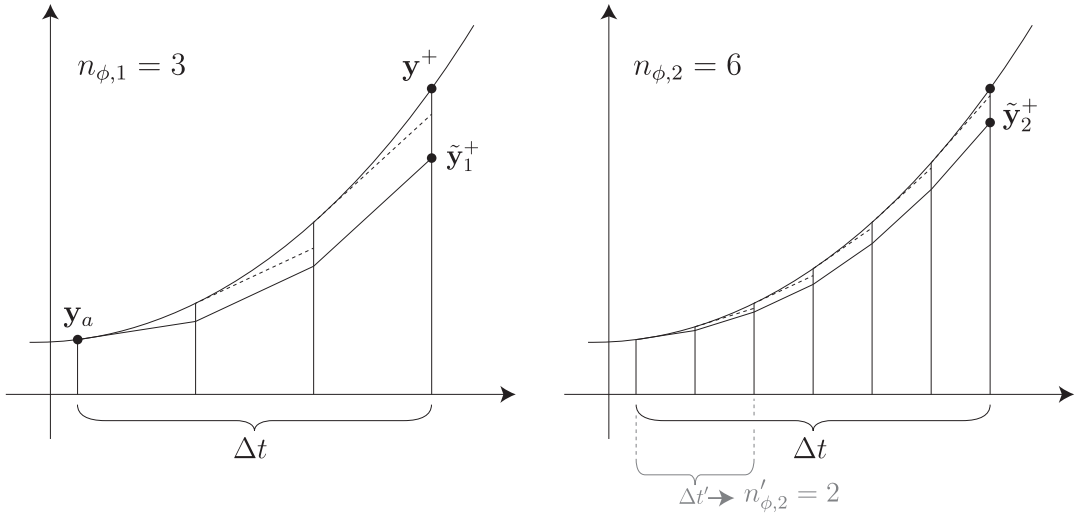

Figure 4.2 – *Illustration of how the numerical integration precision is influenced by Δt and n_ϕ for the Euler's method.*

this divergence, one may analyze the robustness of the control scheme and consider the numerical discretization errors as disturbances. This is indeed, a possible follow-up of the studies presented in this thesis.

Besides the number of evaluations n_ϕ, the error ϵ is also influenced by the time step Δt and the implemented numerical discretization method. Generally, shorter time steps lead to smaller discretization errors. Figure 4.2 illustrates this fact considering a reduced $\Delta t'$. Clearly, the original accumulated error (obtained for $n_{\phi,2} = 6$ and Δt) is obtained for $\Delta t' = \Delta t/3$ and $n'_{\phi,2} = 2$.

Similarly, the numerical integration method should be chosen in order to obtain a good compromise between precision and computational burden. Figure 4.3 compares the Euler's and Runge-Kutta's methods. One may note that improved results are obtained with Runge-Kutta's method. Indeed, Runge-Kutta with $n_\phi = 4$ (known as classical Runge-Kutta's method) is frequently used for the numerical solution of ordinary differential equations. For most of the applications, this represents a good compromise between computation time and numerical precision. The interested reader is referred to the large amount of textbooks addressing the numerical solution of ordinary differential equations, *e.g.* [176, 187].

In spite of the advantages obtained with advanced numerical discretization methods, some specific concerns related to the proposed NMPC scheme should be evaluated. Firstly, the sampling period used for the position tracking of CDPRs is shorter than typical applications of MPC schemes. Section 1.7 recalled the example of $\Delta t = 1$ min used in [136], whereas the control schemes implemented in this thesis use sampling periods close to 10 ms. Moreover, the approximate transition mapping $\tilde{\phi}$ should be integrated in the OCP formulation (4.14) in order to obtain a standard NLP problem (this is discussed in detail in Section 4.3.3.1). Accordingly, intricate formulations of $\tilde{\phi}$ may hinder the numerical solution of (4.14).

Therefore, a rather simple method is proposed in this thesis. Similarly to Chapter 3, $n_\phi = 1$ is chosen. In this case, Euler's and Runge-Kutta's methods are equivalent. For

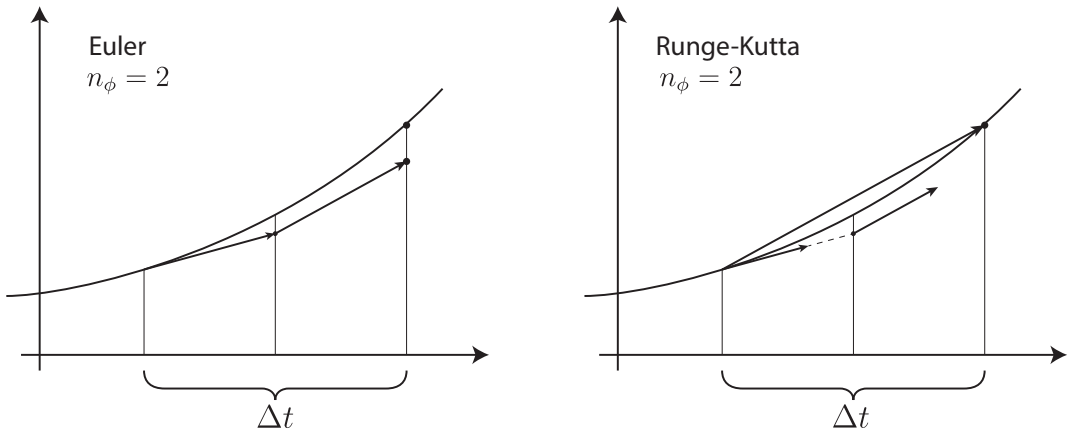

Figure 4.3 – *Comparison between Euler's and Runge-Kutta's methods for the same* n_ϕ.

an actual time t_a, the derivatives of $\boldsymbol{x}$ are considered constant within the sampling period, *i.e.* $\dot{\boldsymbol{x}}(t) \approx \dot{\boldsymbol{x}}(t_a)$, $\ddot{\boldsymbol{x}}(t) \approx \ddot{\boldsymbol{x}}(t_a)$ for $t \in \mathbb{R}_{t_a, t_a+\Delta t}$ and

$$\begin{aligned} \boldsymbol{x}(t_a + \Delta t) &\approx \boldsymbol{x}(t_a) + \Delta t\, \dot{\boldsymbol{x}}(t_a) \\ \dot{\boldsymbol{x}}(t_a + \Delta t) &\approx \dot{\boldsymbol{x}}(t_a) + \Delta t\, \ddot{\boldsymbol{x}}(t_a). \end{aligned} \tag{4.54}$$

Hence, the transition mapping introduced in Chapter 3 is also used here:

$$\begin{aligned} \tilde{\mathbf{y}}^+ = \tilde{\boldsymbol{\phi}}_{\mathbf{y}}(\mathbf{y}, \boldsymbol{\tau}) = \overbrace{\begin{bmatrix} \mathbf{I} & \Delta t\, \mathbf{I} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}}^{\mathbf{A}} \mathbf{y} + \overbrace{\begin{bmatrix} \mathbf{0} \\ \Delta t\, \mathbf{M}(\mathbf{x}_\mathbf{y})^{-1}\, \mathbf{W}(\mathbf{x}_\mathbf{y}) \end{bmatrix}}^{\mathbf{B}(\mathbf{y})} \boldsymbol{\tau} + \\ + \underbrace{\begin{bmatrix} \mathbf{0} \\ \Delta t\, \mathbf{M}(\mathbf{x}_\mathbf{y})^{-1} \big(\mathbf{g}(\mathbf{x}_\mathbf{y}) - \mathbf{C}(\mathbf{x}_\mathbf{y}, \dot{\mathbf{x}}_\mathbf{y}) \dot{\mathbf{x}}_\mathbf{y} \big) \end{bmatrix}}_{\mathbf{v}(\mathbf{y})}. \end{aligned} \tag{4.55}$$

It is worth noting that estimation (4.55) satisfies Assumption 4.8, as proved in the following. For the state $\mathbf{y} \in \mathbb{Y}$ and cable tension vectors $\boldsymbol{\tau}^a$, $\boldsymbol{\tau}^b \in \mathbb{U}$ stated in Assumption 4.8, denote by $\mathbf{f}' \in \mathbb{R}^n$ the wrench vector given by

$$\mathbf{f}' = \mathbf{W}(\mathbf{x}_\mathbf{y})\, \boldsymbol{\tau}^a = \mathbf{W}(\mathbf{x}_\mathbf{y})\, \boldsymbol{\tau}^b, \tag{4.56}$$

where (4.40b) was used. The application of (4.55) leads to

$$\tilde{\boldsymbol{\phi}}_{\mathbf{y}}(\mathbf{y}, \boldsymbol{\tau}^b) = \tilde{\boldsymbol{\phi}}_{\mathbf{y}}(\mathbf{y}, \boldsymbol{\tau}^a) = \mathbf{A}\, \mathbf{y} + \begin{bmatrix} \mathbf{0} \\ \Delta t\, \mathbf{M}(\mathbf{x}_\mathbf{y})^{-1}\, \mathbf{f}' \end{bmatrix} + \mathbf{v}(\mathbf{y}), \tag{4.57}$$

which is equivalent to (4.41).

This means that the presented stability results are valid for a closed-loop system defined as

$$\mathbf{y}_{k+1} = \tilde{\boldsymbol{\phi}}_{\mathbf{y}} \left(\mathbf{y}_k, \boldsymbol{\tau}_{fb}(k, \mathbf{y}_k) \right), \tag{4.58}$$

with the NMPC feedback policy $\boldsymbol{\tau}_{fb}$ defined in Algorithm 1.

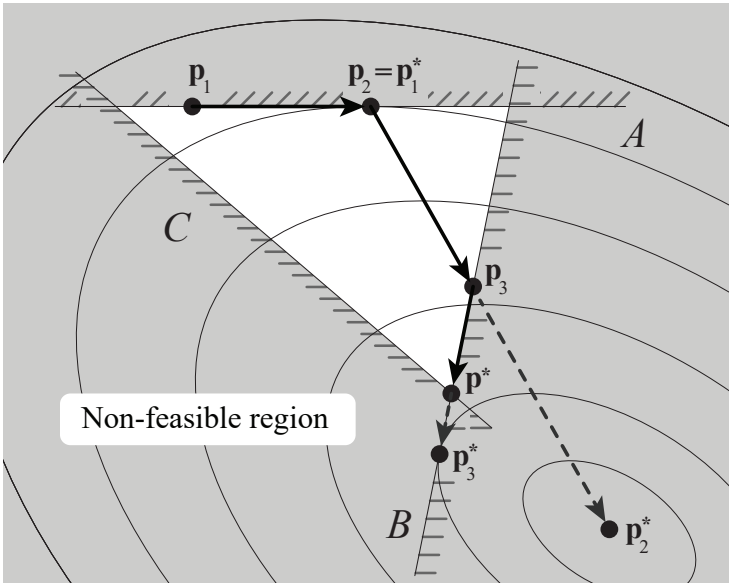

Figure 4.4 – *Active-set method: illustrative example of steps taken for the solution of a constrained optimization problem.*

Nevertheless, a general transition mapping does not necessarily satisfy this assumption. Similarly to the influence of the error ϵ, the presented stability results are valid for systems in which $\|\boldsymbol{\phi}(\mathbf{y},\boldsymbol{\tau}) - \tilde{\boldsymbol{\phi}}(\mathbf{y},\boldsymbol{\tau})\|$ are smaller than the disturbance magnitude established by the robustness of the NMPC controller. The same aforementioned robustness study should be done in order to quantify this magnitude.

4.3.2 Estimation of the desired cable tensions $\tilde{\boldsymbol{\tau}}_r$

The estimation of the desired cable tensions defined according to the optimization (4.13) plays an important role in the stability analysis of the proposed control schemes. The present section introduces an algorithm able to solve this optimization problem.

More in detail, the efficiency of the proposed NMPCs depends on the fast computation of $\tilde{\boldsymbol{\tau}}_r(\mathbf{x},\boldsymbol{\tau})$. For this reason, special attention is dedicated to some details involved in the numerical solution of (4.13).

Denoting $\mathbf{A}_{ineq} = \begin{bmatrix}\mathbf{I} & -\mathbf{I}\end{bmatrix}^T$ and $\mathbf{b}_{ineq} = \begin{bmatrix}\boldsymbol{\tau}_{min}^T & -\boldsymbol{\tau}_{max}^T\end{bmatrix}^T$, the problem (4.13) may be written as

$$\tilde{\boldsymbol{\tau}}_r(\mathbf{x},\boldsymbol{\tau}_{in}) = \arg\min_{\boldsymbol{\tau}'} \|\boldsymbol{\tau}'\|_2^2 \tag{4.59a}$$

$$\text{s.t. } \mathbf{W}(\mathbf{x})\,\boldsymbol{\tau}_{in} = \mathbf{W}(\mathbf{x})\,\boldsymbol{\tau}' \tag{4.59b}$$

$$\mathbf{A}_{ineq}\,\boldsymbol{\tau}' \geqslant \mathbf{b}_{ineq} \tag{4.59c}$$

Taking $\boldsymbol{\tau}' = \boldsymbol{\tau}_{in}$, (4.13b) (and (4.59b)) are satisfied. Moreover, in accordance with the domain declared in (4.13), $\boldsymbol{\tau}_{in} \in \mathbb{U}$. As a result, (4.13c) (and (4.59c)) are satisfied substituting $\boldsymbol{\tau}_{in} = \boldsymbol{\tau}'$. The cable tension vector $\boldsymbol{\tau}_{in}$ is, therefore, in the feasible region defined by (4.59b)-(4.59c) and may be used as a feasible initial guess in an iterative active-set method [99].

Figure 4.4 depicts an illustrative example of the methodology implemented with the active-set. The feasible region is determined by three linear inequalities, represented by A, B and C. Taking as initial guess the vector $\mathbf{p}_1$, the algorithm is able to compute the global optimum $\mathbf{p}^*$ within 3 iterations. In each iteration, an equality constrained optimization problem is solved considering some of the inequalities as *equalities*. In a given iteration, constraints that are considered as equalities are called *active* constraints. The remaining constraints, which are neglected, are called *inactive* constraints. Taking the initial guess $\mathbf{p}_1$ in the first iteration, the inequality A is considered as an equality and inequalities B and C are neglected (inactive). The vector $\mathbf{p}_1^*$ is obtained solving this subproblem and is taken as input for the next iteration.

One may conclude that the cost function may be decreased by setting inequality constraint A as inactive (this is done analyzing the Lagrangian multipliers related to constraint A, considered as equality in iteration 1). Therefore, iteration 2 solves a subproblem neglecting this constraint. The obtained solution is $\mathbf{p}_2^*$. However, this vector is not within the feasible region. Accordingly, the next iteration considers $\mathbf{p}_3 = \mathbf{p}_2 + \alpha_2 \left(\mathbf{p}_2^* - \mathbf{p}_2\right)$, with some $\alpha_2 \in [0, 1]$. Since an additional constraint is faced, iteration 3 considers B as an equality constraint. The solution of the optimization problem taking B as equality constraint returns the vector $\mathbf{p}_3^*$. As in iteration 2, the global optimum is obtained as $\mathbf{p}^* = \mathbf{p}_3 + \alpha_3(\mathbf{p}_3^* - \mathbf{p}_3)$.

Similarly, the active-set method solving (4.59) computes a $\boldsymbol{\tau}_k^*$ in an iteration k as the solution of an equality-constrained QP problem in which some of the inequalities (4.59c) are set as equalities [1]. Denoting $\mathbf{f}_{in} = \mathbf{W}(\mathbf{x})\,\boldsymbol{\tau}_{in}$, the sub-problems considered in an iteration k has the following form

$$\boldsymbol{\tau}_k^* = \arg\min_{\boldsymbol{\tau}} \ \boldsymbol{\tau}^T\boldsymbol{\tau} \tag{4.60a}$$

$$\text{s. t.} \quad \underbrace{\begin{bmatrix} \mathbf{W}(\mathbf{x}) \\ \mathbf{a}_{a_{1,k}} \\ \mathbf{a}_{a_{2,k}} \\ \vdots \end{bmatrix}}_{\mathbf{A}_{eq,k}} \boldsymbol{\tau} = \underbrace{\begin{bmatrix} \mathbf{f}_{in} \\ b_{a_{1,k}} \\ b_{a_{2,k}} \\ \vdots \end{bmatrix}}_{\mathbf{b}_{eq,k}} \tag{4.60b}$$

Vectors $\mathbf{a}_{a_{i,k}}$ are row vectors of $\mathbf{A}_{ineq}$, which have one term equal to 1 or -1 and the remaining elements equal to 0. Similarly, $b_{a_{i,k}}$ are elements of $\mathbf{b}_{ineq}$. The indices $a_{1,k}$, $a_{2,k}$, ... are given by the set of active constraints $\mathcal{A}_k = \{a_{1,k}, a_{2,k}, \ldots, a_{n_{a,k}}\} \subset \mathbb{N}_{1,2m}$, with $n_{a,k}$ the number of active inequalities in iteration k. The overall number of equalities in (4.60) is $n_{e,k} = n_{a,k} + n$. Clearly, considering that $\tau_{min,i} < \tau_{max,i}\ \forall\ i \in \mathbb{N}_{1,m}$, if a constraint $\tau_i = \tau_{min,i}$ is active, then $\tau_i \neq \tau_{max,i}$ (its counterpart for $\tau_{max,i}$ is not active) and vice-versa. Moreover, assuming that matrices $\mathbf{A}_{eq,k}$ are full rank, constraints (4.60b) may be seen as a system of $n_{e,k}$ linearly independent equations with m variables. Therefore, $n_{e,k} \leqslant m$.

The optimization problem (4.60) is strictly convex with an Hessian matrix equals to the identity matrix. A vector $\boldsymbol{\tau} \in \mathbb{R}^m$ satisfying the first-order optimality conditions of

1. Even though the proposed numerical solution generates a sequence of cable tensions $\{\boldsymbol{\tau}_1, \boldsymbol{\tau}_2, \ldots\}$, this sequence does not present any relation with the evolution of tension in time (as it was the case of sequences $\boldsymbol{\tau}(\cdot)$ in Section 4.2).

this equality constrained problem is a global minimum [99, Theorem 16.2]. The first-order optimality conditions (KKT conditions) for (4.60) can be written as

$$\boldsymbol{\tau} + \mathbf{A}_{eq,k}^T \, \boldsymbol{\mu}_k = \mathbf{0} \tag{4.61a}$$

$$\mathbf{A}_{eq,k} \, \boldsymbol{\tau} = \mathbf{b}_{eq,k}, \tag{4.61b}$$

where $\boldsymbol{\mu}_k \in \mathbb{R}^{n_{e,k}}$ is the vector of Lagrange multipliers.

In order to solve (4.61), the matrix $\mathbf{A}_{eq,k}^T$ may be factorized with QR decomposition such that

$$\mathbf{A}_{eq,k}^T = \begin{bmatrix} \mathbf{Y}_k & \mathbf{Z}_k \end{bmatrix} \begin{bmatrix} \mathbf{R}_k \\ \mathbf{0} \end{bmatrix}, \tag{4.62}$$

with $\mathbf{Y}_k$, $\mathbf{Z}_k$ orthonormal and $\mathbf{R}_k$ square and upper triangular.

According to the definition of $\mathbf{Y}_k$ and $\mathbf{Z}_k$, column vectors of $\begin{bmatrix} \mathbf{Y}_k & \mathbf{Z}_k \end{bmatrix}$ form an orthonormal basis for $\mathbb{R}^m$. Therefore, any vector in $\mathbb{R}^m$ may be written as a linear combination of the column vectors of this matrix. Denoting the number of columns of $\mathbf{Y}_k$ and $\mathbf{Z}_k$ as $n_{\mathbf{y}} = n_{e,k}$ and $n_{\mathbf{z}} = m - n_{e,k}$, respectively, this implies that there exists $\boldsymbol{\tau}_{\mathbf{y}} \in \mathbb{R}^{n_{\mathbf{y}}}$ and $\boldsymbol{\tau}_{\mathbf{z}} \in \mathbb{R}^{n_{\mathbf{z}}}$ such that the solution of (4.60) may be written as

$$\boldsymbol{\tau}_k^* = \mathbf{Y}_k \, \boldsymbol{\tau}_{\mathbf{y}} + \mathbf{Z}_k \, \boldsymbol{\tau}_{\mathbf{z}} \tag{4.63}$$

Left multiplying (4.61a) by $\mathbf{Z}_k^T$ and recalling that $\mathbf{Z}_k^T \, \mathbf{A}_{eq,k}^T = \mathbf{Z}_k^T \, \mathbf{Y}_k = \mathbf{0}$,

$$\mathbf{Z}_k^T \, \mathbf{Z}_k \, \boldsymbol{\tau}_{\mathbf{z}} = \mathbf{0} \Rightarrow \boldsymbol{\tau}_{\mathbf{z}} = 0. \tag{4.64}$$

Similarly, since $\mathbf{A}_{eq,k} \, \mathbf{Z}_k = \mathbf{0}$, (4.63) substituted in (4.61b) leads to

$$\mathbf{A}_{eq,k} \, \mathbf{Y}_k \boldsymbol{\tau}_{\mathbf{y}} = \mathbf{b}_{eq,k} \tag{4.65}$$

and using (4.62) with the orthogonality of $\mathbf{Y}_k$ ($\mathbf{Y}_k^T \, \mathbf{Y}_k = \mathbf{I}$),

$$\mathbf{R}_k^T \, \boldsymbol{\tau}_{\mathbf{y}} = \mathbf{b}_{eq,k}. \tag{4.66}$$

Since $\mathbf{R}_k$ is upper triangular, vector $\boldsymbol{\tau}_{\mathbf{y}}$ can be computed with back-substitution in (4.66). One may note that, once $\mathbf{Y}_k$, $\mathbf{R}_k$ and $\mathbf{Z}_k$ are defined, the solution $\boldsymbol{\tau}_k^* = \mathbf{Y}_k \, \boldsymbol{\tau}_{\mathbf{y}} + \mathbf{Z}_k \, \boldsymbol{\tau}_{\mathbf{z}} = \mathbf{Y}_k \, \boldsymbol{\tau}_{\mathbf{y}}$ of the sub-problem (4.60) can be obtained solving (4.66) without great computational effort. Indeed, the same solution may be obtained with

$$\boldsymbol{\tau}_k^* = \mathbf{A}_{eq,k}^{\dagger} \, \mathbf{b}_{eq,k} \tag{4.67}$$

with $\mathbf{A}_{eq,k}^{\dagger}$ the pseudo-inverse of $\mathbf{A}_{eq,k}$. Nevertheless, the back-substitution in (4.66) demands significantly less computational effort than the computation of $\mathbf{A}_{eq,k}^{\dagger}$. Details on the computation of $\mathbf{Y}_k$, $\mathbf{Z}_k$ and $\mathbf{R}_k$ are further discussed below.

Since only some of the inequalities (4.59c) are considered in (4.60), $\boldsymbol{\tau}_k^*$ may violate the cable tension limits. Hence, the definition of a feasible $\boldsymbol{\tau}_{k+1}$ is done as illustrated in Figure 4.5. Denoting $\mathbf{q} \in \mathbb{R}^m$ with $\mathbf{q} = \boldsymbol{\tau}_k^* - \boldsymbol{\tau}_k$, some $\alpha_k \in [0, 1]$ may be defined so that

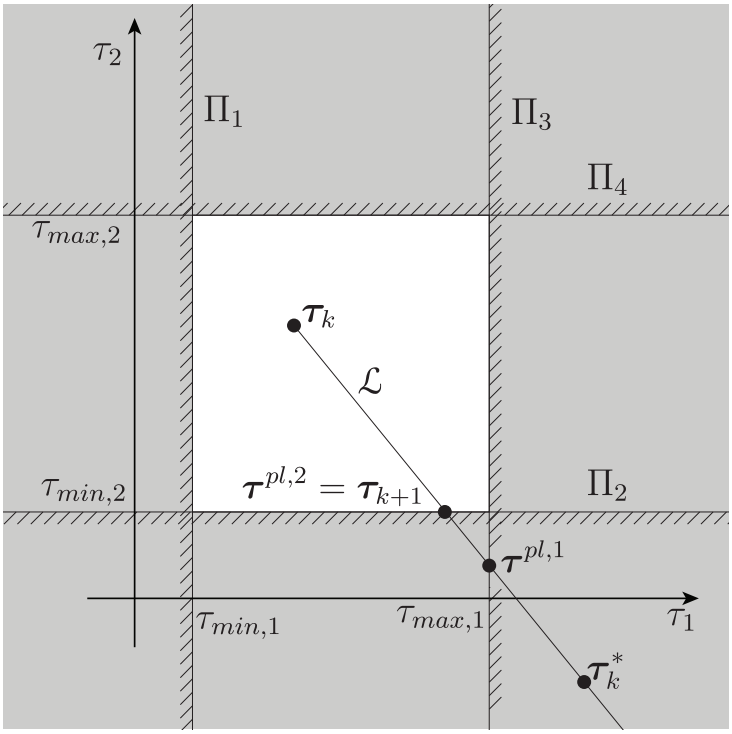

Figure 4.5 – *Notations used in order to compute* α_k*.*

$\boldsymbol{\tau}_{min} \leqslant \boldsymbol{\tau}_{k+1} = \boldsymbol{\tau}_k + \alpha_k \, \mathbf{q} \leqslant \boldsymbol{\tau}_{max}$. More in detail, it is interesting to set α_k as large as possible within $[0, 1]$ without violating the cable tension limits.

