TECHNISCHE UNIVERSITÄT MÜNCHEN
Lehrstuhl für Robotik, Künstliche Intelligenz und Echtzeitsysteme

Automatic Design of Controllers for Modular Reconfigurable Robot Manipulators

Andrea Giusti

Vollständiger Abdruck der von der Fakultät der Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Dr.-Ing. Alin Albu-Schäffer
Prüfer der Dissertation:	1. Prof. Dr.-Ing. Matthias Althoff
	2. Univ.-Prof. Dr.-Ing. (Univ. Tokio) Martin Buss

Die Dissertation wurde am 30.01.2018 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 01.06.2018 angenommen.

Abstract

Modular robotic systems offer the possibility to create a huge range of different robots from arbitrary assembly of modules. These robots will have different kinematics and dynamics, and generating model-based controllers for all possible assemblies of modules *a priori* is impractical. This is especially critical when no assumptions are made on the geometry and number of modules used. Decentralized control, the approach used in most previous work, often leads to complicated control concepts, or to simple control schemes that do not provide global asymptotic stability without knowledge of the overall system dynamics. This thesis addresses the control problem for modular reconfigurable robot manipulators differently, by designing centralized model-based controllers *automatically.* Three contributions are presented: 1) a framework for generating centralized, model-based control for arbitrary assemblies of modules, 2) the enhancement of this framework for providing robustness to model uncertainty and input disturbances, and 3) accounting for joint elasticity in modern compliant actuators when maintaining accurate tracking.

The framework centers around the storage of data within robot modules. Upon assembly, these data are collected and used to generate a model, and hence a controller, automatically. To provide robustness against uncertainty in model parameters and in the control input, interval arithmetic is used to compute worst-case closed-loop perturbations online. These are used as feedback to guarantee closed-loop stability and user-defined tracking performance despite disturbances.

Finally, inherent compliance in actuators is gaining popularity as a way to limit damage to the robot and increase safety for nearby humans. To maintain accurate tracking capabilities even when possible joint elasticity is introduced by modern compliant actuators, a new global tracking control scheme has been introduced. In this approach, the basic idea is to combine the efficient numerical computability of the classical inverse-dynamics control scheme with the robustness typical of passivity-based controllers with respect to model uncertainties.

The framework and control approaches demonstrated theoretically and experimentally in this thesis show that robust, centralized control for modular robots is feasible and practical, even when compliant actuators are employed.

Zusammenfassung

Modulare Robotersysteme bieten die Möglichkeit, aus einer Sammlung von Modulen eine große Bandbreite an unterschiedlichen Robotern zu erstellen. Die Kinematik und Dynamik dieser Roboter wird unterschiedlich sein und es ist nicht praktikabel, eine modellbasierte Regelung für alle möglichen Konstellationen von Modulen a priori zu generieren. Besonders gilt dies, wenn keine Annahmen über die Geometrie und Anzahl an Modulen getroffen werden. Dezentrale Regelung, der überwiegend verfolgte Ansatz in der bisherigen Forschung, führt oft zu komplizierten Regelungskonzepten oder einfachen Reglern, welche ohne Kenntnis der Systemdynamik keine globale, asymptotische Stabilität garantieren können. Diese Dissertation befasst sich mit dem Problem der Regelung modularer, wiederkonfigurierbarer, robotischer Manipulatoren auf andere Weise, indem zentrale, modellbasierte Regler *automatisch* entworfen werden. Drei Beiträge werden vorgestellt: 1) ein Ansatz zur automatischen Herleitung der Regelung arbiträrer Konstellationen von Modulen, 2) die Erweiterung dieses Ansatzes, um Robustheit gegen Modellunsicherheit und Eingangsstörungen zu gewährleisten und 3) Methoden zur präzisen Bahnverfolgung trotz Elastizität in modernen, nachgiebigen Aktoren.

Im Mittelpunkt des Frameworks steht die Speicherung von Daten in den Modulen. Nach der Montage werden diese Daten gesammelt und zur automatischen Herleitung eines Modells und damit eines Reglers verwendet. Um Robustheit gegen Unsicherheit der Modellparameter und des Eingangs zu gewährleisten, werden die Worst-Case-Störungen des geschlossenen Regelkreises mittels Intervallarithmetik berechnet. Diese dienen als Feedback, um sowohl die Stabilität des geschlossenen Regelkreises, als auch die benutzerdefinierte Performanz der Bahnverfolgung trotz Störungen zu garantieren.

Mechanische Nachgiebigkeit in Aktoren wird in der Robotik immer häufiger eingesetzt, um Maschinenschäden zu begrenzen und die Sicherheit von Menschen in der Umgebung zu verbessern. Um präzise Bahnverfolgung trotz möglicher Elastizität in modernen, nachgiebigen Gelenken gewährleisten zu können, wird ein neues, globales Bahnverfolgungskonzept vorgestellt. Die Grundidee dieses Ansatzes ist es, die effiziente numerische Berechenbarkeit der klassischen invers-dynamischen Regler mit der Robustheit zu kombinieren, die für passivitätsbasierte Regelung hinsichtlich Modellunsicherheit typisch ist.

Dieser Ansatz und die in dieser Dissertation theoretisch und experimentell demonstrierten Regelungskonzepte zeigen, dass robuste, zentrale Regelung modularer Roboter praktikabel ist und auch angwendet werden kann, wenn nachgiebige Aktoren zum Einsatz kommen.

Acknowledgements

First of all, I would like to thank my Ph.D. advisor Prof. Dr.-Ing. Matthias Althoff, for the trust, continuous support, and extreme dedication he gave me for the whole period I spent under his supervision. The results achieved in these years would not have been possible without his motivation and guidance. I would also like to thank him for the great working environment he created within the Cyber-Physical Systems (CPS) research group, which I feel extremely lucky to have been a part of. Thanks also to all the components of the CPS research group for the inspiring and friendly working environment we created at the Chair of Robotics and Embedded Systems of the Technische Universität München.

I would also like to thank Dr. Nikos Tsagarakis and Dr.-Ing. Jörn Malzahn for their support for the time spent at the Italian Institute of Technology (IIT) during my research secondments, and for the related research collaborations that subsequently arose. Thanks also to all the labmates at IIT for their contribution to the friendly and enthusiastic environment I found in Genova.

A special thanks goes to all other early-stage researchers of the EC FP7 SMART-E project: Aaron, Alex, Constantin, Esra, Martijn, Mateo, Stefan, Stefania, Stefano, Yasmin, Roy, Taimoor, and Saber, for the memorable times we had during the activities of SMART-E since its beginning. In particular, thanks to Martijn, Esra, and Aaron for the fruitful collaborations we made in these years.

Finally, I would like to thank my family for the support and love I received, especially when I had hard times. In particular, I would not have been able to truly enjoy the choice of pursuing a research path abroad and to pass all the difficulties that this choice involved so far without the unconditional support and love I received from Elena, to whom I address my greatest thank you.