To this end, one may evaluate the intersection of each hyperplane

$$\Pi_i = \{\boldsymbol{\tau} \in \mathbb{U} \mid \mathbf{a}_i \, \boldsymbol{\tau} = b_i\}, \tag{4.68}$$

for $i \in \mathbb{N}_{1,2m}$, and the half-line

$$\mathcal{L} = \{\boldsymbol{\tau} \in \mathbb{R}^m \mid \exists \, \alpha \in \mathbb{R}_+^0 \ : \ \boldsymbol{\tau} = \boldsymbol{\tau}_k + \alpha \, \mathbf{q}\}. \tag{4.69}$$

Half-line $\mathcal{L}$ and hyperplanes Π_i are illustrated in Figure 4.5 for the case $m = 2$. Clearly, for a $j \in \mathbb{N}_{1,m}$, if $q_j > 0$, the half-line $\mathcal{L}$ does not intersect Π_j and intersects Π_{j+m} (hyperplanes defining the lower bound and upper bound of τ_j, respectively). Using the example in Figure 4.5, $q_1 = \tau_{k,1}^* - \tau_{k,1} > 0$ such that $\Pi_1 \cap \mathcal{L} = \varnothing$ and $\Pi_3 \cap \mathcal{L} = \{\boldsymbol{\tau}_1^{pl}\}$. Conversely, if $q_j < 0$, the half-line $\mathcal{L}$ intersects Π_j and does not intersect Π_{j+m}. In Figure 4.5, $\Pi_2 \cap \mathcal{L} = \{\boldsymbol{\tau}_2^{pl}\}$ and $\Pi_4 \cap \mathcal{L} = \varnothing$. Moreover, if $q_j = 0$, the intersection of Π_j (or Π_{j+m}) with $\mathcal{L}$ is either empty or the half-line $\mathcal{L}$ itself. Therefore, Π_j and Π_{j+m} are not considered in the computation of α_k if $q_j = 0$.

Thereby, for each pair Π_j, Π_{j+m} (upper and lower bound for τ_j) with $q_j \neq 0$, only one intersection point is computed. Accordingly, the elements of the vector $\boldsymbol{\tau}_b \in \mathbb{U}$ are defined as

$$\tau_{b,j} = \begin{cases} \tau_{max,j} \text{ if } q_j > 0 \\ \tau_{min,j} \text{ otherwise} \end{cases} \quad \forall \, j \in \mathbb{N}_{1,m}, \tag{4.70}$$

and the intersection of Π_j (or Π_{j+m}) with $q_j \neq 0$ may be expressed as

$$\boldsymbol{\tau}^{pl,j} = \boldsymbol{\tau}_k + \underbrace{\frac{\tau_{b,j} - \tau_{k,j}}{q_j}}_{\alpha'_j} \mathbf{q}. \tag{4.71}$$

If every $\alpha'_j > 1$, then $\boldsymbol{\tau}^*_k \in \mathbb{U}$ and $\boldsymbol{\tau}_{k+1} = \boldsymbol{\tau}^*_k = \boldsymbol{\tau}_k + \mathbf{q}$. Otherwise, the minimal value of α'_j should be taken. In this case, the inequality corresponding to the minimal α'_j (called blocking constraint) is added to the set of active constraints. The index of the inequality in (4.59c) representing the blocking constraint is denoted by j_b.

More precisely, the scalar α_k is computed according to

$$\alpha_k = \min\left(1, \min_{j \in \mathcal{I}_k} \frac{\tau_{b,j} - \tau_j}{q_j}\right), \tag{4.72}$$

with $\mathcal{I}_k \subset \mathbb{N}_{1,m}$ defined as

$$\mathcal{I}_k = \{j \in \mathbb{N}_{1,m} \mid q_j \neq 0\}. \tag{4.73}$$

Once α_k is computed, the next iteration takes the feasible $\boldsymbol{\tau}_{k+1} = \boldsymbol{\tau}_k + \alpha_k \mathbf{q} \in \mathbb{U}$ and, if a blocking constraint is met, the corresponding inequality is added as equality in the set $\mathcal{A}_{k+1}$.

In a given iteration k, a possible solution of (4.60) may be $\boldsymbol{\tau}^*_k = \boldsymbol{\tau}_k$ itself. In this case, $\mathbf{q} = \mathbf{0}$ and the computation of α_k is inappropriate ($\mathcal{I}_k = \varnothing$). Then, if the current $\boldsymbol{\tau}_k$ is not optimal, the active set $\mathcal{A}_k$ should be updated. Possibly, dropping one equality constraint from the set of active constraints may allow a further reduction of the cost function without violating the cable tension limits. The Lagrange multipliers in (4.61a) contains sufficient information to perform the evaluation of which equality constraint should be dropped.

The vector of Lagrangian multipliers $\boldsymbol{\mu}_k$ may be decomposed as $\boldsymbol{\mu}_k = \begin{bmatrix}\boldsymbol{\mu}^T_{eq,k} & \boldsymbol{\mu}^T_{ineq,k}\end{bmatrix}^T$ with $\boldsymbol{\mu}_{eq,k} \in \mathbb{R}^n$ and $\boldsymbol{\mu}_{ineq,k} \in \mathbb{R}^{n_{a,k}}$. The vector $\boldsymbol{\mu}_{eq,k}$ is related to the constraints

$$\mathbf{W}(\mathbf{x})\,\boldsymbol{\tau} = \mathbf{f}_{in}, \tag{4.74}$$

while the vector $\boldsymbol{\mu}_{ineq,k}$ is related to the constraints

$$\begin{bmatrix}\mathbf{a}_{a_1,k} \\ \mathbf{a}_{a_2,k} \\ \vdots\end{bmatrix} \boldsymbol{\tau} = \begin{bmatrix}b_{a_1,k} \\ b_{a_2,k} \\ \vdots\end{bmatrix}. \tag{4.75}$$

According to [99, Theorem 16.4], since the Hessian of (4.59) is positive definite, a $\boldsymbol{\tau}$ satisfying the first order optimality conditions of (4.59) is the global minimum of this QP problem. Moreover, the vector $\boldsymbol{\tau}_k$ is optimal if every element of $\boldsymbol{\mu}_{ineq,k}$ is non-negative for an iteration returning $\boldsymbol{\tau}_{k+1} = \boldsymbol{\tau}^*_k = \boldsymbol{\tau}_k$ [99, Section 16.5]. On the other hand, if one of its elements is negative, this indicates that dropping the corresponding inequality from $\mathcal{A}$ leads to a lower value of the cost function without violating the cable

tension limits. Accordingly, denoting by j_{min} the index of the inequality that leads to the (negative) minimal element of $\boldsymbol{\mu}_{ineq,k}$, the active set considered in the next iteration is taken as

$$\mathcal{A}_{k+1} = \mathcal{A}_k \backslash \{j_{min}\}. \tag{4.76}$$

However, it is necessary to compute $\boldsymbol{\mu}_{ineq,k}$ in order to determine j_{min}. The substitution of (4.63) in (4.61a) followed by the left multiplication of the obtained equation by $\mathbf{Y}_k^T$ lead to

$$\boldsymbol{\tau}_{\mathbf{y}} + \mathbf{Y}_k^T \mathbf{A}_{eq,k}^T \boldsymbol{\mu}_k = \mathbf{0}, \tag{4.77}$$

and reminding that, as in (4.65)-(4.66), $\mathbf{A}_{eq,k}\,\mathbf{Y}_k = \mathbf{R}_k^T$, (4.77) implies that

$$\mathbf{R}_k\,\boldsymbol{\mu}_k = -\boldsymbol{\tau}_{\mathbf{y}}. \tag{4.78}$$

Vector $\boldsymbol{\mu}_k = \begin{bmatrix}\boldsymbol{\mu}_{eq,k}^T & \boldsymbol{\mu}_{ineq,k}^T\end{bmatrix}^T$ can be computed performing back-substitution of (4.78). Moreover, whereas the vector $\boldsymbol{\mu}_{ineq,k}$ is necessary to decide the set $\mathcal{A}_{k+1}$, vector $\boldsymbol{\mu}_{eq,k}$ is unimportant in the numerical algorithm. Therefore, one may perform the back-substitution only for the last $n_{a,k}$ lines of the system (4.78) (the lines corresponding to $\boldsymbol{\mu}_{ineq,k}$).

Summarizing the handling of the active set for two subsequent iterations (k and $k+1$) with non-optimal $\boldsymbol{\tau}_{k+1}$, three actions may be taken. Namely,

- If the solution $\boldsymbol{\tau}_k^*$ (with $\boldsymbol{\tau}_k^* \neq \boldsymbol{\tau}_k$) is feasible, then $\mathcal{A}_{k+1} = \mathcal{A}_k$;
- If $\boldsymbol{\tau}_k^* \notin \mathbb{U}$, the blocking constraint j_b is added such that $\mathcal{A}_{k+1} = \mathcal{A}_k \cup \{j_b\}$;
- If $\boldsymbol{\tau}_k^* = \boldsymbol{\tau}_k$, then j_{min} is dropped such that $\mathcal{A}_{k+1} = \mathcal{A}_k \backslash \{j_{min}\}$.

The Algorithm 2 summarizes the proposed procedure to compute the estimation of desired cable tensions. Algorithm 2 is equivalent to the active set method proposed in [99, Section 16.5]. This algorithm identifies the global minimum of a strictly convex problem in a finite number of iterations. Considering the particular form of (4.59), some adaptations with respect to [99, Algorithm 16.3] have been made in order to improve the computation time.

Indeed, regarding the complexity of implementation, solutions simpler than Algorithm 2 could be obtained. Besides the explicit solution (4.67), alternative simpler methods such as the projected gradient method could lead to good convergence. Nevertheless, we believe that the proposed method presents a reduced computation time. A rigorous evaluation of different numerical solutions of (4.59) may be the subject of future studies.

For each iteration of states 6 to 29 in Algorithm 2, the computation time related to the determination of $\boldsymbol{\tau}_k^*$, $\mathbf{q}$ and $\boldsymbol{\mu}_{ineq,k}$ is small. On the other hand, the QR decomposition necessary to the computation of $\mathbf{Y}_k$, $\mathbf{Z}_k$ and $\mathbf{R}_k$ may lead to significant computational burden. To lower this burden, the fact that at most one inequality is added or suppressed from two successive $\mathcal{A}_k$ and $\mathcal{A}_{k+1}$ may be used so that $\mathbf{Y}_k$, $\mathbf{Z}_k$ and $\mathbf{R}_k$ do not need to be computed from scratch.

The computation of $\mathbf{Y}_1$, $\mathbf{Z}_1$ and $\mathbf{R}_1$ is performed with Householder transformations, as proposed in [176, Section 2.10] and detailed in [176, Section 11.2]. Appendix A

Algorithm 2 Computation of $\tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau})$

Inputs: $\mathbf{x}$, $\boldsymbol{\tau}_{in}$.
Internal variables: $\boldsymbol{\tau}_{min}$, $\boldsymbol{\tau}_{max}$.
Output: $\tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau}_{in})$.

1: **function** $\tilde{\boldsymbol{\tau}}_r(\mathbf{x}, \boldsymbol{\tau}_{in})$
2: $k \leftarrow 1$;
3: $\boldsymbol{\tau}_1 \leftarrow \boldsymbol{\tau}_{in}$;
4: Set $\mathcal{A}_1$ as the lines of $\mathbf{A}_{ineq}\,\boldsymbol{\tau}_1 = \mathbf{b}_{ineq}$ that are satisfied;
5: Set $\mathcal{I}_1$ according to (4.73);
6: **for** $k = 1, 2, ...$ **do**
7: Compute $\mathbf{Y}_k$, $\mathbf{R}_k$ and $\mathbf{Z}_k$ that satisfy (4.62);
8: Compute $\boldsymbol{\tau}_{\mathbf{y}}$ with back-substitution in (4.66);
9: $\boldsymbol{\tau}_k^* \leftarrow \mathbf{Y}_k\,\boldsymbol{\tau}_{\mathbf{y}}$ and $\mathbf{q} \leftarrow \boldsymbol{\tau}_k^* - \boldsymbol{\tau}_k$;
10: **if** $\mathbf{q} = \mathbf{0}$ **then**
11: Compute $\boldsymbol{\mu}_{ineq,k}$ with back-substitution in (4.78);
12: **if** $\boldsymbol{\mu}_{ineq,k} \geqslant \mathbf{0}$ **then**
13: **stop**
14: **else**
15: Determine index j_{min} of (4.59c) leading to minimal element of $\boldsymbol{\mu}_{ineq,k}$;
16: $\mathcal{A}_{k+1} \leftarrow \mathcal{A}_k \backslash \{j_{min}\}$;
17: $\boldsymbol{\tau}_{k+1} \leftarrow \boldsymbol{\tau}_k$;
18: **end if**
19: **else**
20: Compute α_k according to (4.72);
21: **if** there is a blocking inequality **then**
22: Determine the index j_b of (4.59c) that leads to the blocking constraint;
23: $\mathcal{A}_{k+1} \leftarrow \mathcal{A}_k \cup \{j_b\}$;
24: **else**
25: $\mathcal{A}_{k+1} \leftarrow \mathcal{A}_k$;
26: **end if**
27: $\boldsymbol{\tau}_{k+1} \leftarrow \boldsymbol{\tau}_k + \alpha_k\,\mathbf{q}$;
28: **end if**
29: **end for**
30: **return** $\boldsymbol{\tau}_k$;
31: **end function**

briefly discusses this subject. Subsequently, for $k > 1$, $\mathbf{Y}_{k-1}$, $\mathbf{Z}_{k-1}$ and $\mathbf{R}_{k-1}$ are used to obtain their updated counterparts ($\mathbf{Y}_k$, $\mathbf{Z}_k$ and $\mathbf{R}_k$). These matrices need to be updated in two possible scenarios: (i) one equality is added to the active set (step 23 in Algorithm 2) or (ii) one equality is removed (step 16 in Algorithm 2).

Consider firstly case (i). In the iteration k, an inequality $j_b \leqslant m$ (an analogous procedure is used for cases with $m < j_b \leqslant 2\,m$) was added such that

$$\mathbf{A}_{eq,k+1}^T = \begin{bmatrix} \mathbf{A}_{eq,k}^T & \mathbf{e}_{m,j_b} \end{bmatrix}, \tag{4.79}$$

where $\mathbf{e}_{m,j_b} \in \mathbb{R}^m$ given by $\mathbf{e}_{m,j_b} = \begin{bmatrix} 0 & \dots & 1 & \dots & 0 \end{bmatrix}^T$ with the j_b^{th} element equal to 1

and the remaining equal 0.

Since the matrix $\begin{bmatrix}\mathbf{Y}_k & \mathbf{Z}_k\end{bmatrix}$ is orthonormal, $\mathbf{Y}_k\,\mathbf{Y}_k^T + \mathbf{Z}_k\,\mathbf{Z}_k^T = \mathbf{I}$ and (4.79) may be written as

$$\mathbf{A}_{eq,k+1}^T = \begin{bmatrix}\mathbf{Y}_k & \mathbf{Z}_k\end{bmatrix} \begin{bmatrix}\mathbf{R}_k & \mathbf{Y}_k^T\,\mathbf{e}_{m,j_b} \\ \mathbf{0} & \mathbf{Z}^T\,\mathbf{e}_{m,j_b}\end{bmatrix}. \tag{4.80}$$

Denoting the row vectors of $\mathbf{Y}_k$ and $\mathbf{Z}_k$ with $\mathbf{Y}_k = \begin{bmatrix}\mathbf{u}_1^T & \dots & \mathbf{u}_m^T\end{bmatrix}^T$ and $\mathbf{Z}_k = \begin{bmatrix}\mathbf{v}_1^T & \dots & \mathbf{v}_m^T\end{bmatrix}^T$, the equation above may be rewritten as

$$\mathbf{A}_{eq,k+1}^T = \begin{bmatrix}\mathbf{Y}_k & \mathbf{Z}_k\end{bmatrix} \begin{bmatrix}\mathbf{R}_k & \mathbf{u}_{j_b} \\ \mathbf{0} & \mathbf{v}_{j_b}\end{bmatrix}. \tag{4.81}$$

It is possible to define an orthonormal $\hat{\mathbf{Q}}$ such that $\mathbf{v}_r = \hat{\mathbf{Q}}\,\mathbf{v}_{j_b}$ has all its elements equal to zero except for the first element. More precisely,

$$\hat{\mathbf{Q}}\,\mathbf{v}_{j_b} = \begin{bmatrix}\|\mathbf{v}_{j_b}\| \\ \mathbf{0}\end{bmatrix} = \mathbf{v}_r. \tag{4.82}$$

Denoting $v_b = \|\mathbf{v}_{j_b}\|$, (4.82) may be rewritten as $\mathbf{v}_{j_b} = \hat{\mathbf{Q}}^T \begin{bmatrix}v_b & \mathbf{0}\end{bmatrix}^T$ and (4.81) may be decomposed as follows

$$\mathbf{A}_{eq,k+1}^T = \underbrace{\begin{bmatrix}\mathbf{Y}_k & \mathbf{Z}_k\end{bmatrix} \begin{bmatrix}\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \hat{\mathbf{Q}}^T\end{bmatrix}}_{\mathbf{Q}'} \underbrace{\begin{bmatrix}\mathbf{R}_k & \mathbf{u}_{j_b} \\ \mathbf{0} & \mathbf{v}_r\end{bmatrix}}_{\mathbf{R}'} = \mathbf{Q}'\,\mathbf{R}' \tag{4.83}$$

Noting that $\mathbf{Q}'$ is orthonormal and $\mathbf{R}'$ is square and upper triangular, expression (4.83) may be rewritten as

$$\mathbf{A}_{eq,k+1}^T = \begin{bmatrix}\mathbf{Y}_{k+1} & \mathbf{Z}_{k+1}\end{bmatrix} \begin{bmatrix}\mathbf{R}_{k+1} \\ \mathbf{0}\end{bmatrix}, \tag{4.84}$$

which determines the updated $\mathbf{Y}_{k+1}$, $\mathbf{Z}_{k+1}$ and $\mathbf{R}_{k+1}$.

In case (ii), an equality is removed from the active set. The matrix $\mathbf{A}_{eq,k+1}^T$ is obtained removing one column, say column j_r, of $\mathbf{A}_{eq,k}^T$. As a consequence, the corresponding

column of $\mathbf{R}_k$ is also removed and $\mathbf{A}_{eq,k+1}^T$ is given by

$$\mathbf{A}_{eq,k+1}^T = \begin{bmatrix} \mathbf{Y}_k & \mathbf{Z}_k \end{bmatrix} \underbrace{\begin{bmatrix} r_{1,1} & r_{1,2} & \dots & r_{1,j_r-1} & r_{1,j_r+1} & \dots & r_{1,n_{e,k}} \\ 0 & r_{2,2} & \dots & r_{2,j_r-1} & r_{2,j_r+1} & \dots & r_{2,n_{e,k}} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & r_{j_r,j_r-1} & r_{j_r,j_r+1} & \dots & \vdots \\ 0 & 0 & \dots & 0 & r_{j_r+1,j_r+1} & \dots & \vdots \\ 0 & 0 & \dots & 0 & r_{j_r+2,j_r+1} & \dots & \vdots \\ 0 & 0 & \dots & 0 & 0 & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & 0 & r_{n_{e,k},n_{e,k}} \\ 0 & 0 & \dots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}}_{\tilde{\mathbf{R}}}, \tag{4.85}$$

where $r_{i,j}$ is the element in the i^{th} row and j^{th} column of $\mathbf{R}_k$.

Even though $\tilde{\mathbf{R}}$ is not upper triangular, the set of Householder transformations necessary to render $\tilde{\mathbf{R}}$ upper rectangular is cheaper than the QR factorization of $\mathbf{A}_{eq,k+1}^T$ computed from scratch. Hence, an orthonormal $\tilde{\mathbf{Q}}$ can be determined by means of Householder transformations such that

$$\tilde{\mathbf{R}} = \tilde{\mathbf{Q}} \begin{bmatrix} \mathbf{R}_{k+1} \\ \mathbf{0} \end{bmatrix}, \tag{4.86}$$

with upper triangular $\mathbf{R}_{k+1}$. Then, (4.85) may be expressed as

$$\mathbf{A}_{eq,k+1}^T = \underbrace{\begin{bmatrix} \mathbf{Y}_k & \mathbf{Z}_k \end{bmatrix} \tilde{\mathbf{Q}}}_{\mathbf{Q}''} \underbrace{\begin{bmatrix} \mathbf{R}_{k+1} \\ \mathbf{0} \end{bmatrix}}_{\mathbf{R}''} = \mathbf{Q}'' \, \mathbf{R}'', \tag{4.87}$$

which defines $\mathbf{Y}_{k+1}$, $\mathbf{Z}_{k+1}$ and $\mathbf{R}_{k+1}$.

4.3.3 OCP numerical solution

Section 4.2 presented an NMPC scheme considering the existence of a numerical algorithm able to solve the corresponding OCP. The present section introduces a solution using the numerical integration discussed in subsection 4.3.1.

4.3.3.1 NLP Formulation

The discretization (4.55) should be integrated within the OCP formulation so that, for given actual state $\mathbf{y}_a \in \mathbb{Y}$ and sequence of cable tensions $\boldsymbol{\tau}(\cdot) \in \mathbb{S}_{h_p}^m$, the states $\mathbf{y}_{\boldsymbol{\tau}(\cdot)}(i, \mathbf{y}_a)$ can be explicitly considered within the cost function of an NLP. This subject

is addressed in detail in [13, Chapter 11]. A straightforward method is to add the sequence $\tilde{\mathbf{y}} \in \mathbb{S}_{h_p}^{2n}$ of states within the prediction horizon as argument of the NLP and consider the transition mapping (4.55) as equality constraints. This approach leads to the following optimization problem

$$\min_{\substack{\boldsymbol{\tau}(\cdot)\in\mathbb{S}_{h_p}^{m} \\ \tilde{\mathbf{y}}(\cdot)\in\mathbb{S}_{h_p}^{2n}}} \sum_{j}^{h_p} \ell\Big(j+k, \tilde{\mathbf{y}}_j, \boldsymbol{\tau}_j\Big) \tag{4.88a}$$

$$\text{s.t. } \tilde{\mathbf{y}}_0 = \mathbf{y}_a \tag{4.88b}$$

$$\tilde{\mathbf{y}}_j = \tilde{\boldsymbol{\phi}}(\tilde{\mathbf{y}}_{j-1}, \boldsymbol{\tau}_j),\ \forall\, j \in \mathbb{N}_{1,h_p} \tag{4.88c}$$

$$\tilde{\mathbf{y}}_j \in \mathbb{Y},\ \forall\, j \in \mathbb{N}_{1,h_p} \tag{4.88d}$$

$$\boldsymbol{\tau}_j \in \mathbb{U},\ \forall\, j \in \mathbb{N}_{1,h_p} \tag{4.88e}$$

Even though (4.88) is a standard NLP problem, its solution is hindered by the increased number of arguments and constraints.

In order to overcome this drawback, the expression (4.55) can be applied recursively (as in (4.7)) so that states $\mathbf{y}_{\boldsymbol{\tau}(\cdot)}(i, \mathbf{y}_a)$ can be explicitly written *in function* of the actual state $\mathbf{y}_a$ and the sequence $\boldsymbol{\tau}(\cdot)$. More precisely, the recursive application of (4.55) leads to

$$\begin{aligned} &\tilde{\mathbf{y}}_0 = \mathbf{y}_a, \\ &\tilde{\mathbf{y}}_j = \mathbf{y}_{\boldsymbol{\tau}(\cdot)}(j, \mathbf{y}_a) = \mathbf{A}^j\, \mathbf{y}_a + \sum_{i=1}^{j} \Big\{ \mathbf{A}^{j-i}\Big(\mathbf{B}(\tilde{\mathbf{y}}_{j-1})\, \boldsymbol{\tau}_i + \mathbf{v}(\tilde{\mathbf{y}}_{j-1})\Big)\Big\},\ \forall\, j \in \mathbb{N}_{1,h_p}. \end{aligned} \tag{4.89}$$

Denoting $\mathbf{u} = \begin{bmatrix} \boldsymbol{\tau}_1^T & \dots & \boldsymbol{\tau}_{h_p}^T \end{bmatrix}^T$ and $\boldsymbol{\gamma} = \begin{bmatrix} \tilde{\mathbf{y}}_1^T & \dots & \tilde{\mathbf{y}}_{h_p}^T \end{bmatrix}^T$, (4.89) may be written as

$$\boldsymbol{\gamma} = \mathbf{E}\,\mathbf{u} + \boldsymbol{\Psi}, \tag{4.90}$$

where, $\mathbf{E}$ and $\boldsymbol{\Psi}$ are given by

$$\boldsymbol{\Psi} = \begin{bmatrix} \boldsymbol{\psi}_1^T & \boldsymbol{\psi}_2^T & \dots & \boldsymbol{\psi}_{h_p}^T \end{bmatrix}^T \tag{4.91a}$$

$$\mathbf{E} = \begin{bmatrix} \mathbf{E}'_{1,1} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{E}'_{2,1} & \mathbf{E}'_{2,2} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{E}'_{3,1} & \mathbf{E}'_{3,2} & \mathbf{E}'_{3,3} & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{E}'_{h_p,1} & \mathbf{E}'_{h_p,2} & \dots & \dots & \mathbf{E}'_{h_p,h_p} \end{bmatrix}, \tag{4.91b}$$

with

$$\begin{aligned} \boldsymbol{\psi}_j &= \mathbf{A}^j \mathbf{y}_a + \sum_{i=1}^{j} \mathbf{A}^{j-i}\, \mathbf{v}(\tilde{\mathbf{y}}_{i-1}) \\ \mathbf{E}'_{j,i} &= \mathbf{A}^{j-i}\, \mathbf{B}(\tilde{\mathbf{y}}_{j-1}). \end{aligned} \tag{4.91c}$$

Similarly, for given instant $k \in \mathbb{N}$, cable tension sequence $\boldsymbol{\tau}(\cdot) \in \mathbb{S}_{h_p}^{m}$ and states $\tilde{\mathbf{y}} \in \mathbb{S}_{h_p}^{2n}$, vectors $\boldsymbol{\gamma}_{d,k} \in \mathbb{R}^{2n\,h_p}$ and $\tilde{\mathbf{u}}_d \in \mathbb{R}^{m\,h_p}$ are defined as

$$\boldsymbol{\gamma}_{d,k} = \begin{bmatrix} \mathbf{y}_d(k+1)^T & \dots & \mathbf{y}_d(k+h_p)^T \end{bmatrix}^T \tag{4.92}$$

$$\tilde{\mathbf{u}}_d = \begin{bmatrix} \tilde{\boldsymbol{\tau}}_r(\mathbf{x}_{\tilde{\mathbf{y}}_1}, \boldsymbol{\tau}_1)^T & \dots & \tilde{\boldsymbol{\tau}}_r(\mathbf{x}_{\tilde{\mathbf{y}}_{h_p}}, \boldsymbol{\tau}_{h_p})^T \end{bmatrix}^T. \tag{4.93}$$

Considering the definition of stage cost (4.12), the cost functional in (4.14) may be written as

$$\begin{aligned}\sum_{i=1}^{h_p} \ell\Big(i+k, &\mathbf{y}_{\boldsymbol{\tau}(\cdot)}\left(i, \mathbf{y}_a\right), \boldsymbol{\tau}_i\Big) = \\ &= \|\boldsymbol{\gamma} - \boldsymbol{\gamma}_{d,k}\|^2_{\mathbf{K}_\gamma} + \|\tilde{\mathbf{u}}_d - \mathbf{u}\|^2_{\mathbf{K}_\mathbf{u}} \\ &= \left(\mathbf{E}\,\mathbf{u} + \boldsymbol{\Psi} - \boldsymbol{\gamma}_{d,k}\right)^T \mathbf{K}_\gamma (\mathbf{E}\,\mathbf{u} + \boldsymbol{\Psi} - \boldsymbol{\gamma}_{d,k}) + \left(\mathbf{u} - \tilde{\mathbf{u}}_d\right)^T \mathbf{K}_\mathbf{u}\left(\mathbf{u} - \tilde{\mathbf{u}}_d\right) \\ &= \mathbf{u}^T\,\mathbf{H}_{\tilde{\mathbf{y}},k}\,\mathbf{u} + 2\,\mathbf{h}^T_{\tilde{\mathbf{y}},k}\,\mathbf{u} + c_\gamma,\end{aligned} \tag{4.94}$$

with

$$\begin{aligned}\mathbf{H}_{\tilde{\mathbf{y}},k} &= \mathbf{E}^T\,\mathbf{K}_\gamma\,\mathbf{E} + \mathbf{K}_\mathbf{u}, \\ \mathbf{h}_{\tilde{\mathbf{y}},k} &= \left(\boldsymbol{\Psi} - \boldsymbol{\gamma}_{d,k}\right)^T \mathbf{K}_\gamma\,\mathbf{E} - \tilde{\mathbf{u}}_d^T\,\mathbf{K}_\mathbf{u}, \\ c_\gamma &= \left(\boldsymbol{\Psi} - \boldsymbol{\gamma}_{d,k}\right)^T \mathbf{K}_\gamma\left(\boldsymbol{\Psi} - \boldsymbol{\gamma}_{d,k}\right) + \tilde{\mathbf{u}}_d^T \mathbf{K}_\mathbf{u}\,\tilde{\mathbf{u}}_d,\end{aligned} \tag{4.95}$$

and $\mathbf{K}_\gamma \in \mathbb{R}^{h_p 2n \times h_p 2n}$, $\mathbf{K}_\mathbf{u} \in \mathbb{R}^{h_p m \times h_p m}$ given by

$$\begin{aligned}\mathbf{K}_\gamma &= \operatorname{diag}\left(\begin{bmatrix}\mathbf{k}_\mathbf{y}^T & \mathbf{k}_\mathbf{y}^T & \dots & \mathbf{k}_\mathbf{y}^T\end{bmatrix}^T\right), \\ \mathbf{K}_\mathbf{u} &= \operatorname{diag}\left(\begin{bmatrix}\mathbf{k}_{\boldsymbol{\tau}}^T & \mathbf{k}_{\boldsymbol{\tau}}^T & \dots & \mathbf{k}_{\boldsymbol{\tau}}^T\end{bmatrix}^T\right).\end{aligned} \tag{4.96}$$

Therefore, for given $\mathbf{y}_a \in \mathbb{Y}$ and $k \in \mathbb{N}$, the OCP (4.14) may be expressed as

$$\mathbf{u}^* = \underset{\mathbf{u}\in\mathbb{R}^{m\,h_p}}{\arg\min}\ \mathbf{u}^T\,\mathbf{H}_{\tilde{\mathbf{y}},k}\,\mathbf{u} + 2\,\mathbf{h}^T_{\tilde{\mathbf{y}},k}\,\mathbf{u} \tag{4.97a}$$

$$\text{s. t. } \tilde{\mathbf{y}}_0 = \mathbf{y}_a \tag{4.97b}$$

$$\tilde{\mathbf{y}}_j = \tilde{\boldsymbol{\phi}}\Big(\tilde{\mathbf{y}}_{j-1}, \boldsymbol{\tau}_\mathbf{u}(j)\Big)\ \forall\, j \in \mathbb{N}_{1,h_p} \tag{4.97c}$$

$$\tilde{\mathbf{y}}_j \in \mathbb{Y}\ \forall\, j \in \mathbb{N}_{1,h_p} \tag{4.97d}$$

$$\boldsymbol{\tau}_\mathbf{u}(j) \in \mathbb{U}\ \forall\, j \in \mathbb{N}_{1,h_p} \tag{4.97e}$$

where $\boldsymbol{\tau}_\mathbf{u}(j)$ is obtained extracting the j^{th} vector with dimension m from $\mathbf{u}$, such that $\mathbf{u} = \begin{bmatrix}\boldsymbol{\tau}_\mathbf{u}(1)^T & \dots & \boldsymbol{\tau}_\mathbf{u}(h_p)^T\end{bmatrix}^T$. In spite of the similarity between (4.88) and (4.97), these two NLPs present two crucial differences.

Optimization problem (4.97) has a reduced set of arguments ($\tilde{\mathbf{y}}(\cdot)$ is not an argument). More precisely, (4.88) has $(m+n)\,h_p$ arguments while (4.97) has $m\,h_p$.

Moreover, the constraint (4.88c) demands particular attention in the numerical solution of (4.88) because it implies that arguments $\boldsymbol{\tau}(\cdot)$ and $\tilde{\mathbf{y}}(\cdot)$ are consistent with respect to the transition mapping $\tilde{\boldsymbol{\phi}}$. In contrast, constraint (4.97c) is implicitly satisfied. Since the sequence $\tilde{\mathbf{y}}(\cdot)$ is not an argument of (4.97), each vector $\tilde{\mathbf{y}}_j\ \forall\, j \in \mathbb{N}_{1,h_p}$ is attributed according to (4.97c) and, thereby, satisfy these equality constraints. The sequence $\tilde{\mathbf{y}}(\cdot)$ in (4.97) is nothing more than an internal variable (neither output nor argument of the optimization problem) used to compute the matrices $\mathbf{H}_{\tilde{\mathbf{y}},k}$, $\mathbf{h}_{\tilde{\mathbf{y}},k}$ and to check the constraint (4.97d).

Optimization problem (4.97) represents a standard NLP, which can be solved with standard numerical optimization libraries, *e.g.* [101, 188]. Note that matrices $\mathbf{H}_{\tilde{\mathbf{y}},k}$, $\mathbf{h}_{\tilde{\mathbf{y}},k}$

are dependent of $\tilde{\mathbf{y}}(\cdot)$ and, therefore, are not constant. An optimization problem taking as argument $\mathbf{u} \in \mathbb{R}^{m.h_p}$ can be classified as a QP problem only if its cost function can be written as $f(\mathbf{u}) = \mathbf{u}^T \mathbf{H}\, \mathbf{u} + 2\, \mathbf{h}^T \mathbf{u}$, with constant matrix $\mathbf{H}$ and vector $\mathbf{h}$. Hence, (4.97) is not a QP problem.

4.3.3.2 Numerical Solution of the NLP problem

Section 4.3.3.1 introduced a standard nonlinear optimization problem representing the OCP proposed in Section 4.2. Since the CDPR prototypes used in this thesis are not compatible with third party numerical libraries, the present section briefly describes the implemented numerical solution of (4.97). Results presented in Section 4.5 were obtained using this implementation.

The proposed algorithm is inspired by the SQP method. Nocedal and Wright [99, Chapter 18] discuss this class of numerical algorithm in detail. Grüne and Pannek [13, Chapter 12] study the applicability of such algorithms to NMPC solutions. These are iterative algorithms in which, at each iteration, the original nonlinear problem is approximated as a QP problem. Each of these sub-problems may be solved with a standard QP algorithm. An SQP approach is considered appropriate for the solution of (4.97) since the proposed OCP possesses a quadratic cost function.

As discussed in [99, Chapter 6], the computation of the second derivatives of a nonlinear cost function is often computationally expensive. For this reason, the quadratic approximation taken for each iteration of an SQP method commonly considers an estimation of the exact hessian matrix. Typically, this approximation is done with quasi-Newton algorithms such as the BFGS or SR1 methods [99, Sections 6.1 and 6.2]. Indeed, the implemented algorithm uses the BFGS method.

Quasi-Newton algorithms determine approximations of the Hessian matrix of a given cost function considering its first derivatives. The direct application of such methods for (4.97) may be possible but would lead to a troublesome implementation. The first derivatives of the cost function in (4.97a) should take into account the variations of $\tilde{\mathbf{u}}_d$ with respect to $\mathbf{u}$ according to (4.93). The variations of $\tilde{\boldsymbol{\tau}}_r$ with respect to $\mathbf{x} \in \mathbb{X}$ and $\boldsymbol{\tau}_{in} \in \mathbb{U}$ in (4.13) may be not continuous. For this reason, (4.97) is solved iteratively considering, for each iteration, a constant vector $\hat{\mathbf{u}}_d = \tilde{\mathbf{u}}_d$, as described in the following.

Figure 4.6 summarizes the implemented numerical algorithm. Broadly speaking, the algorithm solves iteratively the original NLP problem applying, for each iteration, a standard SQP method for a simpler version of (4.97), which takes a constant $\hat{\mathbf{u}}_d$ instead of $\tilde{\mathbf{u}}_d$.

More in detail, for given actual state $\mathbf{y}_a$ and initial guess $\mathbf{u}_0$, the vectors $\boldsymbol{\gamma}_0$ and $\tilde{\mathbf{u}}_{d,0}$ can be computed using (4.90) and (4.93), respectively. An approximation of the NLP (4.97) is constructed replacing $\tilde{\mathbf{u}}_d$ by a constant $\hat{\mathbf{u}}_d$ in (4.94) and (4.95). The resulting cost function possesses continuous first derivatives that can be computed numerically or analytically. Results presented in Section 4.5 were obtained with first derivatives computed analytically, which were defined with tedious (but straightforward) calcu-

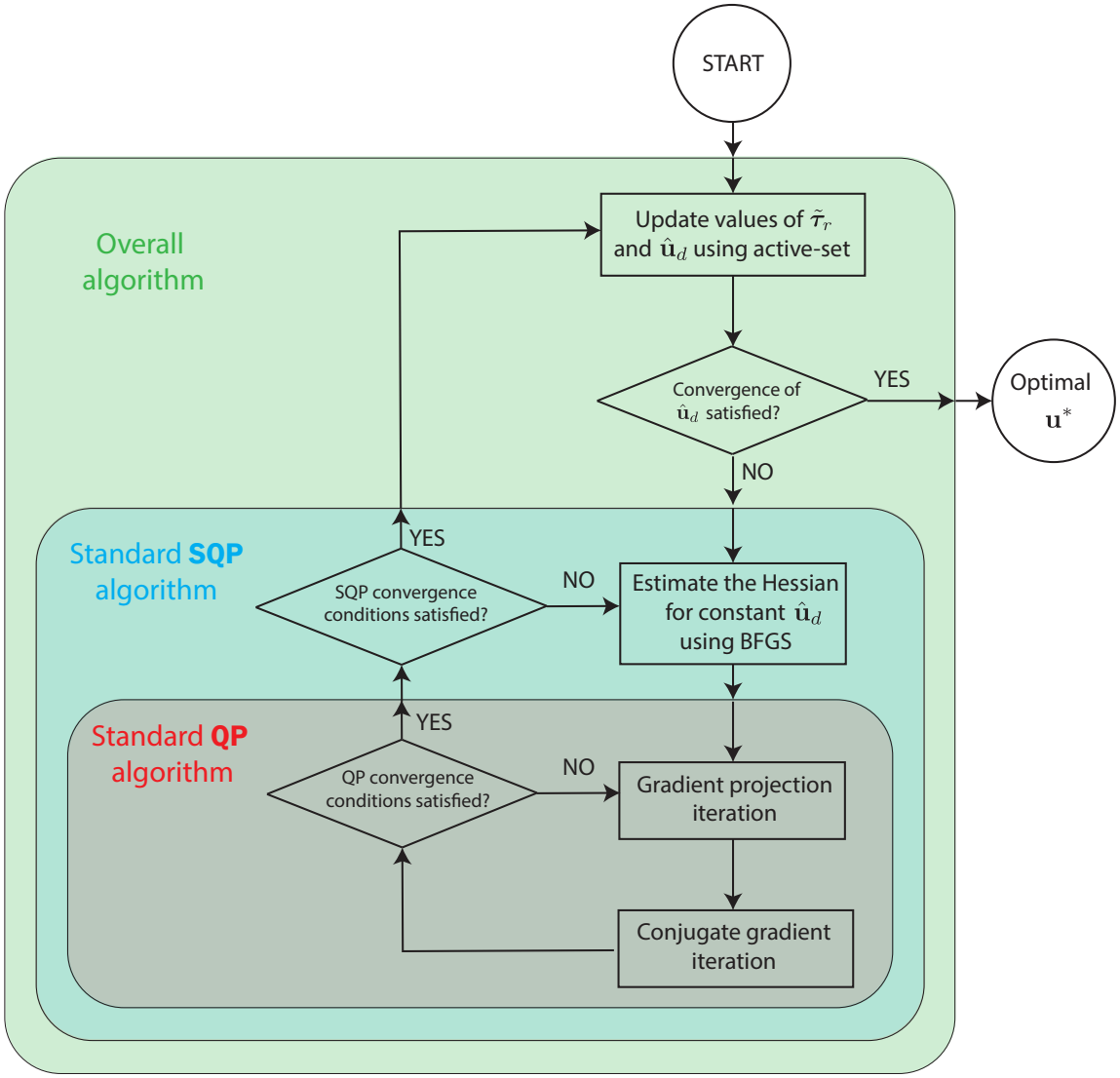

Figure 4.6 – *Summary of the implemented numerical solution of* (4.97).

lations. Based on these derivatives, a standard SQP method can be used, leading to a sequence of cable tensions represented by $\mathbf{u}_1$. This new vector is used to obtain updated γ_1 and $\tilde{\mathbf{u}}_{d,1}$. The procedure is repeated until the convergence of $\hat{\mathbf{u}}_d$ is satisfied.

As described above, for each iteration of the overall algorithm, an SQP algorithm takes place. As indicated by its name, a sequential quadratic programming method solves a sequence of subproblems consisting of QP problems. The Hessian matrix of each one of these QP problems is estimated using the BFGS method. The quadratic programming problems themselves should be solved with a given QP algorithm.

The implemented QP algorithm consists of a combination of the gradient projection and conjugate gradient methods. The gradient projection method (in contrast to the active-set method used for the LMPC in Chapter 3) is known to be particularly suitable for QP problems with box constraints, which is the case of the constraint set $\mathbb{U}$. Indeed, if the state constraint set is given by

$$\mathbb{Y} = \mathbb{R}^{2n}, \tag{4.98}$$

constraints (4.97d) can be neglected and the optimization problem (4.97) is constrained uniquely by the box constraints (4.97e). Section 4.4 discusses that assumption (4.98) may be typically considered. Therefore, Sections 4.4 and 4.5 consider $\mathbb{Y} = \mathbb{R}^{2n}$.

As discussed in [99, Section 16.7], the convergence of the projected gradient method can be significantly improved if it is coupled with the conjugate gradient method ([99, Chapter 5]). For this reason, these two methods are combined in the proposed algorithm.

4.4 Numerical Simulations

This section presents numerical simulations comparing the linear MPC proposed in Chapter 3 with the strategy introduced in Section 4.2. Results show that the linear approximation considered in Chapter 3 may lead to poor performances when the system dynamics presents significant nonlinearities.

More precisely, for a given instant k, the influence of changes on $\mathbf{y}_{k+i}\ \forall\ i \in \mathbb{N}_{1,h_p}$ in the approximation (3.15) may deteriorate the performance obtained with the corresponding linear MPC. The error resulting from approximation (3.15) is increased for substantial variations of $\mathbf{B}(\mathbf{y}_{k+i})$ and $\mathbf{v}(\mathbf{y}_{k+i})$. In general, this issue is present for high velocities and accelerations. A crucial imprecision that may be obtained with the approximation (3.15) is that the wrench matrix $\mathbf{W}$ is considered constant within the prediction horizon. This matrix may change significantly for high velocities, leading to non-negligible variations of $\mathbf{B}(\mathbf{y}_{k+i})$. In contrast, this section shows that the proposed NMPC overcomes this drawback since it considers the original nonlinear model.

In order to stress the differences between these different MPC approaches, results presented in this section simulate the operation of the CDPR HRPCable (Figure 1.1), which can safely operate with relatively high velocities and accelerations in comparison to the Hephaestus prototype.

As for the previous chapter, a typical pick-and-place trajectory is used in order to represent the operation of the robot. Figure 4.8 illustrates the poses visited in this trajectory. The desired trajectory is generated using fifth degree polynomials in time relating two consecutive poses through a straight line path. Maximal linear velocity and acceleration are 0.7 m/s and 0.26 m/s^2.

It is interesting to note that the workspace dimensions should be considered when comparing the magnitude of developed velocities between different prototypes. Typically, velocities with similar order of magnitude may lead to higher variations of $\mathbf{W}$ (and $\mathbf{B}$, consequently) for CDPRs with smaller workspace such as illustrated in Figure 4.7. The same displacement is considered for two planar illustrative CDPRs. One may note that the variations on the cable orientation for the CDPR of Figure 4.7-(a) is greater than the one corresponding to Figure 4.7-(b). This leads to an increased variation on the wrench matrix. Commonly, for given operational space velocities, CDPRs with smaller workspace lead to increased variations of $\mathbf{W}$.

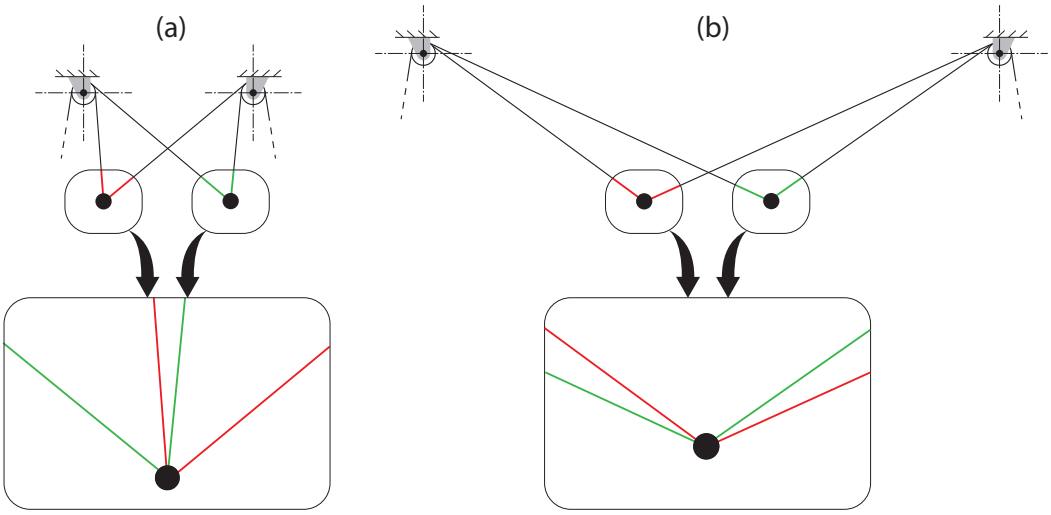

Figure 4.7 – *Comparison of the variation on the cable orientation for illustrative 2-DoF CDPRs with the same platform displacement.*

Both linear MPC (proposed Chapter 3) and NMPC (introduced in Section 4.2) were simulated using the trajectory illustrated in Figure 4.8 and cable tension limits $\boldsymbol{\tau}_{min} = 100$ N, $\boldsymbol{\tau}_{max} = 2000$ N. For the linear MPC, the following controller parameters have been used:

$$\mathbf{k}'_{\mathbf{x}} = 10^6 \begin{bmatrix} 1.4 & 2.2 & 8.8 & 7.7 & 4.4 & 7.6 \end{bmatrix}^T \tag{4.99a}$$

$$\mathbf{k}'_{\dot{\mathbf{x}}} = 10^6 \begin{bmatrix} 0.28 & 0.44 & 1.76 & 1.54 & 0.88 & 1.52 \end{bmatrix}^T \tag{4.99b}$$

$$\mathbf{K}'_u = k'_u \, \mathbf{I} = 1.0 \times 10^{-6} \, \mathbf{I} \tag{4.99c}$$

$$k'_{\Delta u} = 0.026. \tag{4.99d}$$

Gains (4.99) were obtained using a particle swarm optimization algorithm (built-in MATLAB function). More in detail, for a given set of controller gains, the considered objective function computes a weighted sum of

1. The maximal translation error;
2. The maximal orientation error;
3. The RMS value of the variation of cable tensions (computed according to Section 3.2).

Approximate values of the gains were tuned manually and used in order to define the lower and upper bounds for each gain being optimized. Sampling period and prediction horizon were defined as 10 ms and 12, respectively. Results are shown in Figure 4.9.

Due to the reasons discussed above, the linear MPC strategy leads to deteriorated results. The maximal tracking errors reaches almost 36 mm and 0.17°. This indicates that a nonlinear approach is necessary.

Similarly to the procedure used to define the gains of the linear MPC, a particle swarm algorithm was used in order to tune the NMPC gains. In this case, the considered objective function computes a weighed sum of the maximal translation and

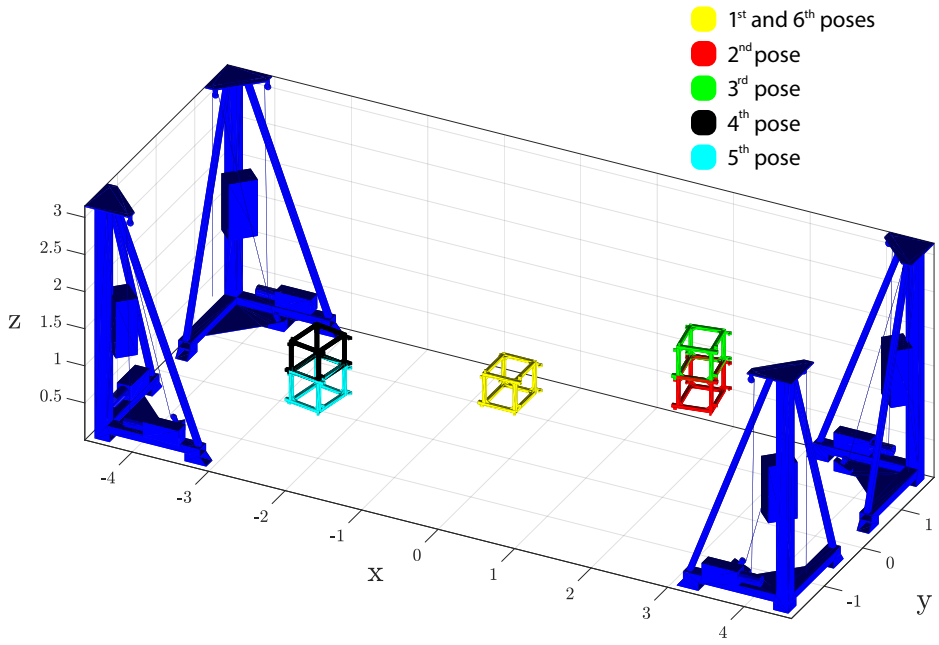

Figure 4.8 – *Visited poses of the simulated trajectory*

rotation errors. Since the NMPC scheme of Section 4.2 does not address the variations of cable tensions, the RMS value of the variation of cable tensions was not considered in the optimization of the NMPC gains. In contrast to the linear MPC, the proposed NMPC does not penalize the variation of cable tensions. The following gains were obtained:

$$\mathbf{k_x} = 10^4 \begin{bmatrix} 1.4 & 1.7 & 1.2 & 1.3 & 1.3 & 0.9 \end{bmatrix}^T \tag{4.100a}$$

$$\mathbf{k_{\dot{x}}} = 10^0 \begin{bmatrix} 10.8 & 13.1 & 9.1 & 10.1 & 9.7 & 7.0 \end{bmatrix}^T \tag{4.100b}$$

$$\mathbf{K_u} = k_\mathbf{u}\,\mathbf{I} = 5.8 \times 10^{-7}\,\mathbf{I}. \tag{4.100c}$$

The same sampling period is used (10 ms) and the prediction horizon is taken equal to 6. Figure 4.10 depicts the results obtained with the proposed minimal state NMPC.

The errors depicted in Figure 4.10 indicate that the proposed NMPC effectively tracks the desired trajectory even with the incidence of significant nonlinearities due to relatively high velocities and accelerations. As a matter of fact, the obtained errors are virtually null (less than 0.05 mm and 0.9×10^{-4} degrees), as expected for a scenario without the occurrence of disturbances or model uncertainties. It is also interesting to highlight that the prediction horizon used in the NMPC scheme is the half of its counterpart used in the linear MPC (6 versus 12). Moreover, increasing further the prediction horizon of the linear MPC controller does not lead to improved performance since the linear approximation is more susceptible to imprecision for longer prediction horizons. The considered value of 12 was the best compromise found.