Trento, January 2018 Andrea Giusti

Contents

1 Introduction 1

1.1 Introduction to the Control Problem 5

1.2 Author's Contribution 9

1.3 Outline of the Thesis 10

2 A Framework for Automatic Centralized Controller Design 13

2.1 Introduction and State of the Art 13

2.1.1 Decentralized Control 13

2.1.2 Automatic Modeling and Centralized Control 15

2.1.3 Network Solutions 17

2.2 Formulation of the Control Problem 18

2.3 Kinematic Modeling using Module Data 19

2.3.1 An Extended D-H Convention 20

2.3.2 Definition of the Module Data for Kinematics 21

2.3.3 From the Module Data to the Kinematic Model 25

2.4 Dynamic Modeling using Module Data 29

2.4.1 Definition of the Module Data for Dynamics 31

2.4.2 From the Module Data to the Dynamical Model 33

2.5 Motion Control 35

2.5.1 Joint-Space Control 36

2.5.2 Kinematic Control 39

2.6 Experimental Application 42

2.6.1 Description of the Modular Robot Test Bed 43

2.6.2 Derivation of the Module Data 43

2.6.3 Validation of the Automatically-Generated Models 47

2.6.4 Evaluation of the Control Performance 48

2.7 Summary 51

3 Automatic Design of Robust Controllers 55

3.1 Introduction and State of the Art 55

3.1.1 Discontinuous, Smoothed, and Continuous Schemes 56

3.1.2 Removal of Implementation Difficulties 57

3.2 Formulation of the Control Problem 58

3.3 Preliminaries on Interval Arithmetic 60

3.4 Interval-Arithmetic-Based Robust Control 61

3.4.1 Derivation of the Controllers 62

3.4.2 Efficient Computation of the Controllers 72

3.4.3 Application to Modular Robot Manipulators 74

3.5 Performance Evaluation using Simulations 75

3.5.1 Classical Robust Control 78

3.5.2 r-$\boldsymbol{\alpha}$ Tracking Control 80

3.5.3 Performance Comparison 81

3.6 Experimental Application 86

3.6.1 Interval-Arithmetic Inverse-Dynamics Control 87

3.6.2 Interval-Arithmetic Passivity-Based Control 89

3.7 Summary 92

4 Control Design with Elastic-Joint Modules 95

4.1 Introduction and State of the Art 95

4.1.1 Regulation and Tracking 96

4.1.2 Efficient Implementability and Robustness 98

4.2 Formulation of the Control Problem 99

4.3 Regulation 100

4.4 Tracking 101

4.4.1 Feedforward/Feedback Tracking Control 102

4.4.2 Inverse-Dynamics Tracking Control 102

4.4.3 Passivity-Based Tracking Control 103

4.4.4 Combined Inverse-Dynamics/Passivity-Based Tracking Control 105

4.5 Efficient Computation of the Control Commands 108

4.6 Robustness of the Combined Inverse-Dynamics/Passivity-Based Control 110

4.7 Performance Evaluation using Simulations 114

4.7.1 Simulation Test Bed 114

4.7.2 Estimation of Joint Accelerations and Jerks 115

4.7.3 Simulation Results 116

4.8 Experimental Application 119

4.8.1 2R Reconfigurable Elastic-Joint Robot 120

4.8.2 7R Mixed Rigid/Elastic Joint Robot 124

4.9 Summary 130

5 Conclusion and Future Directions 133

A Efficient Recursive Algorithms 137

A.1 Standard and Modified Recursive Newton-Euler Algorithms 137

A.2 Interval-Arithmetic-Based Recursive Newton-Euler Algorithm 140

A.3 Computation of the Geometric Jacobian and its Derivative 142

B Test Trajectories 145

References 149

Notations

Conventions

Unless stated otherwise, the following conventions hold: scalar quantities are denoted by plain letters, vectors are denoted by bold lower-case letters, and matrices are denoted by bold upper-case letters. Further, arguments of functions are sometimes omitted to increase readability (e.g., $\mathbf{M}(\mathbf{q}(t)) \rightarrow \mathbf{M}$). More specific conventions are introduced directly in the chapters in which they are employed.

Abbreviations

D-H	Denavit-Hartenberg
DLR	German Aerospace Center
FPGA	field programmable gate array
ffwd-ID/PB	feedforward-inverse-dynamics/passivity-based
IA	interval arithmetic
ID	inverse-dynamics
IAB	interval arithmetic based
$IANEA(\dots)$	interval arithmetic recursive Newton-Euler algorithm
LWA(s)	light-weight arm(s)
PB	passivity-based
PBAFC	passivity-based control with adaptive friction compensation
PD	proportional-derivative
PID	proportional-integral-derivative
NE	Newton-Euler
$NEA(\dots)$	recursive Newton-Euler algorithm

$NEA^*(\dots)$	modified recursive Newton-Euler algorithm
RMMS	reconfigurable modular manipulator system
VDC	virtual decomposition control
module data	set of data that characterize an arbitrary module for kinematics and dynamics
arm-link	link of a modular robot arm assembled from modules
DH	table of the D-H parameters of a robot arm
DynPar	table of the dynamical parameters of a robot arm
ModRob	data structure containing the module data of an assembled modular robot manipulator
S_χ/C_χ	$sin(\chi)/cos(\chi)$
$T_\chi(\cdot)/R_\chi(\cdot)$	homogeneous transformation matrix of the elementary translation/rotation along/around the χ axis
$\lambda_{\min}(\mathbf{A})/\lambda_{\max}(\mathbf{A})$	minimum/maximum eigenvalue of a matrix $\mathbf{A}$

Chapter 1

Introduction

Within the active research areas in robotics, the domain of modular and reconfigurable robots is one of the most promising and timely exploitable for enhancing flexibility of robotic applications. Modular robots are mechatronic systems with some degree of autonomy which can be assembled from a set of interchangeable modules [52]. The modules can be quickly exchanged and rearranged to generate assemblies with different kinematic chains (see Figure 1.1). Their reconfigurability allows a user to quickly assemble task specific robots, which are especially useful in flexible environments, where the task or the surroundings of the robots may frequently change. This is also useful when the environment in which the robots operate cannot be known a priori. Modular robots introduce several beneficial implications such as easy maintenance, robustness, and versatility. Some exemplary scenarios that clarify these benefits are pitched in the following.

In industry, modular robots can be significantly advantageous for both robot manufacturers, by providing a large portfolio of different robots with a few standard modules, and for customers, who can adapt their robots for different applications without purchasing new ones. Another immediately apparent benefit of this technology lies in the reduced down-time that can be guaranteed for automated cells in the event of faults. In fact, modules that include faulty components can be quickly exchanged for new ones, without needing to decommission the whole robot. Similar benefits can be considered for other scenarios in which the fix to be executed can not be easily or safely performed by humans. In these cases, modular robots introduce simplicity in the modules' exchange, in the disposal of the faulty ones, and in the quick recovery of the nominal functionality. In addition, the reconfigurability of these systems can be exploited for enabling tasks that are unforeseeable a priori in flexible environments or for optimizing the

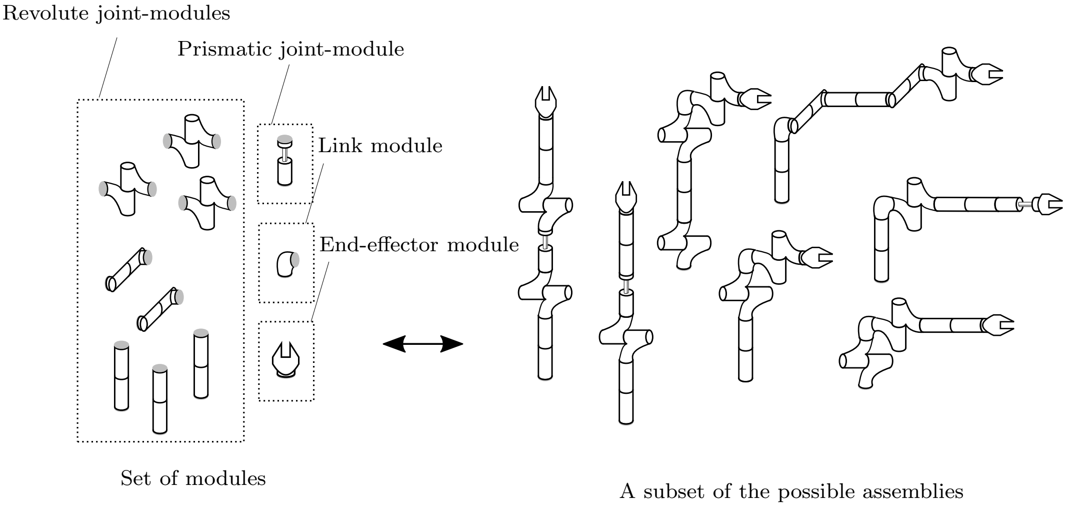


Figure 1.1: Illustration of a modular robot setup.

robotic structure e.g., to minimize energy consumption while performing repetitive tasks or to minimize task execution time. Besides being useful in flexible manufacturing, the abovementioned aspects are particularly advantageous in unstructured environments typical of space operations and exploration [117], automated agriculture [11], and search and rescue scenarios [112].

In recent decades, several modular robotic manipulators have been proposed by research groups worldwide. Early developments of such a robotic concept trace back to the '80s, when the author in [114] introduced the concept of a manipulator that could be assembled in various configurations starting from a set of modules (mechanical joints and links). An extension of this concept was proposed in [85, 93], where the authors introduced the Reconfigurable Modular Manipulator System (RMMS). This platform included configuration independent control software and self-contained modules (see Figure 1.2a). Such work represented a fundamental step forward, since it proved the concept of a modular, reconfigurable, and quickly deployable robotic system. Other robotic solutions based on this concept were subsequently presented. For example, a platform composed of two modular reconfigurable arms for experiments of collaborative robot operations was proposed in [51]. Another platform was introduced by T. Matsumaru from Toshiba Corp. in [72], whose design has been focused on the ease to use and re-assembly for a user. More recently, a spring-assisted modular robot, which shows enhanced manipulation and payload capabilities, was proposed in [66].