As in Section 3.3, it is also pertinent to evaluate the ability of the proposed control scheme to perform the tracking of unfeasible desired trajectories. To this end, a shortened trajectory is studied. The first, second and third poses in Figure 4.8 were

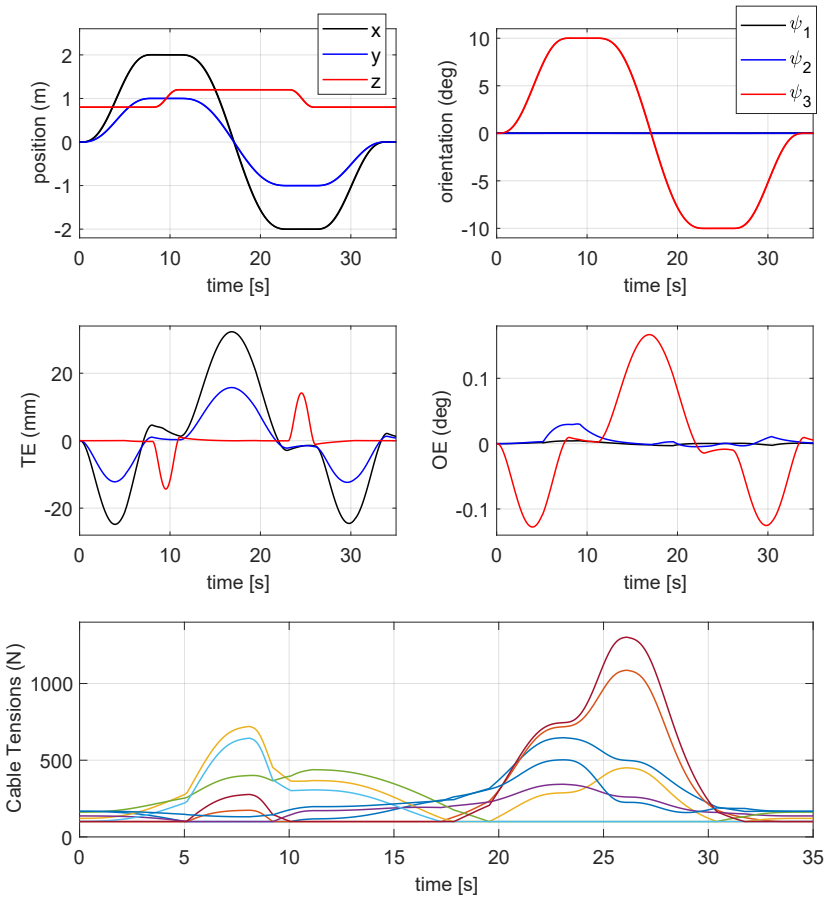

Figure 4.9 – *Simulation results obtained with the linear MPC proposed in Chapter 3.*

considered taking as cable tension limits $\boldsymbol{\tau}_{min} = 100$ N and $\boldsymbol{\tau}_{max} = 500$ N. Results are depicted in Figure 4.11.

As expected, the tracking of the trajectory can be performed without violating the cable tension limits. For time in $7.3 \leqslant t \leqslant 12.1$ (s) the tension of cable 3 is equal to $500 = \tau_{max,3}$ and the desired trajectory cannot be tracked without errors. Accordingly, tracking errors ranging up to 35 mm and 1.3° are obtained. Therefore, one may expect to extend the stability analysis results of Section 4.2.3 for cases in which the feasibility considered in Assumption 4.1 is not satisfied. This is, indeed, a subject currently under study.

It is worth noting that the results presented in this section consider the state constraint set equal to $\mathbb{Y} = \mathbb{R}^{2n}$. As a result, constraint (4.97d) does not affect the solution of (4.97). This indicates that, in practice, the robot workspace may be indirectly delimited by the constraint (4.97e) (cable tension limits). This exempts the control designer from the potentially arduous task of defining the set $\mathbb{Y}$. Moreover, the solution of (4.97) is also facilitated by the reduced number of constraints.

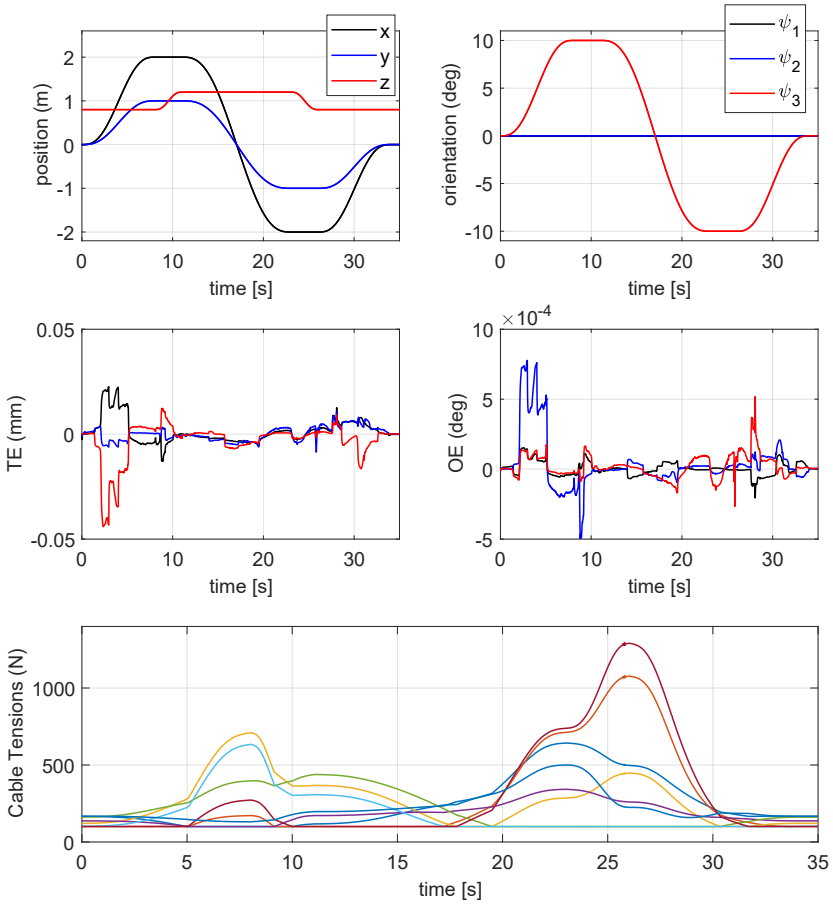

Figure 4.10 – *Simulation results obtained with the NMPC proposed in Section 4.2.*

4.5 Real-Time Experiments

The NMPC scheme proposed in Section 4.2 was implemented on the HRPCable prototype leading to the experimental results detailed in the present section. The performance of the LMPC and NMPC schemes (proposed respectively in Section 3.1 and 4.2) are compared using typical pick-and-place trajectories. These control schemes are also compared with respect to their robustness against payload uncertainties.

The control parameters were tuned by trial and error and are given as follows:

$$\mathbf{k_x} = \begin{bmatrix} 56 & 68 & 48 & 52 & 52 & 36 \end{bmatrix}^T \tag{4.101a}$$

$$\mathbf{k_{\dot{x}}} = \begin{bmatrix} 11 & 13 & 9 & 10 & 10 & 7 \end{bmatrix}^T \tag{4.101b}$$

$$\mathbf{K_u} = k_{\mathbf{u}}\,\mathbf{I} = 6 \times 10^{-7}\,\mathbf{I} \tag{4.101c}$$

$$h_p = 4. \tag{4.101d}$$

4.5.1 Pick-and-place trajectories

Typical pick-and-place trajectories were performed taking the visited poses shown in Figure 4.12. The desired trajectories are defined using 5^{th} degree polynomials based

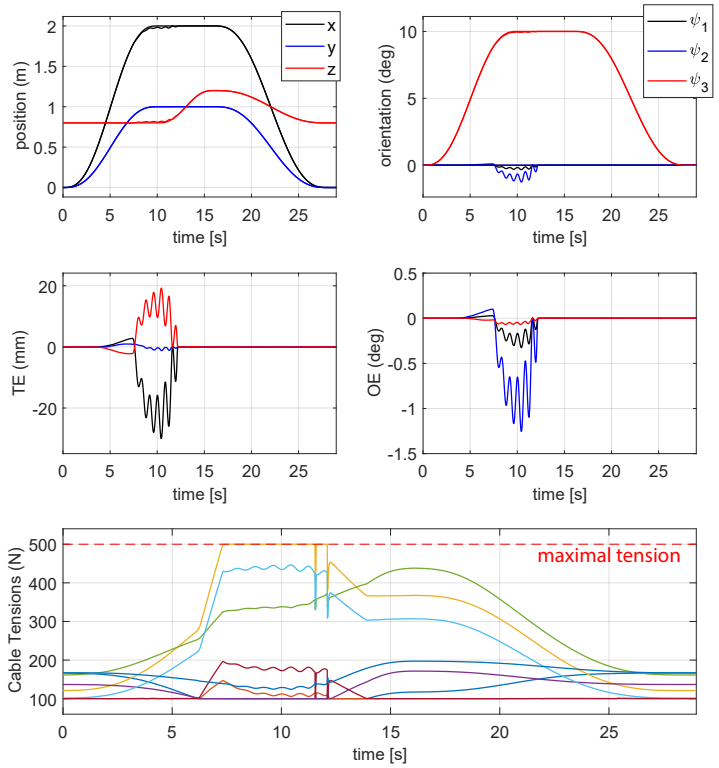

Figure 4.11 – *Application of the proposed NMPC with an unfeasible desired trajectory.*

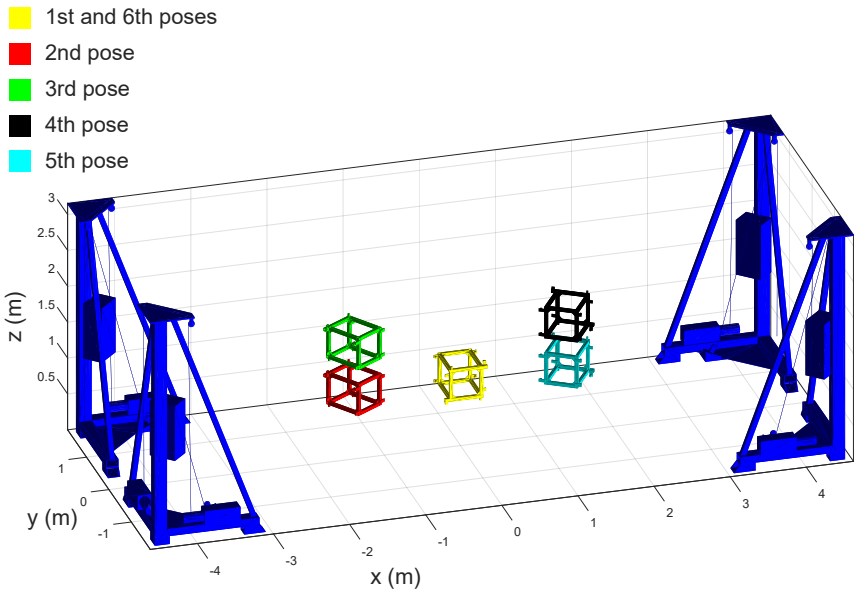

Figure 4.12 – *Visited poses for the real-time experiments.*

on these poses. Similarly to the experiments presented in Section 3.3, two scenarios of maximal cable tension are considered: (i) $\boldsymbol{\tau}_{max} = 400$ N and (ii) $\boldsymbol{\tau}_{max} = 250$ N. These scenarios are addressed respectively in Sections 4.5.1.1 and 4.5.1.2 and both cases consider $\boldsymbol{\tau}_{min} = 100$ N.

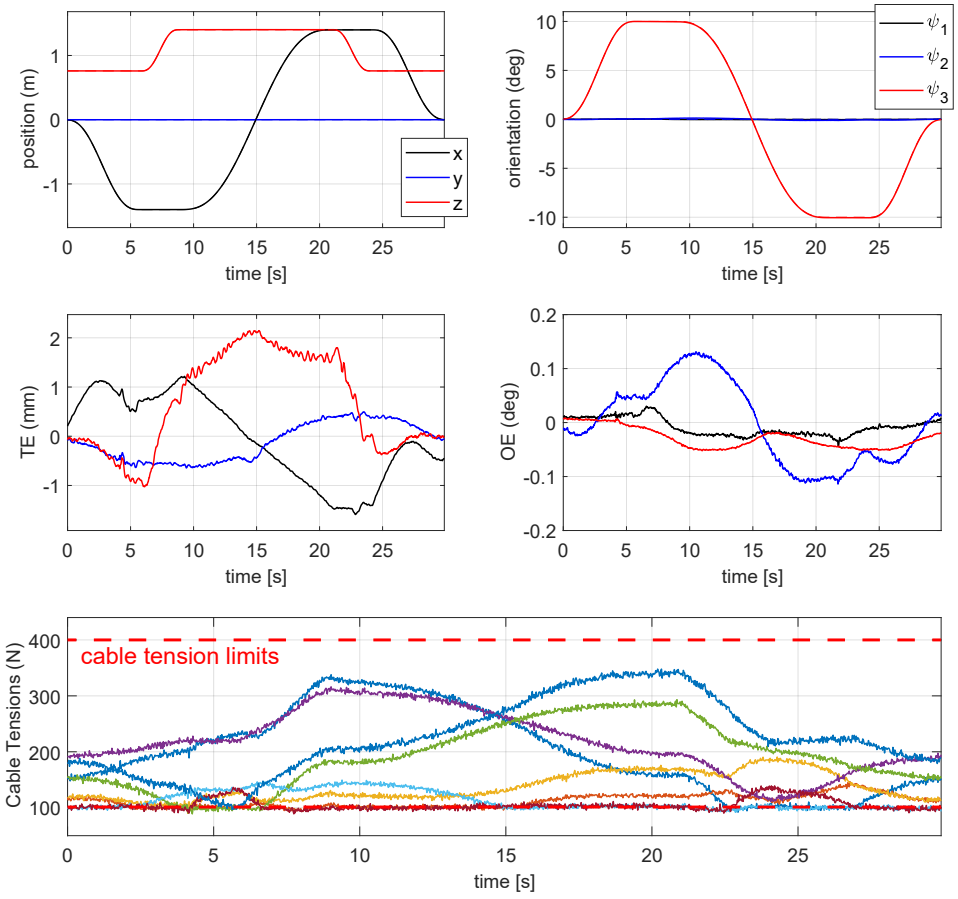

Figure 4.13 – *Experimental results for $\tau_{max} = 400$ N.*

Table 4.1 – *Summary of comparative errors between LMPC and NMPC.*

	RMS		Maximal error	
	TE	OE	TE	OE
LMPC	4.28 mm	0.15°	7.47 mm	0.27°
NMPC	1.52 mm	0.08°	2.34 mm	0.14°
Improvement	64.5%	44.6%	68.6%	48.1%

4.5.1.1 Nominal cable tension limits

Figure 4.13 depicts the experimental results obtained using the proposed NMPC scheme taking cable tension limits of $\tau_{min} = 100$ N and $\tau_{max} = 400$ N. The same trajectory was performed with the LMPC proposed in Section 3.1. Figure 4.14 compares the corresponding results.

In accordance with the goals defined in the present chapter, the comparative results presented in Figure 4.14 indicates that the proposed NMPC scheme leads to significantly better tracking errors than the LMPC approach. Indeed, Table 4.1 shows that significant improvements were obtained considering both the RMS of the tracking errors and the maximal errors.

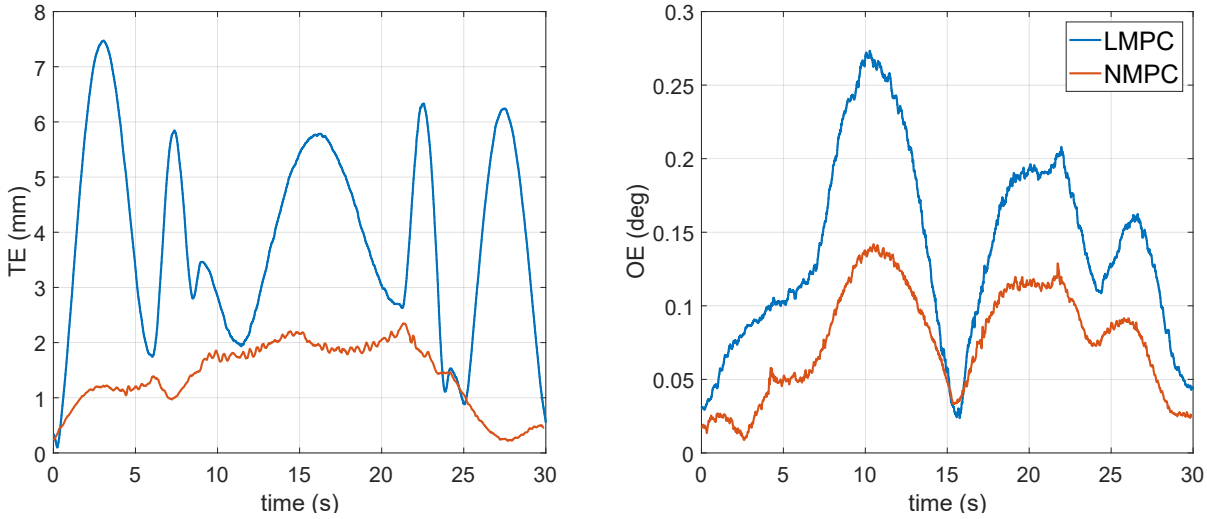

Figure 4.14 – *Tracking errors for LMPC and NMPC.*

4.5.1.2 Reduced maximal cable tension

Chapter 3 highlighted that the main contribution of the LMPC scheme proposed in Chapter 3 lies in its capability to handle cable tension limits explicitly. Thanks to this feature, this LMPC is not prone to the feasibility issues discussed in Section 1.7 and exemplified in Figure 3.9. In order to verify this property, an unfeasible desired trajectory is used in Section 3.3. The same behavior should be validated for the NMPC scheme proposed in the present chapter.

Therefore, as for Section 3.3.2.1, a second scenario is considered taking a maximal cable tension reduced to $\boldsymbol{\tau}_{max} = 250$ N. The obtained results are shown in Figure 4.15. The whole trajectory could be performed without interruption and the restricted cable tension limits were satisfied. Therefore, this experiment demonstrates that the proposed NMPC scheme is not prone to the feasibility issues depicted in Figure 3.9.

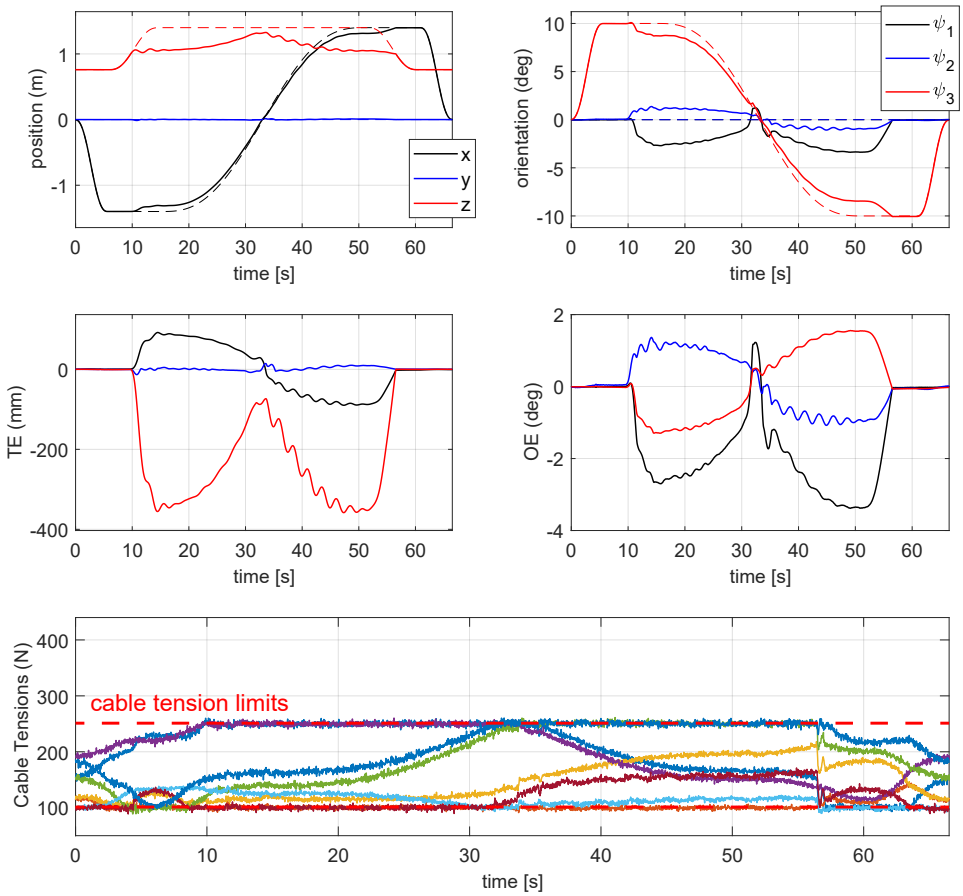

Figure 4.15 – *Experiments taking* $\tau_{max} = 250\ N$.

4.5.2 Robustness against payload uncertainties

In addition to the presented pick-and-place trajectories, the robustness against payload uncertainties of the proposed NMPC scheme was evaluated using the experiment illustrated in Figure 3.10. An additional payload of 15.1 kg was used, representing 65.6% of the platform mass. The obtained results are depicted on Figure 4.16. The same procedure was performed using the LMPC proposed in Chapter 3. Comparative results are presented in Figure 4.17. Table 4.2 shows that, as for the results presented in Section 4.5.1.1, significantly better tracking errors were obtained with the proposed NMPC scheme.

Table 4.2 – *Summary of comparative errors between LMPC and NMPC regarding the robustness against payload uncertainties.*

	RMS		Maximal error	
	TE	OE	TE	OE
LMPC	4.66 mm	0.52°	12.37 mm	1.05°
NMPC	3.92 mm	0.31°	7.83 mm	0.63°
Improvement	16.0%	40.7%	36.7%	40.0%

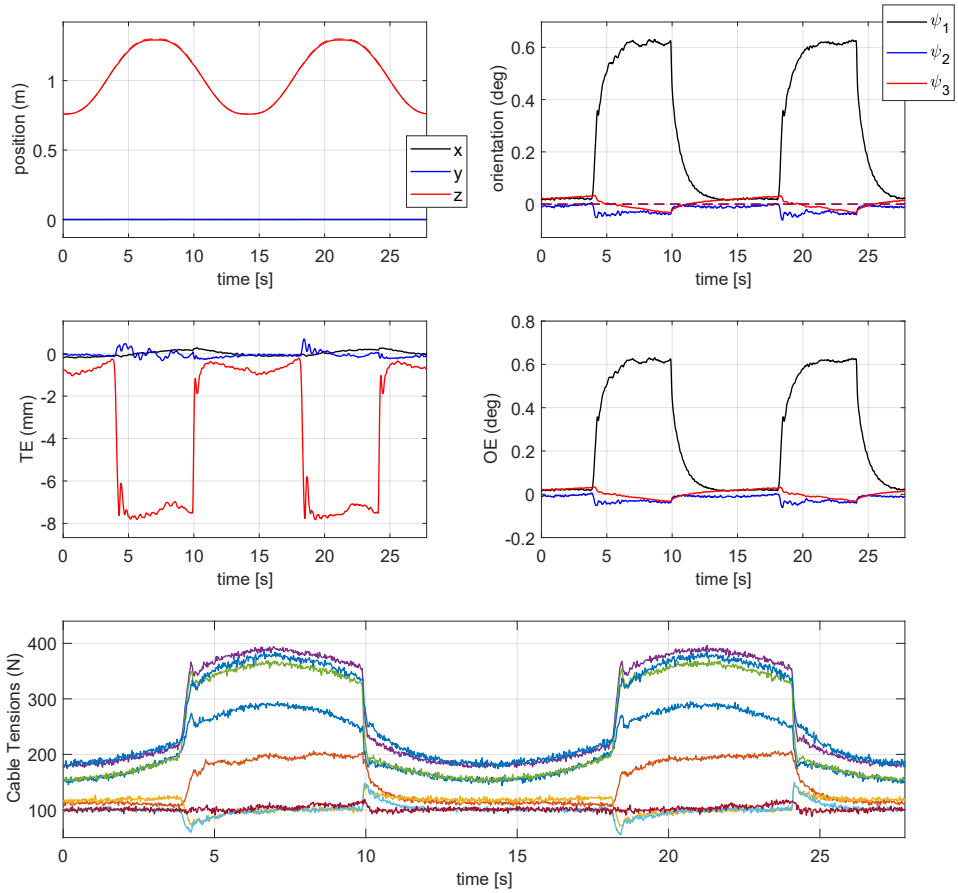

Figure 4.16 – *Experimental results obtained with the NMPC scheme addressing the robustness against payload uncertainties.*

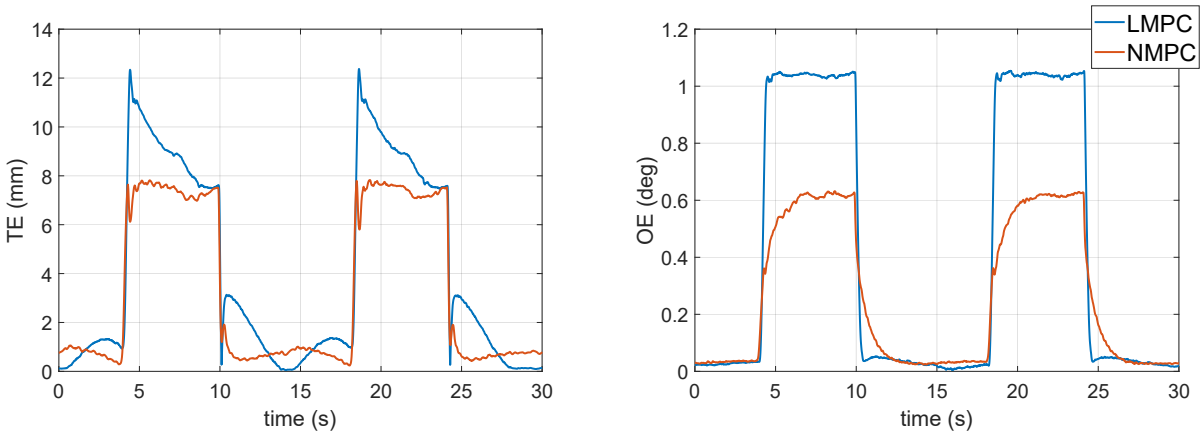

Figure 4.17 – *Tracking errors for the LMPC and NMPC schemes in the experiment testing the robustness against payload uncertainties.*

4.6 Conclusions

This chapter discussed the main drawbacks of the LMPC scheme proposed in Chapter 3. Namely, this approach hindered the stability analysis of the resulting closed-loop system and it may lead to deteriorated tracking errors. To overcome these drawbacks, an NMPC is introduced based on a nonlinear discrete time model and a new cost func-

tional formulation. The stability of the corresponding closed-loop system can be analyzed. Sufficient conditions leading to uniform asymptotic stability were deduced. Numerical simulations and real-time experiments validated the applicability of the proposed NMPC scheme. Significantly better tracking errors were obtained for a typical pick-and-place trajectory. The proposed NMPC scheme also led to improved precision regarding its robustness against payload uncertainties. A video summarizing the main experiments performed with the proposed NMPC scheme can be watched here.