Among the commercially available modular robots are the Light Weight Arms (LWAs) 4P

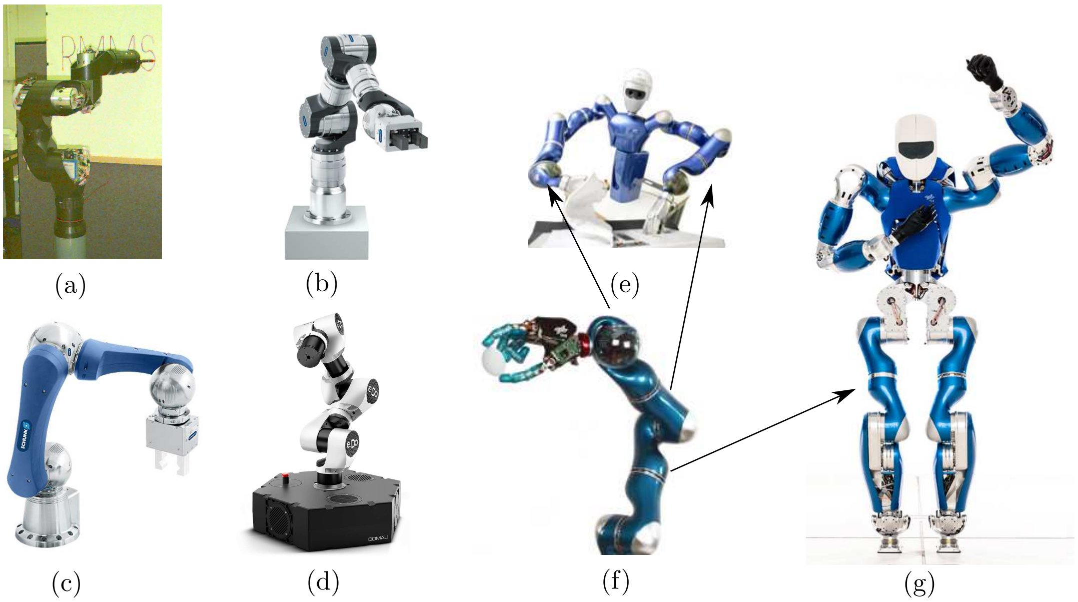


Figure 1.2: Examples of modular robot manipulators developed so far (a, b, c, and d) and robots with strong hardware modularity (f, e, and g). In particular, the figure shows how the modules of the LBR III arm developed at the DLR (f) have been used to build humanoid platforms such as (e) and (g). Sources: Carnegie Mellon University [85,93] for (a), SCHUNK GmbH & Co. KG for (b) and (c), Comau SpA for (d), and German Aerospace Center (DLR) for (e), (f), (g).

and 4D from SCHUNK GmbH & Co. KG (see Figures 1.2b and 1.2c) and the "e.Do" platform from the robot manufaturer Comau SpA (see Figure 1.2d). Further, despite their design as fixed-configuration arms, a strong hardware modularity can be observed in other popular and commercially available robots such as the KUKA light weight arm, from technology developed at the German Aerospace Center (DLR) [47]. The arm originally developed at the DLR has a strong hardware modularity that also allows usage of the modules for realizing humanoid platforms, e.g., [42] (see Figures 1.2f, 1.2e, and 1.2g). Modern modular actuation units with variable stiffness capabilities based on [25] have interestingly been proposed by QBrobotics®. These units can be used for realizing different assemblies by means of link modules of different geometries. Their variable stiffness actuators can be exploited for creating robots which mimic the compliant characteristics of humans.

Modular robots provide flexibility at the price of an increased design complexity. For example, mechanical design optimization cannot be performed considering the complete robot since arbitrarily different assemblies can be composed. The reconfigurability of the systems introduces specific communication requirements for control and data exchange between modules.

Special attention should be paid to the connectors that must be able to properly stand joint forces and torques without being excessively bulky. Besides the above-mentioned electromechanical design aspects, significant difficulties also arise in the use of these robots. Nontrivial aspects that complicate the real-world exploitation of these systems are the assembly selection and the motion-control problems. While the first aspect may prevent the benefits of this class of robots from being properly exploited, the second may lead to motion performances that are significantly inferior with respect to classical fixed-configuration robots.

The assembly selection problem is non-trivial due to the complexity of the combinatorial problem which depends on the task, the set of modules available, and a possibly present cost function to be minimized. This problem has attracted several reasearchers so far and several approaches have been proposed (see e.g., [52] for a complete review). One which considers kinematics only and uses a penalty function for less-likely combinations of modules is proposed in [22]. The authors of [116] propose a task-based configuration synthesis approach to minimize the number of degrees-of-freedom of the arm. A recently proposed time-efficient approach, which is based of subsequent elimination of compositions of modules by means of increasingly complex tests, can be found in [132]. These tests involve kinematic reachability, static force constraints, and obstacole and self collision avoidance. This approach has been subsequently enhanced by considering dynamics and cost-optimal solutions in [53].

The control design of modular robots has been a longstanding problem that traces back to the first developments of this robotic concept [85,93,114]. The problem is that the overall system dynamics and kinematics are typically unknown after an arbitrary assembly. The motion-control performance of controllers designed and tuned before knowing the dynamics of the final assembly (e.g., with classical, simple approaches like Proportional-Integral-Derivative schemes) may be significantly poor and even lead to instabilities [127]. Most work in the control of modular robots proposed the use of advanced decentralized control schemes for controlling such systems e.g., [62,74,121]. The main motivation behind the development of these decentralized controllers is the intrinsic difficulty of obtaining the system dynamics for designing centralized model-based controllers. Unfortunately, forcing modular robots to require advanced decentralized or adaptive controllers for providing satisfactory control performance makes them significantly unattractive with respect to classical fixed-configuration robots. This is due to the fact that classical robots can rely on well-established, high-performance centralized model-based control methods instead (e.g., computed torque control [29]). In fact, while decentralized control approaches treat coupling effects as disturbances to be rejected, centralized model-based controllers can exploit

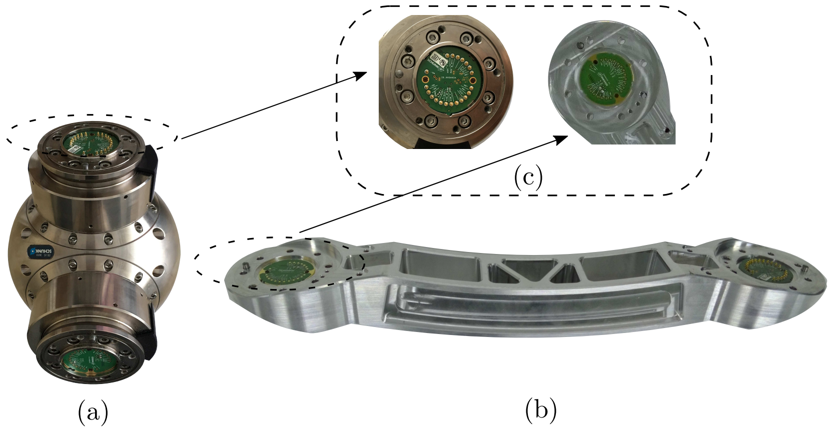


Figure 1.3: Examples of modules: (a) complex joint module, (b) link module, and (c) connectors.

the knowledge of the model for direct compensation, resulting in better performance, especially when direct drive actuators are employed and fast trajectories are required [96, Chapter 8]. In addition, centralized model-based architectures can also be used beneficially for impedance control [78], dynamic scaling of trajectories [49], and fault detection [36].

This thesis is focused on the development of centralized model-based control methods that allow the motion-control performance of modular, reconfigurable manipulators to approach those of their classical fixed-configuration counterparts. In the following, a more detailed introduction to the control problem for this class of robots is provided. The introduction continues with a summary of the contribution made by the author to the state of the art and ends with the outline of the thesis.