General Conclusion

The present thesis contributes to the state-of-the-art on trajectory position control of CDPRs. In the following, the main obtained results are summarized and an overview of possible future works are presented.

Summary of the work

The present thesis was developed within the framework of the european union H2020 project named Hephaestus. **Chapter 1** introduced the main requirements related to the position tracking of the CDPR prototype used in this project. Besides positioning accuracy, one important requirement is that the CDPR should safely operate close to the system constraints. More in detail, the robot operation includes trajectories demanding nominal cable tensions close to the cable tension limits. Moreover, a significant incidence of disturbances and modeling errors is expected.

In this context, particular attention should be devoted to the cable tension limits. A conventional control scheme may return feedback corrections that are not consistent with the interval of admissible tensions. Indeed, the overview of the state-of-the-art on position tracking of CDPRs presented in Chapter 1 indicates that the existing control schemes are prone to this issue. More precisely, these control schemes operating in the aforementioned conditions expected for the Hephaestus project may either return unfeasible desired cable tensions, or simply crash (attain a condition without a well-defined output). Since MPC is one of the few control strategies able to explicitly handle system constraints, this control design methodology is chosen.

Before addressing the MPC control strategies, some preliminary issues had to be solved. Some of these issues are addressed in **Chapter 2**. In order to meet high positioning precision, an FK algorithm considering the pulley kinematics is proposed. An explicit expression for the differential kinematics enabled the implementation of a numerical solution of the nonlinear least squares system representing the FK problem. Its convergence capabilities are evaluated experimentally and numerically.

Moreover, the applicability of different closed-loop cable tension control strategies is studied. Such control schemes aim to generate the desired cable tensions using the measurements obtained with force sensors. Firstly, a torque-based control scheme is analyzed. Simulations and experimental results indicate that this approach leads to crucial issues related to the friction present in the gear trains. In contrast, the pro-

posed velocity-based scheme is able to tackle these issues with a non-model based approach. Accordingly, the proposed strategy may be implemented without the identification of the dynamic parameters of the winches. Details on the numerical and experimental validation are presented. This approach proves to be appropriate for CDPRs presenting winches with large reduction ratio, which leads to significant friction and model uncertainties. Using the aforementioned kinematic model and the velocity-based cable tension control, the implementation of different MPC position tracking control schemes is enabled.

Assuming limited incidence of nonlinearities on the CDPR dynamic system, **Chapter 3** introduces a linear model able to estimate the mobile platform motion within the prediction horizon. Based on this approximation, a linear MPC scheme is proposed. Numerical simulations and experimental tests proved that the linear MPC may safely operate close to system constraints. This capability is validated applying a desired trajectory that cannot be performed without violating the cable tension limits. The proposed linear MPC scheme is able to perform a trajectory as close as possible to the desired one while satisfying cable tension bounds. Conversely, state-of-the-art control schemes are not able to suitably respond under such conditions. Comparing the behavior obtained with the proposed linear MPC and a state-of-the-art control scheme, one may conclude that the capability to operate close to system constraints represents an important result related to the safety of operation of CDPRs.

Additionally, this achievement may enable improvements on the design of CDPRs. Typically, acknowledging the risks taken when operating close to system constraints, the design of a CDPR should lead to oversized dimensions. The proposed MPC scheme, being able to safely operate in these conditions, enables the designer to reduce such oversized dimensions.

Nevertheless, it was noted that the proposed linear MPC may be sensitive to increased nonlinearities. The positioning precision may be deteriorated for trajectories presenting relatively high velocities. Accordingly, **Chapter 4** introduces an NMPC able to consider system nonlinearities. In contrast to its linear counterpart, the proposed NMPC can be classified as a standard tracking NMPC, and the stability of the resulting closed-loop system could be analyzed. In order to fit with the required characteristics of standard tracking NMPCs, an alternative method for the minimization of the cable tensions is proposed. Details on its numerical implementation are presented. The improved performance is validated experimentally and with numerical simulations.

After the construction of the Hephaestus CDPR prototype, the very first tests using this prototype were performed on November 2019. Acceptable results were obtained using a kinematic control (discussed in Section 1.4.1). For this reason, the consortium dedicated low priority to the implementation of advanced control strategies with respect to other primordial tasks. Accordingly, the time available (one week) for the implementation of the proposed MPC schemes on the Hephaestus prototype was not sufficient to obtain significant results. Such implementation remains a focus of possible future works.

It is worth recalling that all the experimental results presented in this thesis were obtained using software and hardware commonly used in the industry. This demon-

strates the applicability of the proposed methods in commercial applications.

Perspectives

Satisfying results were obtained with the numerical algorithms proposed in this thesis. Nevertheless, as highlighted in the course of this manuscript, many of them should be subjected to a rigorous analysis in order to be thoroughly validated. This is the case, for instance, of the numerical solution of the NLP (4.97) proposed in Section 4.3.3.2. Even though the algorithm described in Section 4.3.3.2 led to satisfying results in the studied CDPRs and trajectories, a thorough evaluation of its convergence capabilities may validate its applicability in a general case. Similarly, the Algorithm 2 used to compute $\tilde{\boldsymbol{\tau}}_r$ and the numerical solution of QP problems proposed in Section A.3.2 should be compared to other numerical methods in order to evaluate if the computation time can be reduced.

In addition to the studies presented in this manuscript, some other works were initiated and may be pursued in future works. An MPC scheme based on path following was proposed. Whereas a *trajectory* defines desired poses for each time instant, a *path* defines a set of poses that are used as reference without any connection with time. Therefore, in contrast to position tracking schemes, this path following MPC does not consider a specific desired state for each sampling time. The main goal is to attain a final desired pose as fast as possible following a predefined path. The path is defined as a curve in an n dimensional space (with n the number of DoF of the CDPR) representing the platform poses.

Additionally, preliminary results were obtained using the proposed cable tension control in order to operate a CDPR with an admittance control scheme. Thereby, the robot pose may be passively controlled by external efforts applied on its mobile platform. Moreover, hybrid position-force control was also successfully tested. Some degrees of freedom may be tracked with high stiffness whereas the remaining degrees of freedom are driven by the influence of external wrenches. This approach may be helpful to perform CDPR aided assembly tasks. Tests using the HRPCable prototype indicates that the platform may be positioned by hand while gravitational effects are compensated.

The reader may note that an important result obtained with the control schemes proposed in Chapters 3 and 4 is the capability to track unfeasible trajectories. Nevertheless, Assumptions 4.1 and 4.4 are not satisfied in this case. Accordingly, improved versions of the NMPC proposed in Chapter 4 could be examined in order to guarantee stability in cases where Assumptions 4.1 and 4.4 are violated. The corresponding stability analysis would use tools designed to economic MPC schemes. In accordance with the brief overview on EMPC presented in Section 1.7.2, the stage cost of such control schemes does not satisfy (4.11). As also discussed in Section 1.7.2, while results for constant and periodic reference states are well-known [163,164,166–168], there are very few studies addressing the case of a general desired trajectory [169, 170]. Therefore, a first stability analysis using the theory addressing EMPC would probably consider a time-invariant system (with a constant desired pose).

Appendices

A

Real Time Implementation - TwinCAT

The application of the proposed MPC strategies demands non-trivial numerical operations and algorithms. Most of these operations consist of matrix manipulations. There are several open source libraries that may be used for this purpose. Nevertheless, the software TwinCAT is not compatible in real-time with third party mathematical libraries. More precisely, TwinCAT uses an alternative mathematical library instead of the basic header `math.h`*. Therefore, C++ codes were developed based on the alternative TwinCAT mathematical library. The present appendix discusses some details of this implementation. Section A.1 compares typical programming approaches that can be used for matrix manipulation, highlighting the constraints imposed by the real-time applications in TwinCAT. Section A.2 briefly introduces the main rationale involved in the implementation of the C++ library. Some details of the implemented matrix factorization methods and optimization algorithms are presented in Section A.3.*

Contents

A.1 Motivation

Limitations of memory allocation in real-time are difficult to manage and to predict [189]. For this reason, it is preferable to allocate the necessary memory before the real-time operation. Accordingly, this is also recommended for TwinCAT real-time applications [190]. Hence, variables are often declared using static allocation in robotic applications. Nevertheless, the implementation of the proposed MPC schemes would most probably be troublesome with this approach. This is illustrated in the following.

Generally, the use of interpreted programming languages (such as MATLAB and Python) or libraries of matrix manipulation for compiled programming languages (such as Eigen [191] for C++) enables the user to numerically manipulate matrices in a way similar to the notation commonly used for symbolic matrix operations. For instance, using one of the aforementioned programming solutions, one may compute `C` as the sum of two known matrices `A` and `B` with one line of code similar to the following script:

```
C = A + B;
```
(A.I)

More in detail, in order to compute (A.I), the operator + dynamically allocates the memory equivalent to a matrix with the same size of `A` (and `B`), and computes the sum of each element `A[i][j] + B[i][j]`. Subsequently, the operator = attributes to `C` the obtained values and frees the dynamically allocated memory. Using operators similarly defined, more elaborate matrix operations may be performed. For instance, several operations may concatenated such as in the script below:

```
C = (A' - B*8)*(B + A*B');
```
(A.II)

where the notation defined in MATLAB was used for illustrative purpose.

One may note that this approach may significantly facilitate the implementation of numerical algorithms using complex matrix operations. Nevertheless, dynamic memory allocation is a crucial step for each one of the operators. Using the illustrative operation (A.II), an equivalent C++ code with static memory allocation can be represented as the following:

```
transpose(At*,A*,na,na);
multiplication_by_scalar(B1*,B*,8,na,na);
difference(AtB*,At*,B1*,na,na);
transpose(Bt*,B*,na,na);
multiply(ABt*,A*,Bt*,na,na);
sum(BABt*,B*,ABt*,na,na);
multiply(C*,AtB*,BABt*,na,na);
```
(A.III)

with additional statically allocated matrices `At`, `B1`, `AtB`, `Bt`, `ABt` and `BABt`. These matrices present the same number `na` of rows and columns than `A`, `B` and `C`. In addition,

`A*` represents the pointer to the first element of matrix `A` (and similarly to the remaining matrices).

Clearly, although (A.II) and (A.III) lead to equivalent outputs, the comparison between these scenarios indicates that the implementation of elaborate matrix operations should be facilitated with an approach similar to the one used in (A.II). However, as recommended in [190], memory allocation before the real-time operation should be preferred.

A.2 Basics

The main goal sought in the implementation of the proposed library (called `MatLib`) is to attain similar programming advantages to the interpreted programming languages without the need of dynamic memory allocation. To this end, a large memory is allocated in the initialization of the control scheme (out of real-time) and `MatLib` is responsible to explicitly manage this memory. By this means, `MatLib` enables the user to declare matrices with different sizes using the memory which has been allocated before the real-time operation.

Every operator and function available in this library (and its dependent libraries) are compatible with any 2 dimensional matrix. With the same philosophy, operations with multidimensional matrices are possible. Nevertheless, since the proposed control can be implemented with 2 dimensional matrices, the multidimensional case was not addressed.

Each matrix is declared as an object of the class `Matrix`. This class presents, among others, the members `rows` and `columns` of type `int` determining the number of rows and columns of the matrix. Additionally, the member `p` of type `double *` points to the first element of the corresponding matrix. Once a `Matrix` is declared, the members `p`, `rows` and `columns` may change.

Besides these members, this class presents several overloaded operators so that the matrix operations may be programmed in an intuitive manner (such as in (A.II)). These operators use an object belonging to the class `ProvisoryMatrix`. This object has access to a large memory which is allocated before the real-time operation. In order to illustrate the main used rationale, consider the following definition of the operators `+` and `=` which are applied between two objects belonging to the class `Matrix`:

```
Matrix Matrix::operator+(Matrix M)
{
  MpAllocate(rows, columns, Mp, MpG);
  for(i = 1; i <= rows; i++) { \\
    for (j = 1; j <= columns; j++) {
      Mp(i, j) = (*this)(i, j) + M(i, j);
    }
  }
  return Mp;
}
```

```
void Matrix::operator=(Matrix M)
{
  for (i = 1; i < M.rows + 1; i++) {
    for (j = 1; j < M.columns + 1; j++) {
      (*this)(i,j) = M(i, j);
    }
  }
  MpG.free = 0;
}
```

Objects `Mp` and `MpG` belong, respectively, to the classes `Matrix` and `ProvisoryMatrix`. The function `MpAllocate(rows, columns, Mp, MpG)` reserves the necessary memory in `MpG` and sets-up `Mp` to access this memory. This is done setting the pointer `Mp.p` to the address of the first element within this memory slot and attributing `Mp.rows = rows` and `Mp.columns = columns`. The necessary memory reserved is equal to `rows*columns` numbers of type `double`. The memory is managed by the variable `MpG.free`, which indicates how many numbers of type `double` are being currently used within the memory allocated by `MpG`. Accordingly, the `MpAllocate(rows, columns, Mp, MpG)` sets `MpG.free = MpG.free + rows*columns`. Conversely, this variable is set to `0` whenever the operator = is called, indicating that the memory of managed by `MpG` may be overwritten. The operator `()` taking two integers (say `i` and `j`) as arguments returns the reference of the element in row `i` and column `j` of the corresponding matrix. This way, the returned variable from the operator `()` may be used to read and write the corresponding element of a matrix.

Using a similar methodology, the common matrix operations were programmed using intuitive operators without the need of dynamic memory allocation. The implemented notation is inspired by the one used in MATLAB. Following, a non-extensive list of implemented trivial matrix operations and declarations is presented:

- `Matrix operator() (int i1, int i2, int j1, int j2)`: Returns a submatrix of rows between `i1` and `i2`, columns between `j1` and `j2`. A call `M(i1,i2,j1,j2)` is equivalent to `M(i1:i2,j1:j2)` in MATLAB;
- Basic matrix operations `+`, `-` and `*` (sum, difference and multiplication);
- Method `Matrix trp()`: returns the transposed of the matrix;
- Method `Matrix Norm()`: returns the matrix with the norm of each column vector;
- `Matrix eiFun(int iIn, int mIn)`: returns a vector with `mIn` lines with every element equal to zero, except for the `iIn`th, which is equal to 1;
- `Matrix Eye(int iIn, int jIn)`: returns a `iIn`×`jIn` matrix with every element equal to zero, except for those in the main diagonal, which are equal to 1;
- `Matrix Zeros(int iIn, int jIn)`: `iIn`×`jIn` zero matrix;
- `Matrix Ones(int iIn, int jIn)`: `iIn`×`jIn` matrix with every element equal to 1;
- `Matrix Sum(Matrix Min, int Dir)`: Calculates the sum of each row or

column;

- `Matrix Unit(Matrix A)`: Calculates the unit vector corresponding to each column of the input matrix;
- Basic IIR and FIR filters;
- Rotations matrices and their derivatives in the group SO(3).

As a matter of fact, the operation (A.II) can be performed with the implemented library in TwinCAT with the line of code presented below:

```
C = (A.trp() - B*8)*(B + A*B.trp());
```
(A.IV)

In contrast to the matrices used by the operators presented above, a `Matrix` may present a fixed allocated memory. This is the case for most of the matrices used in more elaborate implemented functionnalities. For instance, matrices used in the dynamic model of the CDPR (such as $\mathbf{M}$, $\mathbf{C}$, and $\mathbf{W}$) are preallocated before the real-time operation.

Besides operations with matrices containing floating point numbers, other functionalities dealing with matrices containing integers as elements (class `IntMatrix`) were also implemented. A short list of implemented features is presented below:

- Columns and row operations: access, add or delete non-consecutive columns or rows;
- `Intersection` and `Union` of two sets represented by two vectors;
- `GetLines(B, A, I1)`: Assign to matrix `B` the lines `I1` of the matrix `A`;
- `iMinVect(vin)` and `iMaxVect(vin)`: Returns the index of the minimal and maximal member of a vector;
- `FindZerosVector`: Finds elements that has absolute value smaller than a threshold.

Thanks to the tools obtained with `MatLib`, an additional module able to simulate the control scheme was implemented in C++ (compatible with TwinCAT). This module simulates the closed-loop system representing the CDPR behavior. This significantly facilitates the *debugging* during the implementation.

A.3 Advanced Matrix Computation

In order to implement the proposed MPC and other control strategies, basic features introduced in Section A.2 were used as a basis to develop methods of matrix algebra. The theory and the implementation in C++ (compatible with TwinCAT) of these methods are briefly discussed in this section. The theory used here is detailed in [88, 99, 176].

A.3.1 Matrix factorization

Several features of matrix algebra are frequently based on matrix factorization. For instance, the solution of linear systems (determined, undetermined or overdetermined), matrix inversion, calculation of the null space, the Moore-Penrose inverse and the Singular Value Decomposition (SVD) are typically based on matrix factorization. This section discusses three important methods: LU, Cholesky and QR. All of them present advantages and drawbacks relevant in the context of this thesis. Accordingly, they were implemented based on [88, 176].

For a given nonsingular square matrix $\mathbf{A}$, the aim of the LU decomposition is to find an upper triangular square matrix $\mathbf{U}$ and a lower traingular matrix $\mathbf{L}$ so that

$$\mathbf{\Pi}\,\mathbf{A} = \mathbf{L}\,\mathbf{U},$$

with $\mathbf{\Pi}$ a permutation matrix resulted from pivoting. This decomposition is particulaly useful for the solution of determined linear systems. Using LU factorization, a linear set may be decomposed as

$$\mathbf{A}\,\mathbf{x} = \mathbf{b} \Rightarrow \mathbf{L}\,\mathbf{U}\,\mathbf{x} = \mathbf{\Pi}^{-1}\mathbf{b} \tag{A.1}$$

which can be solved by

$$\mathbf{L}\,\mathbf{y} = \mathbf{\Pi}^{-1}\mathbf{b} \tag{A.2a}$$
$$\mathbf{U}\,\mathbf{x} = \mathbf{y}. \tag{A.2b}$$

Since both matrices $\mathbf{L}$ and $\mathbf{U}$ are triangular, equations (A.2) may be solved with backsubstitution (firstly (A.2a) and then (A.2b) using the obtained $\mathbf{y}$). LU decomposition was implemented using pivoting and Crout's algorithm. This method is known as the preferrefed way to solve a linear set of equations [176].

If the matrix of interest is symmetric and positive definite, a more convenient factorization may be used, namely, the Cholesky decomposition. In this case, matrix $\mathbf{A}$ may be decomposed as $\mathbf{A} = \mathbf{L}\,\mathbf{L}^T$ and the solution of the linear system may be computed similarly to (A.1)-(A.2). Generally, this method is approximatelly two times faster than a conventional LU decomposition and presents good numerical stability without the need of pivoting. Symmetric positive definite matrices are often present in convex optimization problems. For this reason, Cholesky decomposition was used as described in the Section A.3.2.

Additionally, the QR factorization also presents advantages that were important in several steps in this thesis. Here, a rectangular matrix $\mathbf{B}$ is decomposed as $\mathbf{B} = \mathbf{Q}\,\mathbf{R}$ with $\mathbf{Q}$ square orthonormal and $\mathbf{R}$ upper triangular. This decomposition was implemented using Householder reductions as described in [176, Section 11.2] and [88, Section 5.1]. Clearly, the solution of the linear system $\mathbf{B}\,\mathbf{x} = \mathbf{b}$ can be computed applying backsubstitution for

$$\mathbf{R}\,\mathbf{x} = \mathbf{Q}^T\mathbf{b}. \tag{A.3}$$

Note that, since no permutation matrix is present in (A.3), pivoting is not necessary. However, other applications of the QR factorization motivated its application in this

thesis. As discussed in [88], this factorization is particularly efficient for the solution of least squares problems. Accordingly, this method was used in Section 2.1.3 for the solution of the FK problem.

In addition, for $\mathbf{B} \in \mathbb{R}^{n_1 \times n_2}$, $n_1 > n_2$, the matrices $\mathbf{Q}$ and $\mathbf{R}$ can be written with submatrices as

$$\mathbf{B} = \mathbf{Q}\,\mathbf{R} = \begin{bmatrix} \mathbf{Q}_1 & \mathbf{Q}_2 \end{bmatrix} \begin{bmatrix} \mathbf{R}_1 \\ \mathbf{0} \end{bmatrix}, \tag{A.4}$$

with columns$(\mathbf{Q}_1)$ = rows$(\mathbf{R}_1)$ (number of columns of $\mathbf{Q}_1$ is equal to the number of rows of $\mathbf{R}_1$), it is easy to verify that

$$\mathbf{Q}_2^T\,\mathbf{B} = \mathbf{0}. \tag{A.5}$$

This indicates that $\mathbf{Q}_2$ is in the null-space of $\mathbf{B}^T$. Indeed, QR factorization may be used in order to compute the null-space of a rectangular matrix. This utility was used in Section 4.3.2. This section also details that matrices $\mathbf{Q}$ and $\mathbf{R}$ may be updated with reduced computational cost if one row is added or removed from $\mathbf{B}$, which is useful in active-set optimization algorithms.

A.3.2 Quadratic programming

This section discusses the numerical solution of QP problems of type

$$\min_{\mathbf{u}\in\mathbb{R}^{n_u}} \frac{1}{2}\,\mathbf{u}^T\mathbf{H}\,\mathbf{u} + \mathbf{d}^T\mathbf{u} \tag{A.6a}$$

$$\text{s.t.}\;\; \mathbf{u}_{min} \leqslant \mathbf{u} \leqslant \mathbf{u}_{max} \tag{A.6b}$$

with $\mathbf{H} \in \mathbb{R}^{n_u \times n_u}$ symmetric and positive definite, and $\mathbf{d}$, $\mathbf{u}_{min}$, $\mathbf{u}_{max} \in \mathbb{R}^{n_u}$ constant vectors.

Two methods were considered: active-set and projected gradient [99]. Projected gradient is simpler to implement and may be pertinent for box constraints such as (A.6b). Nevertheless, active-set finds the optimal solution in a finite number of iterations and can easily consider additional constraints (this later advantage may be used in future works). Therefore, the active-set technique was chosen. This method solves a simplified version of (A.6) taking a subset of the constraints as *equality constraints* at each iteration (a detilaed discussion addressing this technique is presented in [99, Section 16.5]). The resulting subproblem in a given iteration can be written as

$$\begin{aligned} \min_{\mathbf{u}} \quad & \frac{1}{2}\mathbf{u}^T\mathbf{H}\mathbf{u} + \mathbf{d}^\mathbf{T}\mathbf{u} \\ \text{s.t.} \quad & \begin{cases} u_i = u_{min,i}, & \text{for } i \in \mathcal{M} \\ u_j = u_{max,j}, & \text{for } j \in \mathcal{N} \end{cases} \end{aligned} \tag{A.7}$$

with $\mathcal{M} \subset \{1, ..., n_u\}$, $\mathcal{N} \subset \{1, ..., n_u\}$ and $\mathcal{M} \cap \mathcal{N} = \varnothing$. Take $\mathcal{A} = (\mathcal{M} \cup \mathcal{N})$ and $\mathcal{F} = \{1, ..., n_u\} \backslash (\mathcal{M} \cup \mathcal{N})$. For notation simplicity, let us consider that $\mathcal{F} = \{1, ..., n_f\}$, $\mathcal{M} = \{n_f + 1\,,\, ...\,,\, n_f + n_m\}$ and $\mathcal{N} = \{n_u - n_n + 1\,,\, ...\,, n_u\}$, with n_f the number

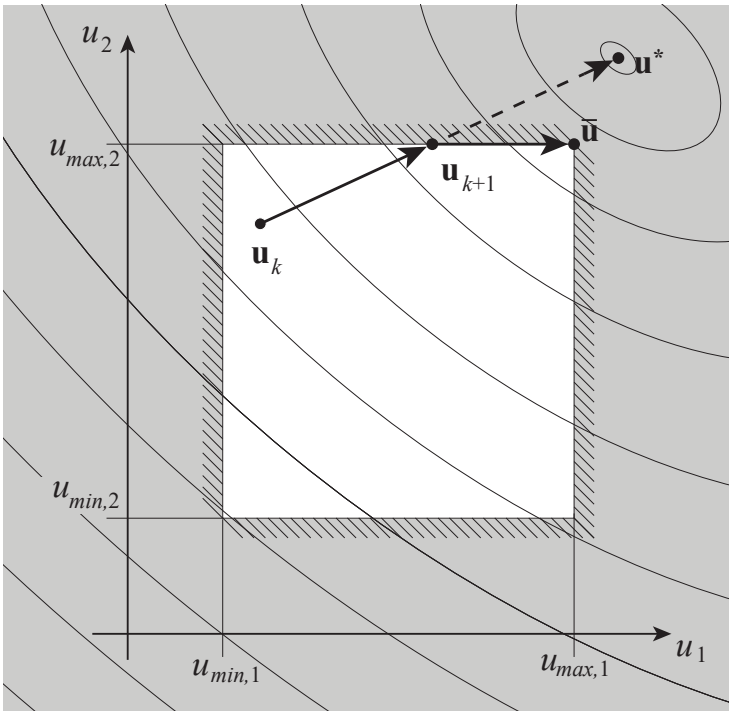

Figure A.1 – *An illustrative example demonstrating that the saturation* (A.8) *may reduce the necessary number of iterations.*

of elements of $\mathcal{F}$, n_m the number of elements of $\mathcal{M}$ and n_n the number of elements of $\mathcal{N}$. $\mathbf{H}$, $\mathbf{U}$ and $\mathbf{d}$ in (A.7) can be written as follows

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_f & \mathbf{H}_1 \\ \mathbf{H}_1^T & \mathbf{H}_2 \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} \mathbf{u}_f \\ \mathbf{u}_m \\ \mathbf{u}_n \end{bmatrix}, \quad \mathbf{d} = \begin{bmatrix} \mathbf{d}_f \\ \mathbf{d}_m \\ \mathbf{d}_n \end{bmatrix},$$

where $\mathbf{H}_f \in \mathbb{R}^{n_f \times n_f}$, $\mathbf{H}_1 \in \mathbb{R}^{n_f \times n_m + n_n}$, $\mathbf{H}_2 \in \mathbb{R}^{n_m + n_n \times n_m + n_n}$, $\mathbf{u}_f \in \mathbb{R}^{n_f}$, $\mathbf{u}_m \in \mathbb{R}^{n_m}$ and $\mathbf{u}_n \in \mathbb{R}^{n_n}$. Note that each element of $\mathbf{u}_m$ is equal to the corresponding element of $\mathbf{u}_{min}$ and each element of $\mathbf{u}_n$ is equal to the corresponding element of $\mathbf{u}_{max}$. The solution of (A.7) is $(\mathbf{u}^*)^T = \begin{bmatrix} (\mathbf{u}_f^*)^T & \mathbf{u}_m^T & \mathbf{u}_n^T \end{bmatrix}$, with $\mathbf{u}_f^*$ the vector that solves $\mathbf{H}_f \, \mathbf{u}_f = -\mathbf{d}_f - \mathbf{H}_1 \begin{bmatrix} \mathbf{u}_m^T & \mathbf{u}_n^T \end{bmatrix}^T$.

It is worth noting that $\mathbf{H}_f$ is symmetric and positive definite. As a consequence, $\mathbf{H}_f$ may be factorized with Cholesky decomposition such that $\mathbf{H}_f = \mathbf{L}\,\mathbf{L}^T$, with $\mathbf{L}$ lower triangular, as described in the last section. Advantages of using this technique were also presented. The optimal $\mathbf{u}_f^*$ is obtained from two successive back substitution procedures: $\mathbf{L}\,\mathbf{s} = -\mathbf{d}_f - \mathbf{H}_1 \begin{bmatrix} \mathbf{u}_m^T & \mathbf{u}_n^T \end{bmatrix}^T$, and then $\mathbf{L}^T \mathbf{u}_f^* = \mathbf{s}$.

Before starting the next iteration of the active-set method, components of $\mathbf{u}_f^*$ should comply with cable tension limits. Good results were obtained using a simple saturation function. More precisely, each element $\overline{u}_i$ of the vector $\overline{\mathbf{u}}$ used in the next iteration is obtained as a saturation of the corresponding element u_i^* of $\mathbf{u}^*$

$$\overline{u}_i = \min\Big(\max(u_i^*, u_{min,i}), u_{max,i}\Big). \tag{A.8}$$

It was noted that, generally, the application of (A.8) reduces the necessary number

of iterations in order to obtain the solution of (A.6) in comparison to the conventional method discussed in Section 4.3.2 (and illustrated in Figure 4.5). The probable cause for this difference is illustrated in Figure A.1. Whereas the conventional method described in Section 4.3.2 would demand two iterations in order to obtain $\overline{\mathbf{u}}$, this vector is obtained in the first iteration using (A.8). This difference related to the necessary number of iterations was noticed experimentally and in numerical simulations. However, the finite termination guaranteed by a conventional method [99, Section 16.5] is not valid in this case. The vector $\overline{\mathbf{u}}$ is used as input for the next iteration of the active-set method changing the set of active constraints, similarly to the algorithm described in Section 4.3.2.

B

Minimum Infinity Norm Redundancy Resolution

A tension distribution algorithm able to minimize the infinity norm of the vector of cable tensions is introduced. As discussed in Section 1.5, most of the TDA implemented for robots with $m > n + 1$ are based on iterative algorithms. In contrast, the proposed TDA presents a closed form expression for the computation of the optimal cable tensions. The algorithm itself is introduced in Section B.1. Section B.2 proves that the algorithm returns the optimal vector of cable tensions. Finally, experimental results are presented in Section B.3.

Contents

B.1 Tension Distribution Algorithm

Considering the tension distribution problem of a CDPR with n degrees of freedom driven by m cables ($m > n$), this section presents a closed-form solution for the minimization of the infinity norm of the cable tensions vector. More precisely, the proposed non-iterative algorithm is able to solve (B.1):

$$\begin{aligned} &\min_{\boldsymbol{\tau}\in\mathbb{R}^m} \|\boldsymbol{\tau}\|_\infty \\ &\text{s. t. } \mathbf{W}\,\boldsymbol{\tau} = \mathbf{f} \\ &\qquad \tau_{min} \leqslant \boldsymbol{\tau} \end{aligned} \tag{B.1}$$

for given wrench matrix $\mathbf{W} \in \mathbb{R}^{n\times m}$, desired wrench $\mathbf{f} \in \mathbb{R}^n$ and minimal cable tension τ_{min} (it is assumed that $\tau_{min,i} = \tau_{min}$, $\tau_{max,i} = \tau_{max}\ \forall\ i \in \mathbb{N}_{1,m}$ in order to simplify the presentation of the algorithm). Note that the definition of the infinity norm leads to

$$\|\boldsymbol{\tau}\|_\infty = \max_i(\tau_i).$$

The proposed algorithm is based on the theory of wrench feasibility discussed in [111, 112]. The tension distribution constraints imposed for cable tensions $\boldsymbol{\tau}$ are

$$\begin{cases} \mathbf{W}\,\boldsymbol{\tau} = \mathbf{f} \\ \tau_{min} \leqslant \boldsymbol{\tau} \leqslant \tau_{max} \end{cases} \tag{B.2}$$

Using the hyperplane shifting method [111, 112], (B.2) can be written as (B.3)

$$\mathbf{C}\,\mathbf{f} \leqslant \mathbf{d} \tag{B.3}$$

where each line of the matrix $\mathbf{C}$ is perpendicular to $n-1$ columns taken among the m columns of $\mathbf{W}$. More specifically, there are $n_c = \binom{m}{n-1}$ possible combinations of vectors for a $\mathbf{W} \in \mathbb{R}^{n\times m}$. Denote $\mathbf{W}_0^k$ the submatrix formed by each one of these combinations with $k \in \mathbb{N}_{1,n_c}$. Each $\mathbf{W}_0^k$ results in two lines (say $\mathbf{c}_l$ and $\mathbf{c}_p$) of $\mathbf{C}$:

$$\begin{cases} \mathbf{c}_l^{\mathrm{T}} &= \text{null}\left\{(\mathbf{W}_0^k)^{\mathrm{T}}\right\} \\ \mathbf{c}_p &= -\mathbf{c}_l \end{cases}. \tag{B.4}$$

Each line $\mathbf{c}_i$ corresponds to the orientation of a hyperplane which restricts the feasible wrenches. Vector $\mathbf{d}$ translates these hyperplanes. Their components d_j are the translation normal to each hyperplane in relation to the origin. These components are given by (B.5)

$$d_j = \tau_{max} \sum_{i\in I_j^+} \mathbf{c}_j\mathbf{w}_i + \tau_{min} \sum_{i\in I_j^-} \mathbf{c}_j\mathbf{w}_i. \tag{B.5}$$

Sets I_j^+, I_j^- and I_j^0 are subsets of $I = \mathbb{N}_{1,m}$ and are defined according to

$$\begin{cases} I_j^+ &= \left\{i \in I \mid \mathbf{c}_j\mathbf{w}_i > 0\right\} \\ I_j^- &= \left\{i \in I \mid \mathbf{c}_j\mathbf{w}_i < 0\right\} \\ I_j^0 &= \left\{i \in I \mid \mathbf{c}_j\mathbf{w}_i = 0\right\} \end{cases} \tag{B.6}$$

For given τ_{min}, $\mathbf{W}$ and $\mathbf{f}$, Hussein *et al.* [31] propose the minimization of τ_{max} as follows

$$\tau^*_{max} = \max_j \frac{\mathbf{c}_j\mathbf{f} - \sum_{i\in I_j^-} \tau_{min}\mathbf{c}_j\mathbf{w}_i}{\sum_{i\in I_j^+} \mathbf{c}_j\mathbf{w}_i}. \tag{B.7}$$

The τ^*_{max} obtained with (B.7) is the minimum value for τ_{max} that may be used in (B.2) for which there is a tension distribution $\boldsymbol{\tau}^*$ satisfying these constraints. Using the theory of the hyperplane shifting method, it is possible to calculate the tension distribution $\boldsymbol{\tau}^* \leqslant \tau^*_{max}$ which satisfies (B.2). In other words, the definition presented in (B.7) calculates the optimal τ^*_{max} and the completion of the TDA demands the calculation of each component of the cable tension vector. The optimality of the algorithm described below is proved in Section B.2.

Naming h the argument j that maximizes the expression present in (B.7). The $\boldsymbol{\tau}^* = \begin{bmatrix}\tau_1^* & \tau_2^* & \dots & \tau_m^*\end{bmatrix}^{\mathrm{T}}$ satisfying (B.1) should also satisfy (B.8):

$$\begin{cases} \tau^*_{max} & \text{if} \quad i \in I_h^+ \\ \tau_i^* = \tau_{min} & \text{if} \quad i \in I_h^- \\ \tau_{min} < \tau_i^* < \tau^*_{max} & \text{if} \quad i \in I_h^0 \end{cases}. \tag{B.8}$$

Therefore, elements $\{\tau_i^*,\ i \in (I_h^+ \cup I_h^-)\}$ are trivially defined using (B.8). The remaining $\{\tau_i^*,\ i \in I_h^0\}$ should be computed considering the equality constraint in (B.2). Define submatrices and subvectors of $\mathbf{W}$ and $\boldsymbol{\tau}^*$ relating to the sets I_h^+, I_h^- and I_h^0 according to

$$\begin{cases} \mathbf{W}_0 & = [\mathbf{w}_i],\ i \in I_h^0 \\ \mathbf{W}_+ & = [\mathbf{w}_i],\ i \in I_h^+ \\ \mathbf{W}_- & = [\mathbf{w}_i],\ i \in I_h^- \end{cases} \qquad \begin{cases} \boldsymbol{\tau}_0^* & = [\tau_i^*],\ i \in I_h^0 \\ \boldsymbol{\tau}_+^* & = [\tau_i^*],\ i \in I_h^+ \\ \boldsymbol{\tau}_-^* & = [\tau_i^*],\ i \in I_h^- \end{cases}. \tag{B.9}$$

Constraints (B.2) impose that

$$\mathbf{W}_0\,\boldsymbol{\tau}_0^* + \mathbf{W}_+\,\boldsymbol{\tau}_+^* + \mathbf{W}_-\,\boldsymbol{\tau}_-^* = \mathbf{f}. \tag{B.10}$$

.

Recalling that (B.8) defines $\boldsymbol{\tau}_+^*$ and $\boldsymbol{\tau}_-^*$, the remaining elements may be calculated using (B.10). In summary:

$$\begin{cases} \boldsymbol{\tau}_+^* = \tau^*_{max} \\ \boldsymbol{\tau}_-^* = \tau_{min} \\ \boldsymbol{\tau}_0^* = \mathbf{W}_0^\dagger\left(\mathbf{f} - \mathbf{W}_+\,\boldsymbol{\tau}_+^* - \mathbf{W}_-\,\boldsymbol{\tau}_-^*\right) \end{cases}, \tag{B.11}$$

where τ^*_{max} is defined according to (B.7). For a full-rank $\mathbf{W}$, Section B.2 proves that solution of (B.1) and TDA proposed in (B.11) are both unique and equivalent.

As an example, for a robot with 6 DoF, $n = 6$. Therefore, once each row of $\mathbf{C}$ is composed of a vector spanning the nullspace of five columns of the wrench matrix,

$$\#I_h^0 = 5,$$

with $\#(\cdot)$ denoting the cardinality of the set $(\cdot)$. Considering a full rank $\mathbf{W}$, a cable robot with $m = 8$ cables has

$$\begin{aligned} \#I_h^0 &= 5 \\ \#(I_h^+ \cup I_h^-) &= 3. \end{aligned} \tag{B.12}$$

B.2 Optimality Proof

The present section proves the equivalence between the tension distribution obtained with the general optimization (B.1) and the closed-form solution (B.11).

Consider given $\mathbf{W}$, $\mathbf{f}$ and τ_{min}. Take also the matrix $\mathbf{C}$ computed as in [112]. Let τ_{max}^* be defined according to

$$\tau_{max}^* = \max_j \frac{\mathbf{c}_j\mathbf{f} - \sum_{i\in I_j^-} \tau_{min}\mathbf{c}_j\mathbf{w}_i}{\sum_{i\in I_j^+} \mathbf{c}_j\mathbf{w}_i} = \frac{\mathbf{c}_h\mathbf{f} - \sum_{i\in I_h^-} \tau_{min}\mathbf{c}_h\mathbf{w}_i}{\sum_{i\in I_h^+} \mathbf{c}_h\mathbf{w}_i} \tag{B.13}$$

where h is equal to the argument which maximizes (B.13). The equation (B.13) can be written as

$$\mathbf{c}_h\mathbf{f} = \sum_{i\in I_h^-} \tau_{min}\mathbf{c}_h\mathbf{w}_i + \sum_{i\in I_h^+} \tau_{max}^*\mathbf{c}_h\mathbf{w}_i \tag{B.14}$$

Suppose that optimization (B.1) obtains a $\|\boldsymbol{\tau}\|_\infty = t_b < \tau_{max}^*$. Let $\mathbf{d}^b$ be the vector calculated using $\tau_{max} = t_b$ in (B.5). See that

$$\mathbf{c}_h\mathbf{f} = \left(\sum_{i\in I_h^-} \tau_{min}\mathbf{c}_h\mathbf{w}_i + \sum_{i\in I_h^+} \tau_{max}^*\mathbf{c}_h\mathbf{w}_i\right) > \left(\sum_{i\in I_h^-} \tau_{min}\mathbf{c}_h\mathbf{w}_i + \sum_{i\in I_h^+} \underbrace{t_b}_{<\tau_{max}^*} \underbrace{\mathbf{c}_h\mathbf{w}_i}_{>0}\right) = d_h^b \tag{B.15}$$

Therefore, h^{th} line of $\mathbf{C}\,\mathbf{f} < \mathbf{d}_b$ is violated. This proves that τ_{max}^* is equal to the solution of (B.1). After that, it is necessary to prove that TD proposed in (B.11) satisfies (B.2) with $\tau_{max} = \tau_{max}^*$.

Indeed, substituting $\tau_{max} = \tau_{max}^*$ in (B.2), $\mathbf{f}$ is feasible. As a consequence, there is at least one $\boldsymbol{\tau}^*$ so that

$$\begin{cases} \mathbf{W}\,\boldsymbol{\tau}^* = \mathbf{f} \\ \tau_{min} \leqslant \boldsymbol{\tau}^* \leqslant \tau_{max}^* \end{cases} \tag{B.16}$$

Therefore, using the same notation introduced in Section B.1

$$\mathbf{W}_0\,\boldsymbol{\tau}_0^* + \mathbf{W}_+\,\boldsymbol{\tau}_+^* + \mathbf{W}_-\,\boldsymbol{\tau}_-^* = \mathbf{f}. \tag{B.17}$$

Multiplying (B.17) by $\mathbf{c}_h$ and using (B.14)

$$\mathbf{c}_h\mathbf{W}_+\boldsymbol{\tau}_+^* + \mathbf{c}_h\mathbf{W}_-\boldsymbol{\tau}_-^* = \sum_{i\in I_h^-} \tau_{min}\mathbf{c}_h\mathbf{w}_i + \sum_{i\in I_h^+} \tau_{max}^*\mathbf{c}_h\mathbf{w}_i. \tag{B.18}$$

Notice that $\mathbf{c}_h\mathbf{W}_0 = \mathbf{0}$. Equation (B.18) can then be written as

$$\sum_{i\in I_h^-} \mathbf{c}_h\mathbf{w}_i(\tau_{min} - \tau^*_{-,i}) + \sum_{i\in I_h^+} \mathbf{c}_h\mathbf{w}_i(\tau^*_{max} - \tau^*_{+,i}) = \mathbf{0}. \tag{B.19}$$

According to (B.6), the definition of I_h^+ and I_h^- imposes that

$$\begin{cases} \mathbf{c}_h\mathbf{w}_i < 0,\ i \in I_h^- \\ \mathbf{c}_h\mathbf{w}_i > 0,\ i \in I_h^+ \end{cases}. \tag{B.20}$$

Besides, (B.16) leads to

$$\begin{cases} \tau_{min} - \tau^*_{-,i} \leqslant 0,\ i \in I_h^- \\ \tau^*_{max} - \tau^*_{+,i} \geqslant 0,\ i \in I_h^+ \end{cases}. \tag{B.21}$$

The combination of (B.20) and (B.21) results in

$$\sum_{i\in I_h^-} \mathbf{c}_h\mathbf{w}_i(\tau_{min} - \tau^*_{-,i}) \geqslant 0 \tag{B.22}$$

$$\sum_{i\in I_h^+} \mathbf{c}_h\mathbf{w}_i(\tau^*_{max} - \tau^*_{+,i}) \geqslant 0. \tag{B.23}$$

According to (B.19), the sum of l.h.s. of (B.22) and (B.23) should be equal to zero. Once $\mathbf{c}_h\mathbf{w}_i \neq 0$ for $i \in (I_h^+ \cup I_h^-)$,

$$\left\{ \begin{array}{ll} \tau^*_{-,i} & = \tau_{min} \\ \tau^*_{+,i} & = \tau^*_{max} \end{array} \right. \Rightarrow \left\{ \begin{array}{ll} \boldsymbol{\tau}^*_- & = \tau_{min} \\ \boldsymbol{\tau}^*_+ & = \tau^*_{max} \end{array} \right. . \tag{B.24}$$

This proves the two first equations of (B.11). The remaining unknown components $\boldsymbol{\tau}^*_0$ should satisfy (B.17):

$$\mathbf{W}_0\,\boldsymbol{\tau}^*_0 = \mathbf{f} - \mathbf{W}_+\,\boldsymbol{\tau}^*_+ - \mathbf{W}_-\,\boldsymbol{\tau}^*_-. \tag{B.25}$$

As stated in the beginning of this section, $\mathbf{W}$ is a full-rank matrix. Consequently, $\mathbf{W}_0$ also has full rank. Besides, $\mathbf{W}_0$ has dimensions of $(n) \times (n-1)$. Thereby, there is a unique $\boldsymbol{\tau}^*_0$ that satisfies (B.25). This vector can be computed as follows

$$\boldsymbol{\tau}^*_0 = \mathbf{W}_0^\dagger \left(\mathbf{f} - \mathbf{W}_+\,\boldsymbol{\tau}^*_+ - \mathbf{W}_-\,\boldsymbol{\tau}^*_-\right). \tag{B.26}$$

Therefore, the optimization (B.1) has a unique solution which is equal to the TDA (B.11). □

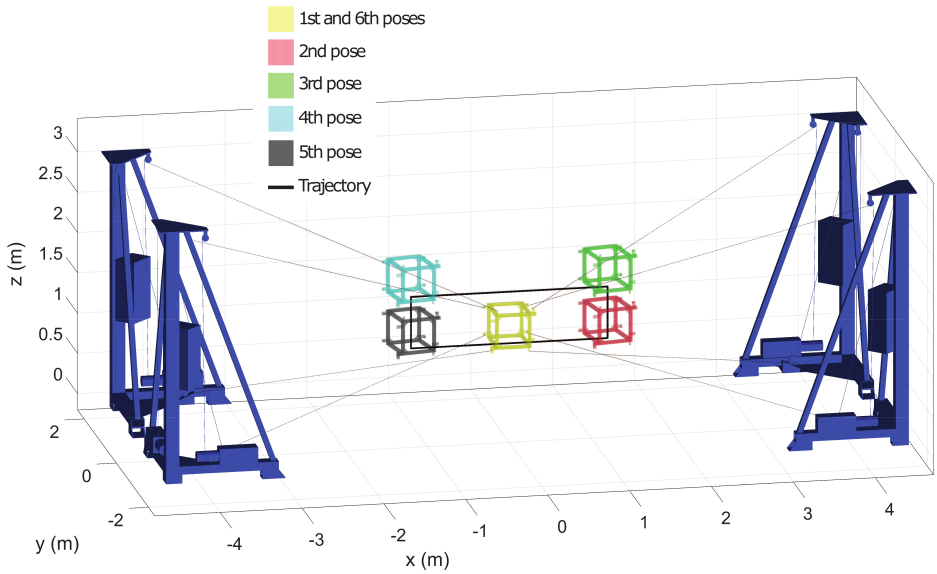

Figure B.1 – *CAD view of HRPCable and visited poses.*

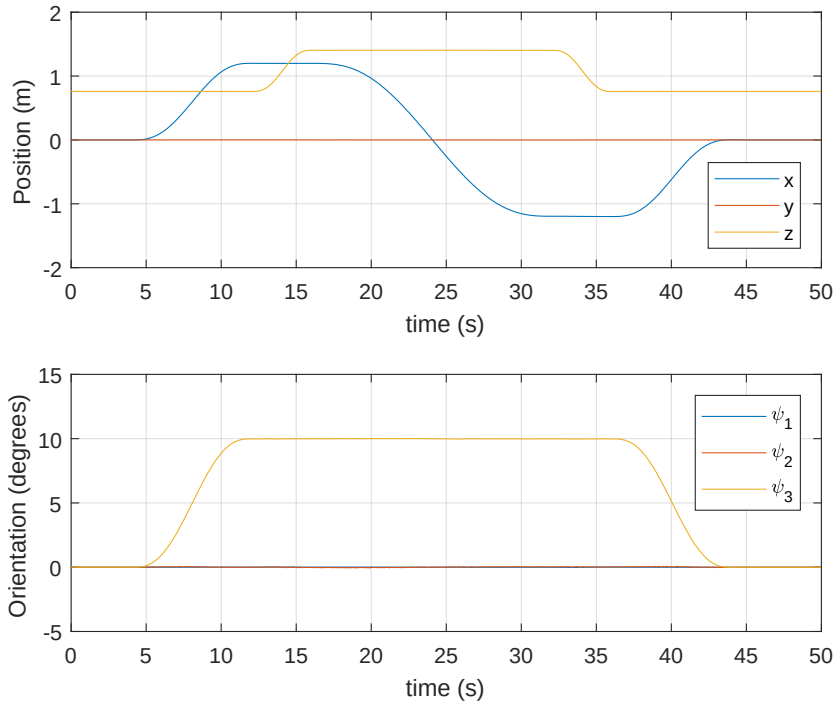

Figure B.2 – *Performed trajectory.*

B.3 Experimental Results

The TDA proposed in Section B.1 was implemented in the HRPCable prototype using the programming tools described in Appendix A. This section presents the main experimental results obtained with this algorithm. The trajectory depicted in Figs. B.1 and B.2 was used in this test.

Figure B.3 compares cable tensions obtained with the proposed tension distribution

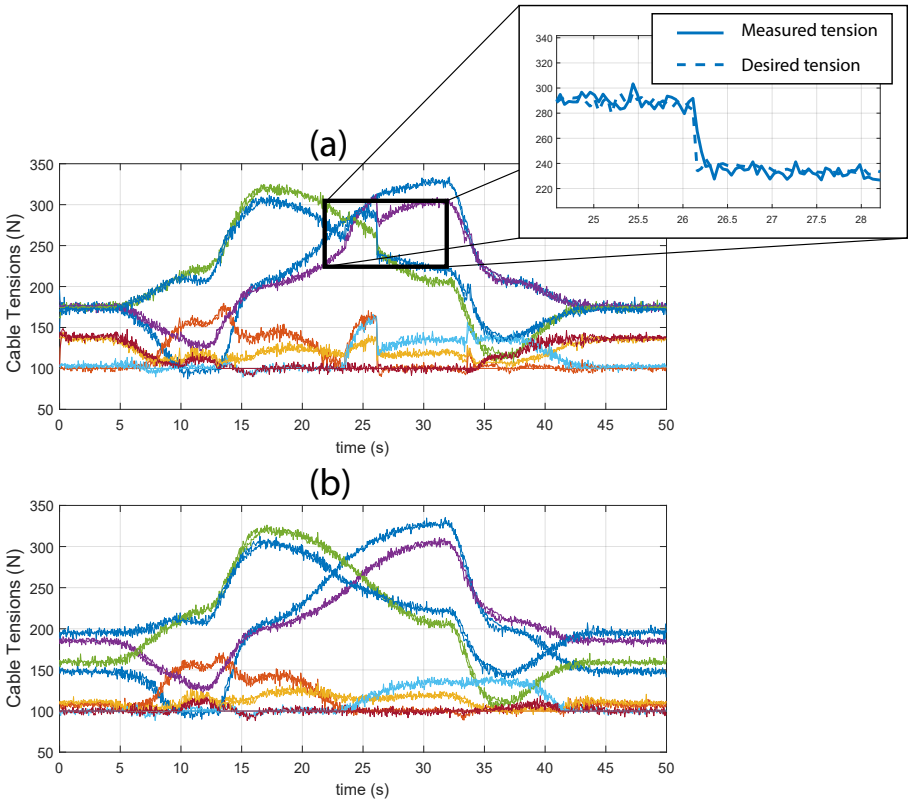

Figure B.3 – *Desired and measured cable tensions using the minimization of (a) infinity norm and (b) 2-norm.*

method (minimization of the infinity norm) and the cable tensions obtained with the minimization of the 2-norm of the cable tensions vector (the method described in [114] was used). As discussed earlier, (B.11) represents a *closed-form* expression able to solve the tension distribution problem in the control of CDPRs. This is an important advantage of the proposed method. Nevertheless, the main drawback resulted from the minimization of the infinity norm is the presence of discontinuous desired cable tensions. This drawback is visible in Figure B.3. At $t \approx 25$ s, one of the desired cable tensions jumps from 290 N to 230 N. However, this same figure also shows that the control of cable tensions can respond sufficiently fast. At any rate, discontinuities on the desired cable tensions should be avoided. Methods able to overcome this issue are still under investigation.

C

On the Proof of Theorem 4.3

This appendix discusses how the Theorem 4.3 is obtained based on the results described in [13]. The general formulation presented in [13, Chapter 6] is applied using the notations introduced in Chapter 4. Theorem 4.3 is based mainly on [13, Theorem 6.24]. The later addresses time-invariant closed-loop systems (with a constant desired state vector). Accordingly, first, a simplified case of Theorem 4.3 is considered taking a constant desired state. Sufficient conditions presented in [13, Theorem 6.24] are reviewed and shown to be equivalent to the conditions of the time-invariant case of Theorem 4.3. The time-varying case is obtained in accordance with the results of [13, Section 6.5].