1.1 Introduction to the Control Problem

This thesis focuses on the control of modular reconfigurable arms that can be assembled from heterogeneous modules. This implies that no specific geometry is assumed and arbitrarily shaped modules may be involved. Throughout this thesis, a module is a rigid object that can be used as a building block for composing a robot arm. This is possible by means of standardized connectors that are part of each module. An example can be found in Figure 1.3c. Hereafter, the connectors considered allow the connection of consecutive modules at only one relative orientation. This is assumed for the sake of simplicity and does not affect the generality of the results presented as it becomes evident later. A fundamental distinction considered is between modules that introduce one or more degrees of freedom to the robotic structure once assembled and those that do not. The former are denoted as joint modules: simple joint modules if they have only one joint axis, and complex joint modules if they have more (see Figure 1.3a). The modules that do not introduce degrees of freedom to the robotic structure are denoted as link modules instead (see Figure 1.3b). Joint and link modules are considered to be rigid bodies. Possible passive compliance is considered, if applicable, to be part of the actuation unit e.g., when series elastic actuators [89] are employed.

In this thesis the motion control problem of modular reconfigurable robot arms is addressed, assuming that a set of heterogeneous modules of the type described above is available. This control problem is more difficult with respect to the case of classical fixed-configuration robots, since the robot kinematics and dynamics can frequently change due to reassembling. This thesis presents methods that allow the deployment of model-based controllers automatically, after the modular arm has been arbitrarily assembled as illustrated in Figure 1.4. The presented approaches account for the new kinematics and dynamics that each new assembly exhibits to guarantee stability and control performance. To this end, the problem of automatic (on-the-fly) generation of model-based control for modular robots with rigid components and transmissions is first addressed, assuming the dynamical parameters of the modules can be known precisely. Second, the assumption of having relatively precise knowledge of the dynamical parameters of the modules is relaxed, and the problem of guaranteeing robust performance when facing model uncertainties and external disturbances is presented. Then, the assumption of complete rigidity of the assembled robot arm is relaxed, and the problem of the inclusion compliant joints is tackled.

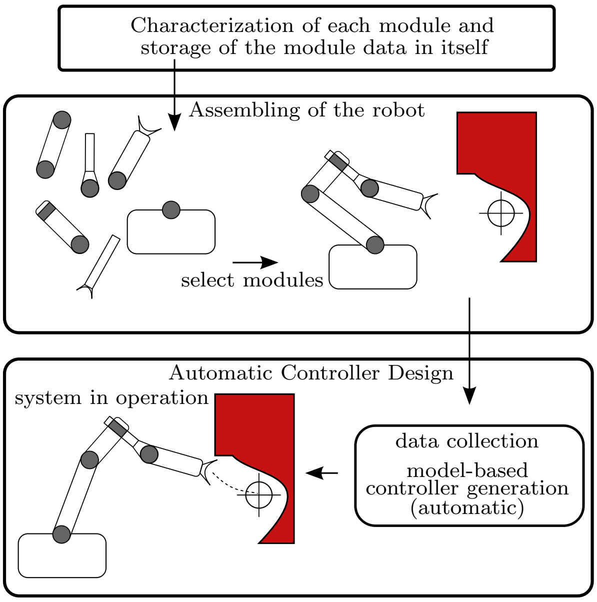

Characterization of each module and storage of the module data in itself
Assembling of the robot
select modules
Automatic Controller Design
system in operation
data collection
model-based controller generation (automatic)

Figure 1.4: Illustration of the automatic centralized controller design approach.

Tracking Control with Rigid Modules

With the generic modular robot setting described above, and considering the case that joint modules do not have significant elasticity, modules can be serially connected to realize an assembly of a rigid robot manipulator. Such assembly can be changed arbitrarily by adding additional modules or by swapping them for new ones with different shapes. Within the considered setting, the main motion-control challenge is that the resulting system kinematics and dynamics may be different for each different assembly of the arm. These models cannot typically be know a priori, especially when considering a large set of available modules and that new arbitrarily shaped modules could enter the set already available. The study of each different assembly, for manually obtaining the respective dynamic and kinematic model, is a nontrivial and time-consuming task which would limit the benefits from quick reconfigurability introduced by this class of robots. This thesis first faces the problem of automatically designing model-based control laws for each new rigid robot assembly, in order to ensure tracking of sufficiently smooth trajectories in joint- and task-space.

Tracking Control with Model Uncertainties

For the same modular-robotic setting described so far, the control problem that arises from input disturbance and uncertain dynamical parameters of the modules is also addressed. In practice, uncertain knowledge is typical for dynamical parameters like inertia tensors, coordinates of the centers of mass, and friction model coefficients. In particular, when the robot grasps unknown payloads, the dynamical parameters of the last body of the arm (end effector with grasped payload) can not be known a priori and may vary significantly from the nominal conditions. For the above-mentioned uncertainties, this thesis focuses on the problem of automatically designing controllers that provide both robust stability and robust performance. With the term *robust stability*, this thesis considers the same meaning and terminology for uniform, ultimate boundedness of the error trajectories used in [30]. A controller that guarantees this practically ensures that the error trajectories will converge to a ball centered at the origin in finite time and stay within that ball for future times, despite disturbances. Similarly, this thesis refers to *robust performance* for a controller that ensures uniform ultimate boundedness of the error trajectories when a user-defined radius of the ball is specified.

Tracking Control with Elastic-Joint Modules

The third fundamental control problem considered in this thesis arises from the possible presence of non-negligible elasticity when compliant actuators are employed. With respect to the rigid case, elastic joints provide enhanced resiliency to the robotic structure, safer interaction with the environment and humans, energy storage, or force control capabilities [109]. The most common compliant actuation units are composed of series elastic actuators [89]. In these cases, each elastic joint can be seen as a serial connection of the rotor, the transmission, and the elastic element (see Figure 1.5). For simplicity, the possible elasticity of the transmission can usually be assumed to be part of the elastic element. The elastic element is then connected to the rigid link of the robotic arm.

The complexity of the motion-control problem increases significantly with respect to the case in which only rigid components are involved. When considering a robot arm fully composed of elastic joints, one can grasp the (conceptual) increase in complexity by considering that the bodies involved in the motion with non-negligible dynamics double. Moreover, the control torque can be directly exerted on a subset of these bodies only (the rotors). In practice, to properly model such robots, the state variables have to be extended (doubled) to capture both motor-side and link-side dynamics, coupled through the elastic elements. The motor-side dynamics describe the motion of the ensemble rotor-transmission while the link-side dynamics describe the motion of the outer robot structure, which is equivalent to the rigid robot case.

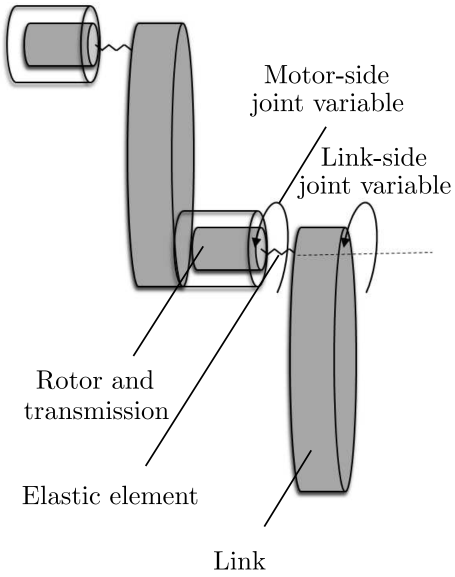


Figure 1.5: Schematic representation of a robot arm with series elastic actuators.

Within the modular reconfigurable robotic setting subject of this thesis, the presence of compliant actuators creates a new class of joint modules to be considered (elastic-joint modules). When the set of available modules has at least one elastic-joint module, a robot (fully or partially) composed of elastic joints can be assembled. As control designers typically consider the presence of elasticity (for maintaining precision and closed-loop stability), so too should any procedure that automatically synthesizes the control after assembly.

When considering the generic scenario of modular reconfigurable robot arms, it is important to stress that the enumeration of all possible link-side models for arbitrary assemblies becomes difficult. This aspect represents a fundamental limitation in the control design process, as can be observed with rigid modules. In summary, the last fundamental problem addressed in this thesis is to provide controllers that can be automatically synthesized after the assembly of modules with possibly compliant actuators. In particular, these control schemes should be able to globally track sufficiently smooth, link-side joint-space trajectories.

1.2 Author's Contribution

New approaches for providing centralized model-based control of modular robot manipulators have been proposed by the author. These methods provide: 1) a systemic way for the *automatic* design of centralized, model-based controllers after reassembling, 2) automatically deployable robust performance control despite the presence of model uncertainties and external disturbances, and 3) a robust scheme for global tracking control in the presence of elastic-joint modules.

Preliminary research results led to a novel method for automatically and systematically designing centralized model-based control laws of modular robot manipulators. This approach was first published in [124], then successively extended and applied on a real robot test bed [127]. With this approach, as demonstrated later in this thesis, modular robots can benefit from the same model-based control schemes available for their fixed structure counterparts, through automatic modeling and controller design. The automatic modeling approach developed can also be exploited by optimization procedures for task-based assembly selection as proposed in the additional contribution [132].