Theorem 4.3 is based mainly on [13, Theorem 6.24]. As discussed in Chapter 4, the results presented in [13, Chapter 6] are based on upper bounds of the cost functional. More precisely, these upper bounds are studied with respect to the *optimal value function*, which is defined in the following.

For a $K \in \mathbb{N}$, the optimal value function $V_K : \mathbb{N} \times \mathbb{Y} \to \mathbb{R}$ is computed based on an instant k and state $\mathbf{y}$ according to

$$\begin{aligned} V_K(k, \mathbf{y}) = \inf_{\boldsymbol{\tau}(\cdot) \in \mathbb{U}^K} & J_K\left(k, \mathbf{y}, \boldsymbol{\tau}(\cdot)\right), \\ \text{s.t. } & \mathbf{y}_{\boldsymbol{\tau}(\cdot)}(j, \mathbf{y}) \in \mathbb{Y} \; \forall \; j \in \mathbb{N}_{1,K} \end{aligned} \tag{C.1}$$

with the cost functional J_K defined in (4.6). The scalar $V_K(k, \mathbf{y})$ is also known as the *cost-to-go* for given instant k and state $\mathbf{y}$. Note that, if the global minimum for the following optimal control problem exists,

$$\begin{aligned} \boldsymbol{\tau}^*(\cdot) = \arg\min_{\boldsymbol{\tau}(\cdot) \in \mathbb{U}^K} & \sum_{j=1}^{K} \ell\Big(j + k, \mathbf{y}_{\boldsymbol{\tau}(\cdot)}\left(j, \mathbf{y}\right), \boldsymbol{\tau}_i\Big) \quad , \\ \text{s.t. } & \mathbf{y}_{\boldsymbol{\tau}(\cdot)}\Big(j, \mathbf{y}\Big) \in \mathbb{Y} \; \forall \; j \in \mathbb{N}_{1,K} \end{aligned} \tag{C.2}$$

then $\boldsymbol{\tau}^*(\cdot)$ satisfies the equality

$$J_K\left(k, \mathbf{y}, \boldsymbol{\tau}^*(\cdot)\right) = V_K(k, \mathbf{y}). \tag{C.3}$$

In order to deduce Theorem 4.3, let us first consider a simplified case with a constant desired state

$$\mathbf{y}_d(k) = \mathbf{y}_d \; \forall \; k \in \mathbb{N} \tag{C.4}$$

with a constant vector $\mathbf{y}_d \in \mathbb{Y}$. In this case, the closed-loop is time-invariant, the proposed NMPC scheme obtained with Algorithm 1 may be written as in [13, Algorithm 3.1] and the [13, Theorem 6.24] may be directly applied as it is. Moreover, the value function V_K is constant with respect to time and may be simplified to

$$V'_K(\mathbf{y}) = V_K(k, \mathbf{y}) \; \forall \; k \in \mathbb{N}. \tag{C.5}$$

Similarly, the stage cost and minimal stage cost can be simplified according to

$$\left.\begin{aligned} \ell_c(\mathbf{y}, \boldsymbol{\tau}) &= \ell(k, \mathbf{y}, \boldsymbol{\tau}) \\ \ell_c^*(\mathbf{y}) &= \ell^*(k, \mathbf{y}) \end{aligned}\right\} \; \forall \; k \in \mathbb{N}. \tag{C.6}$$

The conditions stated in [13, Theorem 6.24] address time-invariant systems and are summarized as

(i) There exist $\alpha_1, \; \alpha_2 \in \mathcal{K}_\infty$ such that for every $\mathbf{y} \in \mathbb{Y}$ the following relation holds

$$\alpha_1(\|\mathbf{y}\|_{\mathbf{y}_d}) \leqslant \ell_c^*(\mathbf{y}) \leqslant \alpha_2(\|\mathbf{y}\|_{\mathbf{y}_d}); \tag{C.7a}$$

(ii) (*[13, Assumption 6.3] with linear* B_K) For each $\mathbf{y} \in \mathbb{Y}$ and $K \in \mathbb{N}$, there exists $\gamma_K \in \mathbb{R}$ with $\gamma_K < \infty$ so that the optimal cost functional $V_K(\mathbf{y})$ satisfies

$$V'_K(\mathbf{y}) \leqslant \gamma_K \, \ell_c^*(\mathbf{y}). \tag{C.7b}$$

Additionally, [13, Lemma 6.6] shows that, if, for each $\mathbf{y} \in \mathbb{Y}$ and $N \in \mathbb{N}$, there exist $\boldsymbol{\tau}(\cdot) \in \mathbb{U}^N$ and real constants $C \geqslant 1$, $\sigma \in (0,1)$ such that

$$\ell_c\left(\mathbf{y}_{\boldsymbol{\tau}(\cdot)}(n,\mathbf{y}),\boldsymbol{\tau}_n\right) \leqslant C\,\sigma^n\,\ell_c^*(\mathbf{y})\;\forall\, n \in \mathbb{N}_{1,N-1}, \tag{C.8}$$

then condition (C.7b) is satisfied with

$$\gamma_K = C\,\frac{1-\sigma^K}{1-\sigma} \tag{C.9}$$

and $\sup_K \gamma_K < \infty\;\forall\, K \in \mathbb{N}$.

Condition (C.8) is referred to as *exponential controllability*. Under this condition, there exists $\boldsymbol{\tau}(\cdot)$ that generates a trajectory with stage cost exponentially converging to zero in time.

Therefore, the conditions (C.7) are equivalently given as (C.7a) and (C.8). Considering (C.4) (and (C.6), consequently), the conditions of Theorem 4.3 are simplified to

(a) There exist $\alpha_1,\,\alpha_2 \in \mathcal{KL}_\infty$ such that

$$\alpha_1(\|\mathbf{y}-\mathbf{y}_d(k)\|) \leqslant \ell_c^*(\mathbf{y}) \leqslant \alpha_2(\|\mathbf{y}-\mathbf{y}_d(k)\|)\;\forall\,\mathbf{y} \in \mathbb{Y}. \tag{C.10}$$

(b) For all $\mathbf{y} \in \mathbb{Y}$, there exist feasible $\boldsymbol{\tau}^e(\cdot) \in \mathbb{S}^m$, real $C < \infty$ and $\sigma \in (0,1)$ satisfying

$$\ell_c\left(\mathbf{y}_{\boldsymbol{\tau}^e(\cdot)}(j,\mathbf{y}),\boldsymbol{\tau}_j^e\right) \leqslant C\,\sigma^j\,\ell_c^*\left(\mathbf{y}\right),\;\forall\, j \in \mathbb{N}. \tag{C.11}$$

These conditions are equivalent to (C.7a) and (C.8).

The time-varying case addressed in Theorem 4.3 is obtained considering [13, Assumptions 6.29 and 6.30], which indicate that the results above can be straightforwardly extended to time-varying systems.

Glossary

Acronyms

CDPR	Cable-Driven Parallel Robot	10
CWM	Curtain Wall Module	13
DDP	Differential Dynamic Programming	30
DoF	Degrees of Freedom	12
DPP	Dynamic Programming Principle	30
EMPC	Economic MPC	37
FK	Forward Kinematic	15
GPC	Generalized Predictive Control	32
IK	Inverse Kinematic	15
JSC	Joint Space Control	17
KKT	Karush–Kuhn–Tucker conditions	24
LP	Linear Programming	22
LQG	Linear Quadratic Gaussian	29
LQR	Linear Quadratic Regulator	20
LMPC	Linear Model Predictive Control	67
MPC	Model Predictive Control	27
NLP	Nonlinear Programming	89
NMPC	Nonlinear MPC	33
OCP	Optimal Control Problem	28
OE	Orientation Error	81
OSC	Operations Space Control	17
QP	Quadratic Programming	24
RHC	Receding Horizon Control	32
SMC	Sliding Mode Control	20
SQP	Sequential Quadratic Programming	89
TDA	Tension Distribution Algorithm	22
TE	Translation Error	81
WCW	Wrench Closure Workspace	11

Sets

$\mathbb{N}$ Strictly positive integers 68
$\mathbb{N}_0 = \mathbb{N} \cup \{0\}$ 68
$\mathbb{N}_{a,b} = \{k \in \mathbb{N}_0 \mid a \leqslant k \leqslant b\}$ 68
$\mathbb{R}_+$ Strictly positive real numbers 88
$\mathbb{R}_+^0 = \mathbb{R}_0 \cup \{0\}$ 88
$\mathbb{R}_+^0$ $\{\mathbf{v} \in \mathbb{R}^{n_v} \mid v_i > 0 \; \forall \; i \in \mathbb{N}_{1,n_v}\}$ 88
$\mathbb{S}^{n_s}$ Infinite sequences of vectors with length n_s.
For a given sequence $\mathbf{s}(\cdot) \in \mathbb{S}^{n_s}$, the k^{th} vector of $\mathbf{s}(\cdot)$ is denoted by $\mathbf{s}_k = \mathbf{s}(k) \in \mathbb{R}^{n_s}$ 68
$\mathbb{S}_{n_{s2}}^{n_{s1}}$ Sequences of n_{s2} vectors with length n_{s1}. 68
$\mathbb{U}$ Set of vectors of feasible cables tensions 88
$\mathbb{U}^{n_u}$ Sequences of n_u vectors of feasible cable tensions 88
$\mathbb{X}$ Set of admissible poses 88
$\mathbb{Y}$ Set of admissible states 88

Variables

A_i Position of the i^{th} drawing pulley 43
$\mathbf{a}_i$ Position vector corresponding to the point A_i 43
B_i Position of the i^{th} attachment point 43
$\mathbf{b}_i$ Position vector of the point B_i 43
$\mathbf{C}$ Coriollis and centripetal force matrix 70
Δt Sampling period 70
$\mathbf{f}$ Wrench applied on the mobile platform 69
$\mathbf{g}$ Vector of gravitational forces 70
$\boldsymbol{\gamma}$ Vector of predicted states within the prediction horizon 71
$\boldsymbol{\gamma}_d$ Vector of desired states within the prediction horizon 72
h_c Control horizon 71
h_p Prediction horizon 71
$\mathbf{I}$ Identity matrix with dimensions suitably chosen 69
$\mathbf{J}$ Jacobian matrix 45
$\boldsymbol{l}$ Cable lengths vector 43
m Number of cables 43

Functions

Classes of Comparison Functions

$\mathcal{K}$:= $\{\alpha \in \mathbb{R}_0^+ \to \mathbb{R}_0^+ \mid \alpha$ continuous, strictly increasing and $\alpha(0) = 0\}$ 88

$\mathcal{L}$:= $\{\delta \in \mathbb{R}_0^+ \to \mathbb{R}_0^+ \mid \delta$ continuous, strictly decreasing and $\lim_{t\to\infty} \delta(t) = 0\}$ 88

$\mathcal{K}_\infty$:= $\{\alpha \in \mathcal{K} \mid \alpha$ unbounded $\}$ 88

$\mathcal{KL}$:= $\{\beta \in \mathbb{R}_0^+ \times \mathbb{R}_0^+ \to \mathbb{R}_0^+ \mid \beta(r,\cdot) \in \mathcal{L}$ and $\beta(\cdot,t) \in \mathcal{K}\}$ 88

Bibliography

[1] Elizabeth Gibney. Nature - Gigantic Chinese telescope opens to astronomers worldwide. https://www.nature.com/articles/d41586-019-02790-3, 2019. vii, 11

[2] Rui Yao, Xiaoqiang Tang, Jinsong Wang, and Peng Huang. Dimensional optimization design of the four-cable-driven parallel manipulator in FAST. *IEEE/ASME Transactions on Mechatronics*, 15(6):932–941, 2010. vii, 10, 11, 16

[3] Marc Gouttefarde, Jean François Collard, Nicolas Riehl, and Cédric Baradat. Geometry Selection of a Redundantly Actuated Cable-Suspended Parallel Robot. *IEEE Transactions on Robotics*, 31(2):501–510, 2015. vii, 10, 11

[4] Haption. Inca. https://www.haption.com/en/products-en/inca-en.html, 2020. vii, 10, 11

[5] Philipp Miermeister, Maria Lächele, Rainer Boss, Carlo Masone, Christian Schenk, Joachim Tesch, Michael Kerger, Harald Teufel, Andreas Pott, and Heinrich H Bülthoff. The CableRobot simulator large scale motion platform based on Cable Robot technology. In *IEEE International Conference on Intelligent Robots and Systems*, pages 3024–3029. IEEE, oct 2016. vii, 11, 12

[6] James Albus, Roger Bostelman, and Nicholas Dagalakis. The NIST robocrane. *Journal of Robotic Systems*, 10(5):709–724, 1993. vii, 10, 17, 18

[7] Shiqing Fang, Daniel Franitza, Marc Torlo, Frank Bekes, and Manfred Hiller. Motion control of a tendon-based parallel manipulator using optimal tension distribution. *IEEE/ASME Transactions on Mechatronics*, 9(3):561–568, 2004. vii, 18, 19, 22, 25, 53

[8] Johann Lamaury and Marc Gouttefarde. Control of a Large Redundantly Actuated Cable-Suspended Parallel Robot. In *IEEE International Conference on Robotics and Automation*, page 211. IEEE, 2013. vii, viii, x, 19, 20, 25, 26, 27, 37, 50, 53, 82, 83

[9] Clement Gosselin and Martin Grenier. On the determination of the force distribution in overconstrained cable-driven parallel mechanisms. *Meccanica*, 46(1):3–15, 2011. viii, 23

[10] Richard Bellman. *Dynamic Programming*. Princeton University Press, Princeton, 1957. viii, 30, 31

[11] Andreas Pott. *Cable-Driven Parallel Robots: Theory and Application*. Springer International Publishing, 1st edition, 2018. 5

[12] European Union. Hephaestus Project. https://www.hephaestus-project.eu/, 2016. 5, 13

[13] Lars Grüne and Jürgen Pannek. Nonlinear model predictive control. In *Nonlinear Model Predictive Control*, pages 45–69. Springer, 2017. 7, 29, 30, 33, 36, 37, 88, 95, 113, 116, 153, 154, 155

[14] Lawrence L Cone. Skycam: an Aerial Robotic Camera System. *Byte*, 10(10):122, 1985. 10

[15] Casey Lambert, Meyer Nahon, and Dean Chalmers. Implementation of an aerostat positioning system with cable control. *IEEE/ASME Transactions on Mechatronics*, 12(1):32–40, 2007. 10

[16] Makoto Sato. Development of string-based force display: SPIDAR. In *8Th International Conference on Virtual Systems and Multimedia*, 2002. 10

[17] Martin J D Otis, Thien-Ly Nguyen-Dang, Thierry Laliberte, Denis Ouellet, Denis Laurendeau, and Clement Gosselin. Cable tension control and analysis of reel transparency for 6-dof haptic foot platform on a cable-driven locomotion interface. *International Journal of Electrical and Electronics Engineering*, 3(1):16–29, 2009. 10

[18] Martin J D Otis, Marielle Mokhtari, Charles du Tremblay, Denis Laurendeau, François-Michel De Rainville, and Clement Gosselin. Hybrid control with multicontact interactions for 6 DOF haptic foot platform on a cable-driven locomotion interface. In *2008 Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pages 161–168. IEEE, 2008. 10

[19] S Kawamura, W Choe, S Tanaka, and S R Pandian. Development of an ultrahigh speed robot FALCON using wire drive system. *Proceedings - IEEE International Conference on Robotics and Automation*, 1(1):215–220, 1995. 10, 11

[20] Robert Dekker, Amir Khajepour, and S Behzadipour. Design and testing of an ultra-high-speed cable robot. *International Journal of Robotics and Automation*, 21(1):25–34, 2006. 10

[21] Zhaokun Zhang, Zhufeng Shao, and Liping Wang. Optimization and implementation of a high-speed 3-DOFs translational cable-driven parallel robot. *Mechanism and Machine Theory*, 145:103693, 2020. 10

[22] Manfred Hiller, Shiqing Fang, Sonja Mielczarek, Richard Verhoeven, and Daniel Franitza. Design, analysis and realization of tendon-based parallel manipulators. *Mechanism and Machine Theory*, 40(4):429–445, 2005. 10

[23] Sana Baklouti, Eric Courteille, Stéphane Caro, and Mohamed Dkhil. Dynamic and oscillatory motions of cable-driven parallel robots based on a nonlinear cable tension model. *Journal of Mechanisms and Robotics*, 9(6), 2017. 11

[24] Sana Baklouti, Eric Courteille, Philippe Lemoine, and Stéphane Caro. Vibration reduction of Cable-Driven Parallel Robots through elasto-dynamic model-based control. *Mechanism and Machine Theory*, 139:329–345, 2019. 11

[25] Xavier Weber, Loic Cuvillon, and Jacques Gangloff. Active vibration canceling of a cable-driven parallel robot in modal space. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1599–1604. IEEE, 2015. 11

[26] Jeremy Begey, Loic Cuvillon, Maximilian Lesellier, Marc Gouttefarde, and Jacques Gangloff. Dynamic Control of Parallel Robots Driven by Flexible Cables and Actuated by Position-Controlled Winches. *IEEE Transactions on Robotics*, pages 1–8, 2018. 11, 16, 98

[27] Valentin Schmidt and Andreas Pott. Increase of position accuracy for cable-driven parallel robots using a model for elongation of plastic fiber ropes. In *Mechanisms and Machine Science*, volume 43, pages 335–343. Springer, 2017. 11

[28] Marc Gouttefarde and Clement Gosselin. Analysis of the wrench-closure workspace of planar parallel cable-driven mechanisms. *IEEE Transactions on Robotics*, 22(3):434–445, 2006. 12, 88, 98

[29] Marco Carricato and Jean Pierre Merlet. Stability analysis of underconstrained cable-driven parallel robots. *IEEE Transactions on Robotics*, 29(1):288–296, 2013. 12, 22

[30] Ghasem Abbasnejad and Marco Carricato. Direct Geometrico-static Problem of Underconstrained Cable-Driven Parallel Robots With n Cables. *IEEE Transactions on Robotics*, 31(2):468–478, apr 2015. 12, 22

[31] Hussein Hussein, Joao Cavalcanti Santos, and Marc Gouttefarde. Geometric Optimization of a Large Scale CDPR Operating on a Building Facade. In *IEEE International Conference on Intelligent Robots and Systems*, pages 5117–5124, 2018. 13, 147

[32] So Ryeok Oh and Sunil Kumar Agrawal. Generation of feasible set points and control of a cable robot. *IEEE Transactions on Robotics*, 22(3):551–558, 2006. 15

[33] A B Alp and S K Agrawal. Cable suspended robots: design, planning and control. In *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, volume 4, pages 4275–4280. IEEE, 2002. 15

[34] Jason Pusey, Abbas Fattah, Sunil Kumar Agrawal, and Elena Messina. Design and workspace analysis of a 6-6 cable-suspended parallel robot. *Mechanism and Machine Theory*, 39(7):761–778, 2004. 15

[35] Andreas Pott. Influence of pulley kinematics on cable-driven parallel robots. In *Latest Advances in Robot Kinematics*, pages 197–204. Springer, 2012. 16, 42, 43

[36] Valentin Schmidt and Andreas Pott. Implementing Extended Kinematics of a Cable-Driven Parallel Robot in Real-Time. In Tobias Bruckmann and Andreas Pott, editors, *Cable-driven parallel robots*, pages 287–298. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. 16

[37] Joao Cavalcanti Santos, Ahmed Chemori, and Marc Gouttefarde. Redundancy Resolution integrated Model Predictive Control of CDPRs : Concept , Implementation and Experiments. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 3889–3895, Paris, France, 2020. 16

[38] H Max Irvine. *Cable structures*. Dover Publications, 1992. 16

[39] Kris Kozak, Qian Zhou, and Jinsong Wang. Static analysis of cable-driven manipulators with non-negligible cable mass. *IEEE Transactions on Robotics*, 22(3):425–433, 2006. 16

[40] Marc Gouttefarde, Dinh Quan Nguyen, and Cédric Baradat. Kinetostatic analysis of cable-driven parallel robots with consideration of sagging and pulleys. In *Advances in Robot Kinematics*, pages 213–221. Springer, 2014. 16

[41] Hussein Hussein, Marc Gouttefarde, and François Pierrot. Static Modeling of Sagging Cables with Flexural Rigidity and Shear Forces. In Jadran Lenarcic and Vincenzo Parenti-Castelli, editors, *Advances in Robot Kinematics*, pages 310–318, Cham, 2019. Springer International Publishing. 16

[42] D Q Nguyen, Marc Gouttefarde, O Company, and F Pierrot. On the simplifications of cable model in static analysis of large-dimension cable-driven parallel robots. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 928–934, 2013. 16

[43] J-P. Merlet. Some properties of the Irvine cable model and their use for the kinematic analysis of cable-driven parallel robots. *Mechanism and Machine Theory*, 135:271–280, 2019. 16

[44] Mohammad A Khosravi and Hamid D Taghirad. Dynamic modeling and control of parallel robots with elastic cables: Singular perturbation approach. *IEEE Transactions on Robotics*, 30(3):694–704, jun 2014. 16, 20, 56

[45] Hamed Jamshidifar, Saeid Khosravani, Baris Fidan, and Amir Khajepour. Vibration Decoupled Modeling and Robust Control of Redundant Cable-Driven Parallel Robots. *IEEE/ASME Transactions on Mechatronics*, 23(2):690–701, 2018. 16

[46] Herbert Goldstein, Charles Poole, and John Safko. *Classical mechanics*. American Association of Physics Teachers, 3rd edition, 2002. 16

[47] R Chellal, L Cuvillon, and Edouard Laroche. Model identification and vision-based H-infinity position control of 6-DoF cable-driven parallel robots. *International Journal of Control*, 90(4):684–701, 2017. 16, 20, 50, 55

[48] Tej Dallej, Marc Gouttefarde, Nicolas Andreff, Pierre Elie Hervé, and Philippe Martinet. Modeling and vision-based control of large-dimension cable-driven parallel robots using a multiple-camera setup. *Mechatronics*, 61(April):20–36, 2019. 17

[49] Z Zake, F Chaumette, N Pedemonte, and S Caro. Vision-Based Control and Stability Analysis of a Cable-Driven Parallel Robot. *IEEE Robotics and Automation Letters*, 4(2):1029–1036, 2019. 17

[50] Jean Pierre Merlet. Solving the forward kinematics of a Gough-type parallel manipulator with interval analysis. *The International Journal of Robotics Research*, 23(3):221–235, 2004. 17

[51] Alessandro Berti, Jean Pierre Merlet, and Marco Carricato. Solving the direct geometrico-static problem of underconstrained cable-driven parallel robots by interval analysis. *International Journal of Robotics Research*, 35(6):723–739, 2016. 17

[52] Jean Pierre Merlet. On the real-time calculation of the forward kinematics of suspended cable-driven parallel robots. In *2015 IFToMM World Congress Proceedings, IFToMM 2015*, 2015. 17

[53] T Bruckmann, Lars Mikelsons, Thorsten Brandt, Manfred Hiller, and Dieter Schramm. Wire Robots Part I: Kinematics, Analysis & Design. In *Parallel Manipulators, New Developments*, volume 1, pages 109–132. InTech, 2008. 17, 43

[54] Philipp Miermeister, Werner Kraus, and Andreas Pott. Differential kinematics for calibration, system investigation, and force based forward kinematics of cable-driven parallel robots. In *Cable-Driven Parallel Robots*, pages 319–333. Springer, 2013. 17

[55] Andreas Pott. An algorithm for real-time forward kinematics of cable-driven parallel robots. In *Advances in Robot Kinematics: Motion in man and machine*, pages 529–538. Springer, 2010. 17

[56] S Kawamura, H Kino, and Choe Won. High-speed manipulation by using parallel wire-driven robots. *Robotica*, 18(1):13–21, 2000. 18

[57] D Sun, R Lu, J K Mills, and C Wang. Synchronous tracking control of parallel manipulators using cross-coupling approach. *International Journal of Robotics Research*, 25(11):1137–1147, 2006. 18

[58] Weiwei Shang, Bingyuan Zhang, Bin Zhang, Fei Zhang, and Shuang Cong. Synchronization Control in the Cable Space for Cable-Driven Parallel Robots. *IEEE Transactions on Industrial Electronics*, 66(6):4544–4554, 2019. 18, 25, 53, 98

[59] Werner Kraus, Valentin Schmidt, Puneeth Rajendra, and Andreas Pott. System identification and cable force control for a cable-driven parallel robot with industrial servo drives. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 5921–5926. IEEE, 2014. 19, 25, 60

[60] Werner Kraus. *Force control of cable-driven parallel robots*. Stuttgarter Beiträge zur Produktionsforschung, 2016. 19, 60

[61] Richard P Paul. *Robot manipulators: mathematics, programming, and control: the computer control of robot manipulators*. MIT Press, Cambridge. Massachusetts, Massachusetts, 1981. 19

[62] Hassan K Khalil. *Nonlinear control*. Pearson Higher Ed, 2015. 19, 70, 88, 95

[63] Moharam Habibnejad Korayem, H Tourajizadeh, and M Bamdad. Dynamic load carrying capacity of flexible cable suspended robot: Robust feedback linearization control approach. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 60(3-4):341–363, 2010. 19, 20

[64] Mohammad A Khosravi and Hamid D Taghirad. Robust PID control of fully-constrained cable driven parallel robots. *Mechatronics*, 24(2):87–97, 2014. 20

[65] Petar Kokotovic, Hassan K Khali, and John O'reilly. *Singular perturbation methods in control: analysis and design*, volume 25. Siam, 1999. 20

[66] Moharam Habibnejad Korayem and H Tourajizadeh. Maximum DLCC of spatial cable robot for a predefined trajectory within the workspace using closed loop optimal control approach. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 63(1):75–99, 2011. 20

[67] Sigurd Skogestad and Ian Postlethwaite. *Multivariable feedback control: analysis and design*, volume 2. Wiley New York, 2007. 20

[68] Edouard Laroche, R Chellal, Loic Cuvillon, and Jacques Gangloff. A preliminary study for H infinity control of parallel cable-driven manipulators. In *Mechanisms and Machine Science*, volume 12, pages 353–369. Springer, 2013. 20