As for all model-based control approaches, it is important to consider that model uncertainties are crucial for motion-control performance. A novel interval-arithmetic-based robust control method has been developed and was first published in [125]. This approach has been complemented with the work in [126] to remove its main drawback of a potentially high computational cost. For the first time, these contributions enable the on-the-fly commissioning of *robust* model-based controllers for modular reconfigurable robots. Another implication of the work in [126] is that it enabled the development of robust trajectory scaling and collision detection strategies as shown in [133]. Interestingly, the proposed approach additionally paved the way for the development of an effective robust control method for continuum robots, which has been published in [131].

The consideration of possible elasticity in the joints that may be introduced by modern compliant actuators led to the development of a new control scheme that has unique benefits with respect to the state-of-the-art approaches. This approach is proposed in [128] and allows merging the robustness typical of passivity-based control schemes with the efficient numerical computability of inverse-dynamics schemes. A detailed analysis of robustness and additional experimental results were collected in [129]. Finally, a new framework for enhancing flexibility in automation, as a result of a combination of the framework of programming by demonstration and modular robots, has been developed and successfully applied to realistic industrial scenarios [130].

Throughout this thesis, the phrases “the author’s contribution” and “the author’s work” refer to works that have been authored or co-authored by the author of this thesis.

1.3 Outline of the Thesis

This thesis is built around the three fundamental control problems previously mentioned. Three chapters have been provided to describe solutions to these problems. Each of these chapters starts with an introductory section, which also discusses the state-of-the-art works related to the respective problems. The core part of Chapter 2 provides a framework for automatic synthesis of the centralized control that accounts for the kinematics and dynamics of each new robot assembly. The basic idea is that the task of modeling the entire arm is translated into the task of modeling each module instead. This has the benefit that the parameters of the modules are not assembly dependent and can be obtained only once. For this purpose, a systematic procedure and a novel notation both for kinematics and dynamics are detailed in Chapter 2. The application of this approach on a real modular robot is described, along with experimental results that verify its effectiveness.

Starting from the results in Chapter 2, a possible extension is provided in Chapter 3 which relaxes the assumption of perfect knowledge of modules’ parameters. Here, a new control method for robust control of robot arms is described. Most robust controllers require empirical estimation of model perturbations, undermining the quick deployability which is necessary for modular robots. The new proposed approach solves this problem using interval arithmetic for automatic online computation of worst-case perturbations due to model uncertainties. Interestingly, this method enables the automatic deployment of the robust control with superior performance compared to earlier solutions. For this approach, the relevant preliminaries, the theoretical derivation, a simulation comparison, and experimental results are presented in Chapter 3.

Chapter 4 describes possible solutions to the automatic synthesis of controllers in the presence of joint modules with non-negligible elasticity. This is possible by using some existing approaches for the control of elastic-joint robots in conjunction with the framework described in Chapter 2. A novel approach which provides enhanced robustness is presented in detail along with the analysis of its robustness with respect to model uncertainties. Simulation results show the performances, and the experimental results its real world applicability. Each of the above mentioned chapters is concluded with a summary section which wraps up the main outcomes. The conclusion of the thesis is provided in Chapter 5 and includes possible future research directions.

Chapter 2

A Framework for Automatic Centralized Controller Design

After outlining the main works which constitute the state of the art of modular-robot control, this chapter focuses on a modern approach that aims at removing—at the root—the main limitations related to the motion-control of this class of robots. With the core content largely based on the results in [124, 127], this chapter presents a framework for automatically designing model-based controllers of modular manipulators, ranging from module modeling to tracking task-space trajectories. This chapter includes a description of an experimental application to a commercially available modular robot manipulator and related experimental results.

2.1 Introduction and State of the Art

The control problem of modular robot manipulators has attracted significant attention in recent decades. As it is the case for classical fixed-configuration robots, two main trends can also be observed for controlling modular robots: decentralized and centralized approaches.

2.1.1 Decentralized Control

Most published works for controlling modular robots have focused on decentralized schemes. The main motivation of all these approaches is the impracticability of obtaining a dynamical model for all possible assemblies of modular robots for designing a corresponding model-based controller. This problem becomes more critical the larger the set of available modules, due to the increase of the number of possible combinations of modules. The probably simplest decentralized controllers that can be employed are the same that have become the most employed in industry for classical fixed-configuration robots. These approaches are typically based on the Proportional, Integral, and Derivative (PID) action and are designed for each joint by considering the dynamics of the actuator only and by assuming the couplings due to the movements of the whole arm as disturbances to be rejected. A detailed description of such classical approaches can be found e.g., in [96, Section 8.3]. These simple schemes may provide acceptable control performance when using sufficiently light-weight modules, in which the resulting coupling effects can be considered as negligible. However, while for classical robots the structure does not change and fine-tuning can be done only once, there is no guarantee that no instabilities will appear during or after re-tuning when using arbitrary modules and changing the robot assembly. In practice, the use of simple, model-free decentralized controllers that do not guarantee global asymptotic stability (e.g., PID) may lead to time consuming tuning phases that undermine the benefits of the fast reconfiguration capabilities of modular reconfigurable robots. Approaches that enhance the simple PID structure with nonlinear terms for obtaining global stability have also been proposed e.g., in [56, 67]. However, the tuning in these cases in nontrivial as well, since the knowledge of bounds on norms of model terms is necessary. Obtaining these bounds is difficult for modular reconfigurable manipulators whose structure can arbitrarily and frequently change.

Several decentralized control schemes specifically designed for modular robots have been developed in recent decades. The authors in [74] introduce a scheme based on fuzzy gain tuning of distributed PID controllers. This scheme addresses the regulation control problem of modular robots and interestingly requires no knowledge of the dynamic parameters of the manipulator. The authors in [121] consider the dynamics of the assembled robot arm as those of a set of interconnected subsystems and design an adaptive control action to cancel the couplings. That work is strongly related to [106], in which Tang *et. al.* propose a decentralized robust control method for mechanical systems with its application to robot tracking control. Liu *et. al.* in [65] propose a decentralized control method based on joint torque sensing for compensating model uncertainties resulting from reconfiguration, which relies on a decomposition-based robust control scheme [64] for compensating remaining uncertainties, such as friction. Li *et. al.* in [62] propose a decentralized and robust control method which combines a linear PD action and a nonlinear term to provide a controller which has a simple resulting structure and requires low tuning effort after a reconfiguration. A control scheme based on the use of both the Virtual Decomposition Control (VDC) framework [122] and embedded FPGA devices has been proposed in [123]. This approach does not require torque sensing and distributes the computations of the robot dynamics to the FPGA devices of each actuator. Even though the authors report high tracking performance in experiments, this method presents high implementation complexity compared to the previously mentioned decentralized schemes and requires the kinematics to be computed in a centralized fashion.

2.1.2 Automatic Modeling and Centralized Control

While decentralized control approaches treat coupling effects as disturbance to be rejected, centralized model-based controllers can exploit the knowledge of the model for direct compensation. This results in better performance, especially when direct drive actuators are employed and/or fast trajectories are required [96, Chapter 8].

The author's work in [124, 127] proposes an alternative to the current research trend of developing decentralized control schemes by fostering the automatic generation of centralized controllers with a systematic approach, using data that are stored in the modules or in a database. These data are used to automatically generate the complete model of the arm and centralized model-based controllers, once gathered at a central control unit after each assembly. Automating the generation of the centralized model-based control for modular robots has the remarkable advantage of solving the control-related difficulties at their origin.

For realization of centralized model-based controllers for modular robots, the automatic derivation of the model of the arm for each new assembly is crucial. It is important to note that one of the seminal works in modular robotics mentions the use of data stored in the modules for automatically obtaining the gravity vector in [85], without details on the approach followed for automatically obtaining the dynamics of the assembled arm. Other authors also focused on the derivation of complete models from modules such as in [26, 27, 73] based on [86] and the Product of Exponentials (PoE) formulation. However, these methods have not shown seamless applicability to arbitrarily shaped modules and are more suitable for modular structures with specific geometries [16]. On the other hand, the author's work in [124, 126] is based on an extension of the standard Denavit-Hartenberg (D-H) convention [41] and has been specifically developed to be systematically applicable to arbitrary modules. Since the D-H convention is among the most widely employed methods for kinematic modeling of standard manipulators, a collateral benefit of extending it instead of relying on the PoE formalism is that most researchers and practitioners may find it more familiar. Please note that even though beneficial features have been reported when using the PoE formulation when performing kinematic calibration, the recent development of an automatic approach for going from the D-H to the PoE formulation

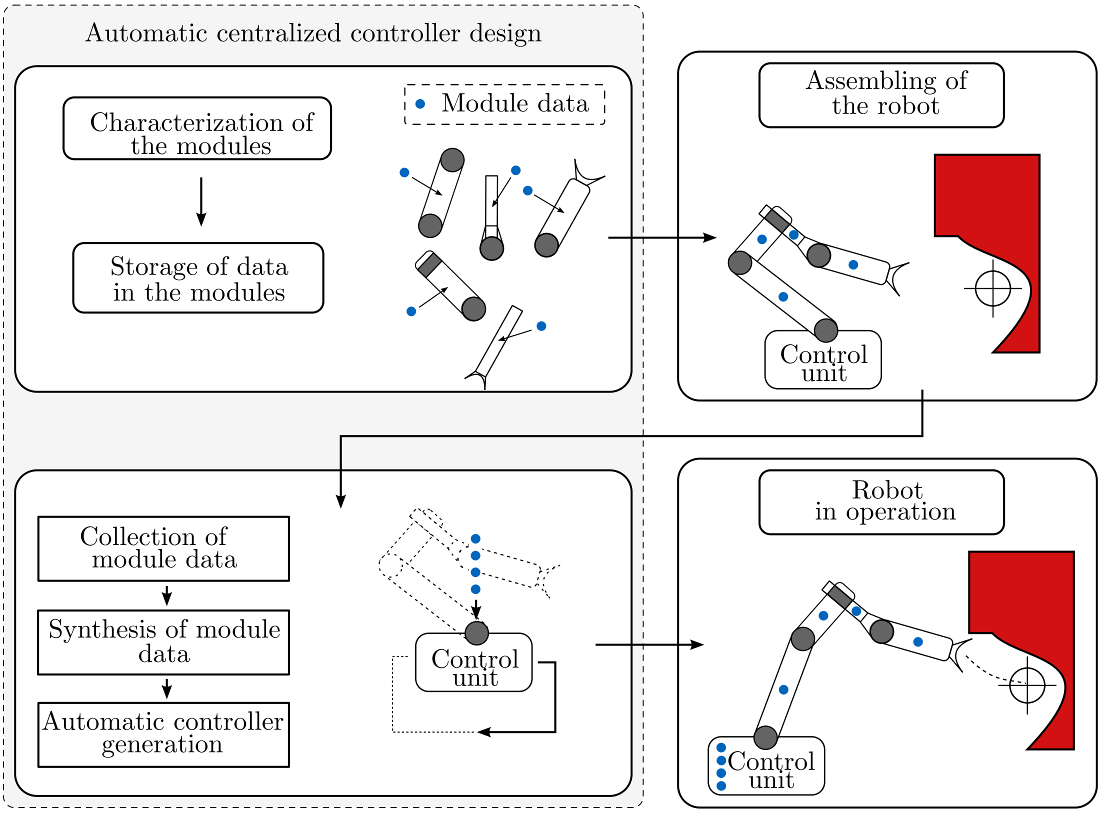

Automatic centralized controller design
Characterization of the modules
Module data
Storage of data in the modules
Assembling of the robot
Control unit
Collection of module data
Synthesis of module data
Automatic controller generation
Control unit
Robot in operation
Control unit

Figure 2.1: Illustration of the control approach application.

and vice versa in [113] removes the exclusivity of the benefits that have been typically associated so far with the PoE formulation only.

Being based on an extension to the standard D-H convention, the author's work in [124,126] is related to the previous works [14,16] and [57]. While [57] deals with revolute joints only, in [14,16] the problem of the D-H convention's non-uniqueness for some special cases of consecutive joint axes [96, Section 2.8.2] has not been considered. This is an important aspect because e.g., having parallel joint axes is not uncommon and falls into the category of these special cases. The previously mentioned extension to the standard D-H convention has been proposed in the author's work [124] as a solution for these shortcomings. As illustrated in Figure 2.1, the basic idea of the author's method starts with the characterization of each module to obtain a set of data. These data consist of a unique identification number and specific kinematic/dynamic parameters of the modules that will be described in detail in the next sections. The data are derived only once at the time of the module development or calibration. They are assembly independent and will be simply called *module data* in this thesis. After the robot is assembled, the central control unit collects the module data into an array of structures and processes them for automatic generation of the assembled robot description and centralized model-based control laws that let the robot operate with global asymptotic stability. After assembly, the data are gathered by an appropriate communication network which is also used to control the robot.

In the next Sections 2.3 and 2.4, the author's systematic method for modeling modular robots is presented. For both kinematics and dynamics, the approach starts by presenting a way to characterize arbitrary modules. In both cases, the details on how the data are processed recursively to obtain the respective kinematic and dynamical model of the assembled arm follow. The details of the implementation of centralized joint-space control schemes and closed loop inverse kinematic algorithms for solving numerically online the inverse kinematic problem are presented in Section 2.5.

2.1.3 Network Solutions

A suitable network topology for practical implementation of the approach in [124, 127] is a tree structure, since it supports serial and branch-structured manipulators and is composed of a coordinator (central control unit), routers (intermediate modules) and end-devices (end effectors). In principle, with this structure, the coordinator can get measurements (e.g., joint position and velocity) from the modules, set input commands to the actuators, and set/get data in/from the memory database of the modules (e.g., the module data). Additionally, knowing the routing tables enables in principle the central control unit to detect the robot configuration. SpaceWire[1] is an example of a suitable communication network for controlling modular robot manipulators. Its implementation for controlling a system of this class with high speed requirements and also with inter-module communication capabilities is described in [61]. An alternative is EtherCAT[2] which has been used in [9], where the actuation units have strong hardware modularity. When considering EtherCAT for controlling modular robots where modules can be arbitrarily swapped and replaced, the generation of the network description for each new assembly is required. However, this does not require significant time delays or particular low-level control knowledge. A simple communication bus (e.g., CAN bus[3]) could also be used. In this case, if the sequence of the modules that compose a new assembly is

[1] http://spacewire.esa.int/content/Home/HomeIntro.php

[2] http://www.ethercat.org/

[3] http://www.can-cia.org/

not given by the user, additional communication lines should be made available as described in [115] for self-detection of the configuration of the modules.

2.2 Formulation of the Control Problem

Once assembled, each robot arm has N degrees of freedom, where N is the total number of joint axes introduced by the joint modules. After assembly, the robot arm does not differ from a classical fixed-configuration robot. Therefore, using bold symbols for vectors and matrices, the dynamical model of such a system can be written as follows [96, Chapter 7]:

$$\mathbf{M}\left(\mathbf{q}\right)\ddot{\mathbf{q}} + \mathbf{C}\left(\mathbf{q},\dot{\mathbf{q}}\right)\dot{\mathbf{q}} + \mathbf{f}\left(\dot{\mathbf{q}}\right) + \mathbf{g}\left(\mathbf{q}\right) = \mathbf{u}, \tag{2.1}$$

where $\mathbf{q} \in \mathbb{R}^N$ is the vector of the joint positions, $\mathbf{M}\left(\mathbf{q}\right) \in \mathbb{R}^{N\times N}$ is the symmetric and positive definite inertia matrix, $\mathbf{C}\left(\mathbf{q},\dot{\mathbf{q}}\right)\dot{\mathbf{q}}$ (with $\mathbf{C}\left(\mathbf{q},\dot{\mathbf{q}}\right) \in \mathbb{R}^{N\times N}$) is the vector of the Coriolis and centrifugal terms, $\mathbf{f}\left(\dot{\mathbf{q}}\right) \in \mathbb{R}^N$ is the vector of friction terms, $\mathbf{g}\left(\mathbf{q}\right) \in \mathbb{R}^N$ is the vector of gravity terms, and $\mathbf{u} \in \mathbb{R}^N$ is the vector of the actuation forces/torques. It can be shown that with a suitable factorization of $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})$, the matrix $\mathbf{N}(\mathbf{q},\dot{\mathbf{q}}) = \dot{\mathbf{M}}(\mathbf{q}) - 2\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})$ is skew-symmetric, and therefore

$$\mathbf{x}^T\mathbf{N}(\mathbf{q},\dot{\mathbf{q}})\mathbf{x} = 0, \quad \forall\mathbf{x} \in \mathbb{R}^N, \tag{2.2}$$

which can be beneficially used for control design [18, Section 2.1].