[69] So Ryeok Oh and Sunil Kumar Agrawal. Nonlinear sliding mode control and feasible workspace analysis for a cable suspended robot with input constraints and disturbances. In *Proceedings of the 2004 American Control Conference*, volume 5, pages 4631–4636. IEEE, 2004. 20

[70] J J Slotine. Sliding controller design for non-linear systems. *International Journal of Control*, 40(2):421–434, 1984. 20, 74

[71] Gamal El-Ghazaly, Marc Gouttefarde, and Vincent Creuze. Adaptive terminal sliding mode control of a redundantly-actuated cable-driven parallel manipulator: CoGiRo. In *Mechanisms and Machine Science*, volume 32, pages 179–200. Springer, 2015. 20, 25, 98

[72] Christian Schenk, Heinrich H Bulthoff, and Carlo Masone. Robust adaptive sliding mode control of a redundant cable driven parallel robot. In *2015 19th International Conference on System Theory, Control and Computing, ICSTCC 2015 - Joint Conference SINTES 19, SACCS 15, SIMSIS 19*, pages 427–434. IEEE, 2015. 20, 25

[73] Bin Zi, B Y Duan, J L Du, and H Bao. Dynamic modeling and active control of a cable-suspended parallel robot. *Mechatronics*, 18(1):1–12, 2008. 20

[74] Hamed Jabbari Asl and Farrokh Janabi-Sharifi. Adaptive neural network control of cable-driven parallel robots with input saturation. *Engineering applications of artificial intelligence*, 65:252–260, 2017. 20

[75] Shuai Li and Damiano Zanotto. Tracking Control of Fully-Constrained Cable-Driven Parallel Robots using Adaptive Dynamic Programming. In *IEEE International Conference on Intelligent Robots and Systems*, pages 6781–6787. IEEE, 2019. 20

[76] Jean-Jacques E ; Weiping Li Slotine. On The Adaptive Control of Robot Manipulators. *The International Journal of Robotics Research*, 6(3):49–59, 1987. 20

[77] Johann Lamaury, Marc Gouttefarde, Ahmed Chemori, and Pierre Elie Herve. Dual-space adaptive control of redundantly actuated cable-driven parallel robots. In *IEEE International Conference on Intelligent Robots and Systems*, pages 4879–4886. IEEE, 2013. 20, 25, 26, 27, 37, 53, 55

[78] Andreas Müller. On the terminology and geometric aspects of redundant parallel manipulators. In *Robotica*, volume 31, pages 137–147, Brooklyn, New York, USA, 2013. ASME. 21

[79] E Sahin Conkur and Rob Buckingham. Clarifying the definition of redundancy as used in robotics. *Robotica*, 15(5):583–586, 1997. 21

[80] Jean Pierre Merlet. *Parallel robots*, volume 128. Springer Science and Business Media, 2006. 21

[81] L Gagliardini, Stéphane Caro, Marc Gouttefarde, and A Girin. Discrete reconfiguration planning for Cable-Driven Parallel Robots. *Mechanism and Machine Theory*, 100:313–337, 2016. 21

[82] Saeed Abdolshah, Damiano Zanotto, Giulio Rosati, and Sunil Kumar Agrawal. Optimizing Stiffness and Dexterity of Planar Adaptive Cable-Driven Parallel Robots. *Journal of Mechanisms and Robotics*, 9(3):31004, 2017. 21

[83] L Gagliardini, Stéphane Caro, Marc Gouttefarde, and A Girin. A reconfiguration strategy for Reconfigurable Cable-Driven Parallel Robots. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1613–1620, 2015. 21

[84] D Q Nguyen, Marc Gouttefarde, O Company, and F Pierrot. On the analysis of large-dimension reconfigurable suspended cable-driven parallel robots. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5728–5735, may 2014. 21

[85] Philipp Miermeister and Andreas Pott. Design of Cable-Driven Parallel Robots with Multiple Platforms and Endless Rotating Axes. In Andrés Kecskeméthy and Francisco Geu Flores, editors, *Interdisciplinary Applications of Kinematics*, pages 21–29, Cham, 2015. Springer International Publishing. 21

[86] Thomas Reichenbach, Philipp Tempel, Alexander Verl, and Andreas Pott. Static Analysis of a Two-Platform Planar Cable-Driven Parallel Robot with Unlimited Rotation. In Andreas Pott and Tobias Bruckmann, editors, *Cable-Driven Parallel Robots*, pages 121–133, Cham, 2019. Springer International Publishing. 21

[87] P Bosscher and I Ebert-Uphoff. Wrench-based analysis of cable-driven robots. In *Proc. of the 2004 IEEE Int. Conf. on Robotics and Automation*, volume 5, pages 4950–4955. IEEE, 2004. 21

[88] Gene H Golub and C F V Loan. Matrix computations, Forth Edition, 2013. 22, 47, 139, 140, 141

[89] Dimitris Bertsimas and John Tsitsiklis. *Introduction to linear optimization*. Athena Scientiific, Massachusetts, 4th editio edition, 1997. 22

[90] J A Snyman and A M Hay. Analysis and Optimization of a Planar Tendon-Driven Parallel Manipulator. In J Lenarčič and C Galletti, editors, *On Advances in Robot Kinematics*, pages 303–312, Dordrecht, 2004. Springer Netherlands. 22

[91] Wei-Jung Shiang, D Cannon, and J Gorman. Optimal force distribution applied to a robotic crane with flexible cables. In *IEEE International Conference on Robotics and Automation*, volume 2, pages 1948–1954. IEEE, 2000. 22, 23

[92] R K Ahuja. Minimax linear programming problem. *Operations Research Letters*, 4(3):131–134, 1985. 23

[93] Per Henrik Borgstrom, Brett L Jordan, Gaurav S Sukhatme, Maxim A Batalin, and William J Kaiser. Rapid computation of optimally safe tension distributions for parallel cable-driven robots. *IEEE Transactions on Robotics*, 25(6):1271–1281, 2009. 23

[94] T Bruckmann, Andreas Pott, and M Hiller. Calculating force distributions for redundantly actuated tendon-based Stewart platforms. *Advances in Robot Kinematics*, pages 403–412, 2006. 23, 24

[95] T Bruckmann, Andreas Pott, Manfred Hiller, and Daniel Franitza. A Modular Controller for Redundantly Actuated Tendon-Based Stewart Platforms. *EuCoMes, The First Conference on Mechanism Science*, pages 1–12, 2006. 23, 24

[96] Martin J D Otis, Simon Perreault, Thien Ly Nguyen-Dang, Patrice Lambert, Marc Gouttefarde, Denis Laurendeau, and Clement Gosselin. Determination and management of cable interferences between two 6-DOF foot platforms in a cable-driven locomotion interface. *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans*, 39(3):528–544, 2009. 23, 24

[97] Mahir Hassan and Amir Khajepour. Analysis of bounded cable tensions in cable-actuated parallel manipulators. *IEEE Transactions on Robotics*, 27(5):891–900, 2011. 23, 24

[98] Hamid D Taghirad and Yousef B Bedoustani. An analytic-iterative redundancy resolution scheme for cable-driven redundant parallel manipulators. *IEEE Transactions on Robotics*, 27(6):1137–1143, 2011. 23, 24

[99] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science and Business Media, 2006. 24, 34, 105, 107, 109, 110, 116, 118, 139, 141, 143

[100] James Demmel. LAPACK: A portable linear algebra library for supercomputers. In *IEEE Control Systems Society Workshop on Computer-Aided Control System Design*, pages 1–7, dec 1989. 24

[101] B Ford, J Bentley, J J du Croz, and S J Hague. The NAG library 'machine'. *Software: Practice and Experience*, 9(1):65–72, 1979. 24, 115

[102] H W Kuhn and A W Tucker. Nonlinear Programming. In *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492, Berkeley, Calif., 1951. University of California Press. 24

[103] Joshua T Bryson, Xin Jin, and Sunil Kumar Agrawal. Optimal Design of Cable-Driven Manipulators Using Particle Swarm Optimization. *Journal of Mechanisms and Robotics*, 8(4):41003, 2016. 24

[104] Andreas Pott, T Bruckmann, and Lars Mikelsons. Closed-form Force Distribution for Parallel Wire Robots. In *Computational Kinematics*, pages 25–34. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. 24

[105] Andreas Pott. An improved force distribution algorithm for over-constrained cable-driven parallel robots. In *Mechanisms and Machine Science*, volume 15, pages 139–146. Springer, 2014. 24

[106] Wen Bin Lim, Song Huat Yeo, and Guilin Yang. Optimization of tension distribution for cable-driven manipulators using tension-level index. *IEEE/ASME Transactions on Mechatronics*, 19(2):676–683, 2014. 24

[107] Hamed Jamshidifar, Amir Khajepour, Baris Fidan, and Mitchell Rushton. Kinematically-Constrained Redundant Cable-Driven Parallel Robots: Modeling, Redundancy Analysis, and Stiffness Optimization. *IEEE/ASME Transactions on Mechatronics*, 22(2):921–930, 2017. 24, 25

[108] Kun Yu, Leng-Feng Lee, and Venkat N Krovi. Simultaneous Trajectory Tracking and Stiffness Control of Cable Actuated Parallel Manipulator. In *Volume 7: 33rd Mechanisms and Robotics Conference, Parts A and B*, pages 55–63, 2009. 24

[109] Mojtaba Azadi, S Behzadipour, and Garry Faulkner. Antagonistic variable stiffness elements. *Mechanism and Machine Theory*, 44(9):1746–1758, 2009. 25

[110] Han Yuan, Eric Courteille, and Dominique Deblaise. Static and dynamic stiffness analyses of cable-driven parallel robots with non-negligible cable mass and elasticity. *Mechanism and Machine Theory*, 85:64–81, mar 2015. 25

[111] Samuel Bouchard, Clement Gosselin, and Brian Moore. On the Ability of a Cable-Driven Robot to Generate a Prescribed Set of Wrenches. *Journal of Mechanisms and Robotics*, 2(1):11010, 2010. 25, 146

[112] Marc Gouttefarde and S Krut. Characterization of Parallel Manipulator Available Wrench Set Facets. *Advances in Robot Kinematics: Motion in Man and Machine*, pages 475–482, 2010. 25, 146, 148

[113] Lars Mikelsons, T Bruckmann, Manfred Hiller, and Dieter Schramm. A real-time capable force calculation algorithm for redundant tendon-based parallel manipulators. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 3869–3874. IEEE, 2008. 25

[114] Marc Gouttefarde, Johann Lamaury, Christopher Reichert, and T Bruckmann. A Versatile Tension Distribution Algorithm for n-DOF Parallel Robots Driven by n + 2 Cables. *IEEE Transactions on Robotics*, 31(6):1444–1457, 2015. 25, 151

[115] Johann Lamaury and Marc Gouttefarde. A Tension Distribution Method with Improved Computational Efficiency. In *Cable-Driven Parallel Robots*, pages 71–85. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. 25

[116] Werner Kraus, Philipp Miermeister, Valentin Schmidt, and Andreas Pott. Hybrid Position/Force Control of a Cable-Driven Parallel Robot with Experimental Evaluation. In Paulo Flores and Fernando Viadero, editors, *New Trends in Mechanism and Machine Science*, pages 553–561, Cham, 2015. Springer International Publishing. 26

[117] Christopher Reichert, Katharina Muller, and T Bruckmann. Robust internal force-based impedance control for cable-driven parallel robots. In *Mechanisms and Machine Science*, volume 32, pages 131–143. Springer, Cham, 2015. 26

[118] Gabriel Meunier, Benoit Boulet, and Mayer Nahon. Control of an overactuated cable-driven parallel mechanism for a radio telescope application. *IEEE Transactions on Control Systems Technology*, 17(5):1043–1054, 2009. 26

[119] Jean Pierre Merlet. On the workspace of suspended cable-driven parallel robots. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2016-June, pages 841–846. IEEE, 2016. 27, 88, 98

[120] Jun-Mu Heo, Sung-Hyun Choi, and Kyoung-Su Park. Workspace analysis of a 6-DOF cable-driven parallel robot considering pulley bearing friction under ultra-high acceleration. *Microsystem Technologies*, 23(7):2615–2627, 2017. 27

[121] Jan Marian Maciejowski. *Predictive control: with constraints*. Pearson education, 2002. 27, 33, 71

[122] E F Camacho and C B Alba. *Model predictive control.* Springer Science and Business Media, 2013. 27, 33

[123] Ernest Bruce Lee and Lawrence Markus. Foundations of optimal control theory. Technical report, Minnesota Univ, 1967. 27

[124] Richard Vinter. *Optimal control*. Springer Science and Business Media, 2010. 28

[125] Dean A Carlson, Alain B Haurie, and Arie Leizarowitz. *Infinite horizon optimal control: deterministic and stochastic systems*. Springer Science & Business Media, 2012. 29

[126] I M Gelfand and S V Fomin. Calculus of variations. *Revised English edition translated and edited by Richard A. Silverman*, 1963. 29

[127] V G Boltyanskiy, Revaz V Gamkrelidze, Y E F MISHCHENKO, and L S Pontryagin. *Mathematical theory of optimal processes*. Interscience Publishers, 1962. 29

[128] Eduardo D Sontag. *Mathematical control theory: deterministic finite dimensional systems*, volume 6. Springer Science and Business Media, 2013. 29

[129] Richard E Bellman and Stuart E Dreyfus. *Applied dynamic programming*. Princeton university press, 1962. 30

[130] S Sen and S J Yakowitz. A quasi-newton differential dynamic programming algorithm for discrete-time optimal control. *Automatica*, 23(6):749–752, 1987. 30

[131] Y Tassa, N Mansard, and E Todorov. Control-limited differential dynamic programming. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1168–1175, 2014. 30

[132] 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. 32, 36

[133] A I Propoı. Application of linear programming methods for the synthesis of automatic sampled-data systems. *Avtomat. i Telemeh*, 24:912–920, 1963. 32

[134] Jacques Richalet, A Rault, J L Testud, and J Papon. Model predictive heuristic control. *Automatica (Journal of IFAC)*, 14(5):413–428, 1978. 32, 33

[135] D W Clarke, C Mohtadi, and P S Tuffs. Generalized predictive control—Part I. The basic algorithm. *Automatica*, 23(2):137–148, 1987. 32

[136] P M Hidalgo and C B Brosilow. Nonlinear model predictive control of styrene polymerization at unstable operating points. *Computers and Chemical Engineering*, 14(4):481–494, 1990. 33, 103

[137] John W Eaton and J B Rawlings. Model-Predictive Control of Chemical Processes. *Chemical Engineering Science*, 47(4):705–720, 1992. 33

[138] Carlos E García, David M Prett, and Manfred Morari. Model predictive control: Theory and practice—A survey. *Automatica*, 25(3):335–348, 1989. 33

[139] M Neunert, C de Crousaz, F Furrer, M Kamel, F Farshidian, R Siegwart, and J Buchli. Fast nonlinear Model Predictive Control for unified trajectory optimization and tracking. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1398–1404, may 2016. 33, 34

[140] R Ginhoux, J Gangloff, M de Mathelin, L Soler, M M A Sanchez, and J Marescaux. Active filtering of physiological motion in robotized surgery using predictive control. *IEEE Transactions on Robotics*, 21(1):67–79, 2005. 33

[141] Jacques A Gangloff and Michel F de Mathelin. High-speed visual servoing of a 6-d.o.f. manipulator using multivariable predictive control. *Advanced Robotics*, 17(10):993–1021, jan 2003. 33

[142] Andrés Vivas and Philippe Poignet. Predictive functional control of a parallel robot. *Control Engineering Practice*, 13(7):863–874, 2005. 33, 34

[143] Shahab Heshmati-Alamdari, Alina Eqtami, George C Karras, Dimos V Dimarogonas, and Kostas J Kyriakopoulos. A self-triggered visual servoing model predictive control scheme for under-actuated underwater robotic vehicles. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 3826–3831, may 2014. 33, 34

[144] R Gonzalez, M Fiacchini, J L Guzman, and T Alamo. Robust tube-based MPC for constrained mobile robots under slip conditions. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 5985–5990, dec 2009. 33, 34

[145] G P Incremona, A Ferrara, and L Magni. MPC for Robot Manipulators With Integral Sliding Modes Generation. *IEEE/ASME Transactions on Mechatronics*, 22(3):1299–1307, jun 2017. 33, 34

[146] C Hsieh and J Liu. Nonlinear model predictive control for wheeled mobile robot in dynamic environment. In *2012 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 363–368, jul 2012. 33, 34

[147] J Nubert, J Köhler, V Berenz, F Allgöwer, and S Trimpe. Safe and Fast Tracking on a Robot Manipulator: Robust MPC and Neural Network Control. *IEEE Robotics and Automation Letters*, 5(2):3050–3057, apr 2020. 33, 34

[148] Z Li, Y Yuan, F Ke, W He, and C Su. Robust Vision-Based Tube Model Predictive Control of Multiple Mobile Robots for Leader–Follower Formation. *IEEE Transactions on Industrial Electronics*, 67(4):3096–3106, apr 2020. 33, 34

[149] K Worthmann, M W Mehrez, M Zanon, G K I Mann, R G Gosine, and M Diehl. Model Predictive Control of Nonholonomic Mobile Robots Without Stabilizing Constraints and Costs. *IEEE Transactions on Control Systems Technology*, 24(4):1394–1406, jul 2016. 33, 34

[150] Romuald Ginhoux, Jacques A Gangloff, Michel F de Mathelin, Luc Soler, and Jacques A Joël Leroy. Model predictive control to cancel repetitive deformations of organs in robotized laparoscopic surgery. *IFAC Proceedings Volumes*, 36(17):431–436, 2003. 33, 34

[151] J B Rawlings and D Q Mayne. *Model predictive control: Theory and design*. Nob Hill Pub., 2009. 33

[152] Lalo Magni, Davide Martino Raimondo, and Frank Allgöwer. Nonlinear model predictive control. *Lecture Notes in Control and Information Sciences*, 2009. 33

[153] James B. Rawlings, David Q. Mayne, and Moritz M. Diehl. *Model Predictive Control: Theory, Computation, and Design*. Cheryl M. Rawlings, 2nd edition, 2019. 34

[154] Lars Grüne. NMPC without terminal constraints. *IFAC Proceedings Volumes*, 45(17):1–13, 2012. 36

[155] Andrea Boccia, Lars Grüne, and Karl Worthmann. Stability and feasibility of state constrained MPC without stabilizing terminal constraints. *Systems & Control Letters*, 72:14–21, 2014. 36

[156] David Mayne. An apologia for stabilising terminal conditions in model predictive control. *International Journal of Control*, 86(11):2090–2095, nov 2013. 36, 37

[157] Nils Altmüller, Lars Grüne, and Karl Worthmann. Performance of NMPC Schemes without Stabilizing Terminal Constraints. In Moritz Diehl, Francois Glineur, Elias Jarlebring, and Wim Michiels, editors, *Recent Advances in Optimization and its Applications in Engineering*, pages 289–298, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. 36

[158] J B Rawlings, Dennis Bonne, John B Jørgensen, Aswin N Venkat, and Sten Bay Jørgensen. Unreachable setpoints in model predictive control. *IEEE Transactions on Automatic Control*, 53(9):2209–2215, 2008. 37

[159] M Diehl, Rishi Amrit, and J B Rawlings. A Lyapunov function for economic optimizing model predictive control. *IEEE Transactions on Automatic Control*, 56(3):703–707, 2011. 37

[160] Rishi Amrit, J B Rawlings, and David Angeli. Economic optimization using model predictive control with a terminal cost. *Annual Reviews in Control*, 35(2):178–186, 2011. 37

[161] David Angeli, Rishi Amrit, and J B Rawlings. On average performance and stability of economic model predictive control. *IEEE Transactions on Automatic Control*, 57(7):1615–1626, 2012. 37

[162] Lars Grüne and Anastasia Panin. On non-averaged performance of economic MPC with terminal conditions. *Proceedings of the IEEE Conference on Decision and Control*, 54rd IEEE(Cdc):4332–4337, 2015. 37

[163] Matthias A Müller and Lars Grüne. Economic model predictive control without terminal constraints for optimal periodic behavior. *Automatica*, 70:128–139, 2016. 37, 131

[164] Lars Grüne. Economic receding horizon control without terminal constraints. *Automatica*, 49(3):725–734, 2013. 37, 131

[165] Jan C Willems. Dissipative dynamical systems part I: General theory. *Archive for Rational Mechanics and Analysis*, 45(5):321–351, 1972. 37

[166] Matthew Ellis, Helen Durand, and Panagiotis D Christofides. A tutorial review of economic model predictive control methods. *Journal of Process Control*, 24(8):1156–1178, 2014. 37, 131

[167] Johannes Kohler, Matthias A Muller, and Frank Allgower. On Periodic Dissipativity Notions in Economic Model Predictive Control. *IEEE Control Systems Letters*, 2(3):501–506, 2018. 37, 131

[168] Johannes Kohler, Matthias A Muller, and Frank Allgower. Nonlinear reference tracking: An economic model predictive control perspective. *IEEE Transactions on Automatic Control*, 64(1):254–269, 2019. 37, 131

[169] Zihang Dong and David Angeli. Analysis of economic model predictive control with terminal penalty functions on generalized optimal regimes of operation. *International Journal of Robust and Nonlinear Control*, 28(16):4790–4815, nov 2018. 37, 131

[170] Tim Martin, Philipp N. Köhler, and Frank Allgöwer. Dissipativity and economic model predictive control for optimal set operation. *Proceedings of the American Control Conference*, 2019-July:1020–1026, 2019. 37, 131

[171] Mikhail Katliar, Jörg Fischer, Gianluca Frison, M Diehl, Harald Teufel, and Heinrich H Bülthoff. Nonlinear Model Predictive Control of a Cable-Robot-Based Motion Simulator. *IFAC-PapersOnLine*, 50(1):9833–9839, 2017. 37, 38

[172] Gianluca Frison, Hans Henrik Brandenborg Sørensen, Bernd Dammann, and John Bagterp Jørgensen. High-performance small-scale solvers for linear model predictive control. In *2014 European Control Conference (ECC)*, pages 128–133. IEEE, 2014. 37

[173] Hans Joachim Ferreau, Christian Kirches, Andreas Potschka, Hans Georg Bock, and M Diehl. qpOASES: a parametric active-set algorithm for quadratic programming. *Mathematical Programming Computation*, 6(4):327–363, dec 2014. 37

[174] Ronghuai Qi, Mitchell Rushton, Amir Khajepour, and William W Melek. Decoupled modeling and model predictive control of a hybrid cable-driven robot (HCDR). *Robotics and Autonomous Systems*, 118:1–12, 2019. 38

[175] Valentin Lorenz Schmidt. *Modeling Techniques and Reliable Real-Time Implementation of Kinematics for Cable-Driven Parallel Robots using Polymer Fiber Cables*. University of Stuttgart, 2017. 42

[176] William H Press, Saul A Teukolsky, William T Vetterling, and Brian P Flannery. Numerical Recipes in C ++ The Art of Scientific Computing Second Edition. Technical report, Cambridge University Press, 1992. 47, 103, 110, 139, 140

[177] Charles W Wampler. Forward displacement analysis of general six-in-parallel SPS (Stewart) platform manipulators using soma coordinates. *Mechanism and Machine Theory*, 31(3):331–337, 1996. 49

[178] Manfred Husty, Sonja Mielczarek, and Manfred Hiller. A redundant spatial Stewart-Gough platform with a maximal forward kinematics solution set. In *Advances in Robot Kinematics*, pages 147–154. Springer, 2002. 49

[179] C Shiqi, Z Jiyun, L Bingwen, X Zhan, and F Yanqiu. Nonlinear dynamic model and governing equations of low speed and high load planetary gear train with respect to friction. In *IEEE International Conference on Consumer Electronics, Communications and Networks*, pages 3274–3278, 2011. 52

[180] D C H Yang and J Y Lin. Hertzian Damping, Tooth Friction and Bending Elasticity in Gear Impact Dynamics. *ASME Journal of Mechanisms, Transmissions, and Automation in Design*, 109(2):189–196, jun 1987. 52

[181] Joseph Edward Shigley. Mechanical Engineering Design, 1972. 56

[182] Joao Cavalcanti Santos, Ahmed Chemori, and Marc Gouttefarde. Model Predictive Control Applied to Large-Dimension Cable-Driven Parallel Robots. In *Cable-driven parallel robots*, 2019. 74

[183] BRAD PADEN and RAVI PANJA. Globally asymptotically stable 'PD+' controller for robot manipulators. *International Journal of Control*, 47(6):1697–1712, jun 1988. 74

[184] Lorenzo Gagliardini, Marc Gouttefarde, and Stéphane Caro. Determination of a Dynamic Feasible Workspace for Cable-Driven Parallel Robots. In Jadran Lenarčič and Jean-Pierre Merlet, editors, *Advances in Robot Kinematics*, pages 361–370. Springer International Publishing, Cham, 2016. 88, 98

[185] Andreas Pott and Werner Kraus. Determination of the wrench-closure translational workspace in closed-form for cable-driven parallel robots. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2016-June, pages 882–887. IEEE, 2016. 88, 98

[186] C Henry Edwards, David E Penney, and David T Calvis. *Differential equations and boundary value problems*. Pearson Education Limited, 2016. 102

[187] Ernst Hairer, Syvert P. Nørsett, and Gerhard Wanner. *Solving Ordinary Differential Equations I, Nonstiff Problems*. Springer, Berlin, 2nd edition, 1993. 103

[188] Sergey Bochkanov and Vladimir Bystritsky. ALGLIB-a cross-platform numerical analysis and data processing library. *ALGLIB Project. Novgorod, Russia*, 2011. 115

[189] I Puaut. Real-time performance of dynamic memory allocation algorithms. In *Proceedings 14th Euromicro Conference on Real-Time Systems. Euromicro RTS 2002*, pages 41–49, 2002. 136

[190] Beckhoff. *TwinCAT 3 - C++ Manual*. Beckhoff, 1.7 edition, 2019. 136, 137

[191] Gaël Guennebaud, Benoît Jacob, and Others. Eigen v3. http://eigen.tuxfamily.org, 2010. 136