Within the considered setting, the main challenge to the design is that (2.1) may be different for each different assembly of the arm. These models cannot typically be know a priori, especially when considering a large set of available modules and that new, arbitrarily shaped modules could enter the set of modules available. This thesis first faces the problem of automatically designing model-based control laws for each new assembly, to ensure tracking with global asymptotic stability trajectories in joint-space $\mathbf{q}_d(t)$, which are at least twice differentiable:

$$\lim_{t\to\infty} \|\underbrace{\mathbf{q}_d(t) - \mathbf{q}(t)}_{:=\,\mathbf{e}(t)}\| = 0,$$

where $\|\mathbf{e}\left(t\right)\|$ is the Euclidean norm of the joint-space tracking error vector. In addition, the problem of computing the desired joint-space trajectories from task-space ones through inverse kinematics is also considered, targeting

$$\lim_{t\to\infty} \|\mathbf{e}_p\left(t\right)\| = 0, \quad \lim_{t\to\infty} \|\mathbf{e}_o\left(t\right)\| = 0,$$

where $\mathbf{e}_p \in \mathbb{R}^3$ and $\mathbf{e}_o \in \mathbb{R}^3$ are the tracking error vectors for position and orientation of the end effector, respectively. For solving this problem, it is assumed that the dynamical parameters of modules can be obtained with high accuracy (e.g., by using modern CAD software or direct measurements).

2.3 Kinematic Modeling using Module Data

Automatic modeling of the kinematics can be achieved by relying on a proper characterization of each module and exploiting an extension of the standard D-H convention [41]. Before presenting the proposed extension, the standard convention is briefly recalled for the sake of clarity.

Standard D-H Convention

The forward kinematics for serial robot manipulators is commonly derived by multiplying the homogeneous transformation matrices relating consecutive link-fixed reference frames, from the base to the end-effector. The D-H convention provides a systematic method for the assignment of the link-fixed frames. According to the standard D-H convention (see Figure 2.2a), for the ith link, z_i is placed along the axis of the joint at the connection with link$_{i+1}$. The x_i axis lies along the common normal between the axis z_{i-1} and z_i pointing toward link$_{i+1}$. The origin o_i of the frame fixed to link$_i$ is set at the intersection of the common normal with z_i. Finally, the y_i axis completes the right-handed coordinate system (which is not shown in Figure 2.2). To define the relative transformation of coordinates of subsequent frames, four parameters are introduced: a_i, d_i, α_i, θ_i. With reference to Figure 2.2a, a_i is the distance between $o_{i'}$ and o_i along the common normal between the axis z_{i-1} and z_i, d_i is the distance between o_{i-1} and $o_{i'}$ along z_{i-1}, θ_i is the angle between x_{i-1} and $x_{i'}$ about z_{i-1}, and, finally, α_i is the angle between z_i and z_{i-1} about x_i. Among these parameters, d_i is variable when the joint is prismatic, and θ_i when the joint is revolute. By adopting this convention and using S_χ/C_χ as abbreviations of $sin(\chi)/cos(\chi)$, the homogeneous transformation matrix relating the frame of link$_i$ to the one of link$_{i-1}$ is [96, Section 2.8.2]

$$\mathbf{A}_i^{i-1} = \begin{pmatrix} \mathbf{R}_i^{i-1} & \mathbf{U}_i^{i-1} \\ \mathbf{0}^T & 1 \end{pmatrix} = \begin{pmatrix} C_{\theta_i} & -S_{\theta_i}C_{\alpha_i} & S_{\theta_i}S_{\alpha_i} & a_iC_{\theta_i} \\ S_{\theta_i} & C_{\theta_i}C_{\alpha_i} & -C_{\theta_i}S_{\alpha_i} & a_iS_{\theta_i} \\ 0 & S_{\alpha_i} & C_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{2.3}$$

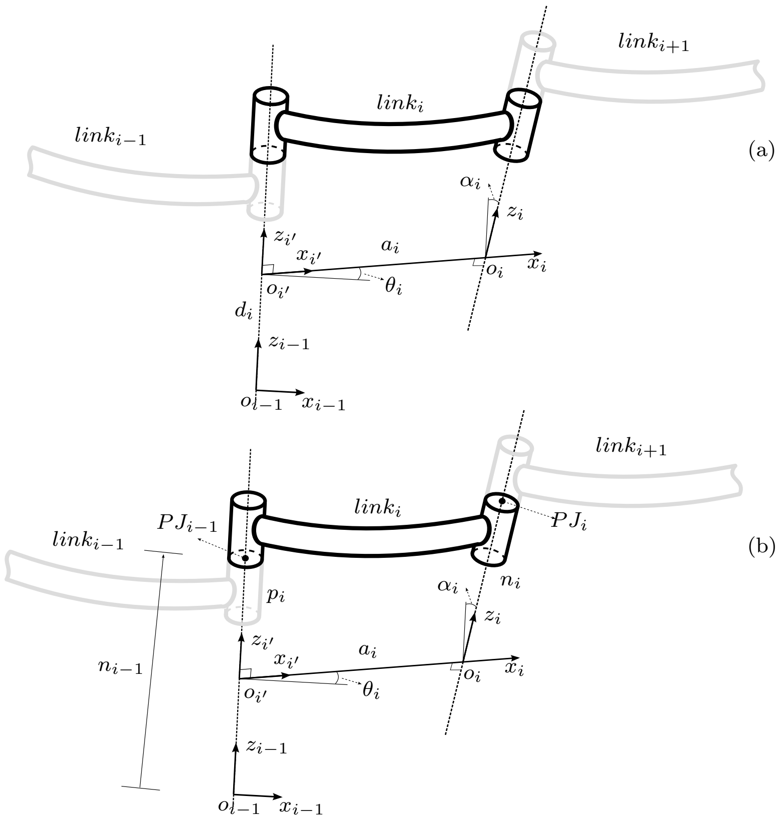


Figure 2.2: Representation of a link of a manipulator, showing the parameters for kinematics using (a) the standard D-H convention and (b) its extension introduced in [124].

2.3.1 An Extended D-H Convention

To address the automatic nature of the presented approach, it is important to resolve the problem that the standard D-H convention is not unique. In fact, for some relative orientations of subsequent joint axes (i.e., parallel, intersect, or overlapped) the modeler has partial freedom to place the link-fixed frames. Even though this freedom is usually exploited to simplify the modeling procedure, it represents a problem for an automatic approach that requires a deterministic rule for assigning the frames. The frames can also be set deterministically in these cases by extending the standard D-H convention. In particular, with reference to Figure 2.2b:

- when the z axes intersect, the x_i unit vector is obtained from their cross product;
- when the z axes are parallel, the x_i unit vector is set along the common normal between them and the origin o_i is set at the joint connection PJ_i;
- when the z axes are superimposed, the x_i unit vector is aligned with x_{i-1} and the origin

o_i is set at the joint connection PJ_i.

In addition to the standard D-H parameters for each link, as shown in Figure 2.2b, two additional parameters are present (p_i and n_i) that are needed to completely describe the transformations of coordinates from one joint to the successive one. These parameters are also required for automated derivation of the set of D-H parameters for each link as it is clarified later. The parameter p_i is the z coordinate of the point PJ_{i-1} measured from $o_{i'}$, and the parameter n_i is the z coordinate of the point PJ_i measured from o_i. Considering the joint variable q_i, noticing in Figure 2.2b that the following relations hold is straightforward:

$$d_i = \begin{cases} n_{i-1} - p_i, & \text{(revolute joint)}, \\ n_{i-1} - p_i + q_i, & \text{(prismatic joint)}. \end{cases} \tag{2.4}$$

Furthermore, to consider a possible angular offset between consecutive x axes when the joint is in zero position ($q_i = 0$), a parameter γ_i is introduced and simply included as follows:

$$\theta_i = \begin{cases} \gamma_i + q_i, & \text{(revolute joint)}, \\ \gamma_i, & \text{(prismatic joint)}. \end{cases} \tag{2.5}$$

It is not difficult to notice that given a_i, α_i, γ_i, p_i, n_i, and the type of the joint actuation for each link of the manipulator, the parameters d_i and θ_i of the standard D-H convention can be easily computed by using (2.4) and (2.5), while a_i and α_i simply remain the same.

The above-described extension of the D-H convention is applicable to any serial manipulator and can be used to model the kinematics in a systematic and unique way. The automatic modeling approach described in this section uses this extension in two cases. One is after the robot is assembled and the module data for kinematics are collected at the central control unit. In fact, in this case the set of parameters of the extended D-H convention for the assembled manipulator are automatically obtained, from which the derivation of the standard D-H parameters follows directly as described above. The second case when this extended convention is employed is to define a notation for characterizing heterogeneous modules and thus provide a systematic way for deriving the module data for kinematics of arbitrary modules.

2.3.2 Definition of the Module Data for Kinematics

In general, the kinematic characterization of the modules allows one to parametrize the transformation of coordinates from a frame placed at the input connector of a module to a frame placed at the output one. This is described for joint modules first, and then for link modules. Considering that every link of an assembled modular manipulator (arm-link) is composed of modules, a procedure for the synthesis of the module data for kinematics to obtain the parameters of the extended D-H convention (a_i, α_i, γ_i, p_i, n_i) is also necessary, and is later presented in Subsection 2.3.3.

The procedure for characterizing a joint module can be described by considering the exemplary joint module represented in Figure 2.3a. The first step is to fix a frame of reference for the proximal part and the distal part, located in the center of the interfaces of the input and output connector, respectively. The axes x_{in} and x_{out} are aligned with a unique and standardized direction on the connection interface such that, when subsequent modules are connected, the x axis of the output frame of one module is superimposed onto the x axis of the input frame of the consecutive one. The z axes are normal to the respective connector interfaces: z_{in} points in towards the input connector, and z_{out} points outwards from the output one (see Figures 2.3b and 2.3c). The y axes are selected to complete the respective right-handed frames of reference (not shown in the figure). To characterize both the proximal and the distal parts with a set of parameters, a similar approach of the extended D-H convention described previously is employed, and four parameters for the proximal part (a^{pl}, α^{pl}, p^{pl}, n^{pl}) and for the distal part (a^{dl}, α^{dl}, p^{dl} and n^{dl}) are obtained (see Figures 2.3b and 2.3c). These parameters depend on the geometry of the module and can be inferred considering two auxiliary frames for both the proximal and the distal parts. For the proximal part, with reference to Figure 2.3c, the first auxiliary frame has its origin at o'_{pl}, the intersection of the common normal between z_{in} and the joint axis with z_{in}. The axis x'_{pl} is set along the common normal pointing towards the distal part, and z'_{pl} is set along z_{in}. The y axes are not shown because they simply complete the respective right-handed reference frames. The second auxiliary frame of the proximal part has its origin in o''_{pl}, at the intersection of the common normal with the joint axis; its axis x''_{pl} is set along the common normal and points towards the distal part, and z''_{pl} is set along the joint axis. The four parameters for the proximal part have the following meanings:

- a^{pl} is the distance between o'_{pl} and o''_{pl} along the common normal;
- α^{pl} is the angle between the axis z_{in} and the joint axis around x''_{pl};
- p^{pl} is the z coordinate of the input connection point o_{in} from o'_{pl};
- n^{pl} is the z coordinate of the joint connection point PJ from o''_{pl}.

For the distal part illustrated in Figure 2.3b, two auxiliary frames are analogously introduced and the four parameters for the distal part can be similarly obtained. These parameters indeed have similar meanings:

- a^{dl} is the distance between o'_{dl} and o''_{dl}, along the common normal;

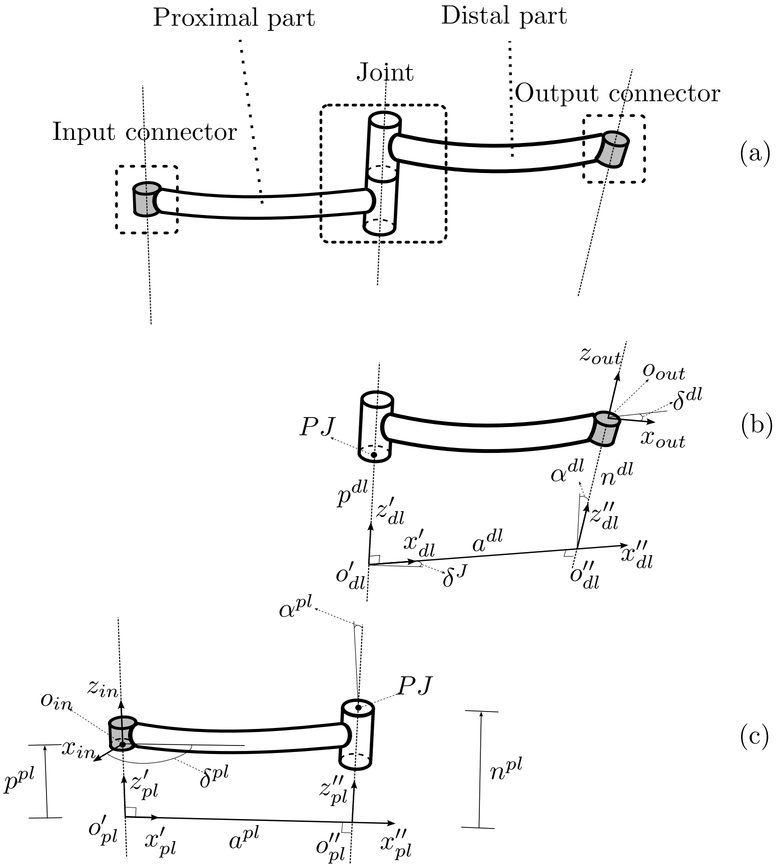


Figure 2.3: Kinematic notation for characterizing a simple joint module. The connectors are indicated in light-grey color, (a) is the entire module, (b) the distal part, and (c) the proximal part.

- α^{dl} is the angle between the joint axis and z_{out} around x''_{dl};
- p^{dl} is the z coordinate of the joint connection point PJ from o'_{dl};
- n^{dl} is the z coordinate of the output connection point o_{out} from o''_{dl}.

The complete parametrization of the transformation of coordinates from the input frame to the output one of a joint module requires three additional parameters: δ^{pl}, δ^{dl}, δ^J. With reference to Figures 2.3c and 2.3b: δ^{pl} is the angle between x'_{pl} and x_{in}, δ^{dl} is the angle between x''_{dl} and x_{out}, and δ^J is the angle between x''_{pl} and x'_{dl} when the joint is in its zero position. Please note that all the angles of this notation are positive counterclockwise.

The same systematic approach used for joint modules can be employed for characterizing link modules with a set of parameters. With reference to the exemplary link module of Figure 2.4, the input and output frames are first placed at the respective connectors with the

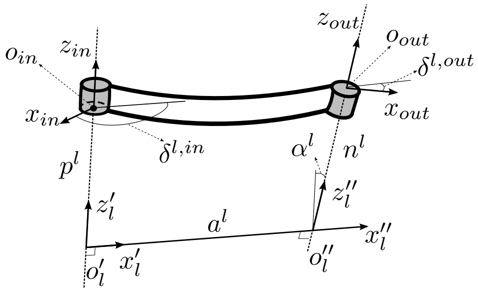


Figure 2.4: Kinematic notation for characterizing a link module. The connectors are indicated in light-grey color.

same procedure followed for joint modules. Once these frames are fixed, two auxiliary frames should be considered analogously to the procedure for the distal (or proximal part) of joint modules. The first one has its origin at o'_l, the intersection of the common normal between z_{in} and z_{out} with z_{in}. The axis x'_l is set along the common normal pointing towards the output connector, z'_l is set along z_{in}, and y'_l completes the right-handed frame of reference. The second auxiliary frame has its origin in o''_l at the intersection of the common normal with z_{out}. Its axis x''_l is set along the common normal and points in the same direction of x'_l, z''_l is set along z_{out}, and finally, y''_l completes the right-handed frame of reference. As for the proximal or distal parts of joint modules, the parameters of a link module can now be obtained from the module's geometry:

- a^l is the distance between o'_l and o''_l along the common normal;
- α^l is the angle between the axis z_{in} and z_{out} about x''_l;
- p^l is the z coordinate of the input connection point o_{in} from o'_l;
- n^l is the z coordinate of the output connection point o_{out} from o''_l.

Finally, the parametrization of the transformation of coordinates from the input frame to the output one is completed with the angle $\delta^{l,in}$ between x'_l and x_{in}, and the angle $\delta^{l,out}$ between x''_l and x_{out}. Please note that end-effector modules can be considered as link modules once the output frame is fixed on the end effector body in a pose convenient for the user.

For characterization of both joint and link modules, particular cases of the relative orientation between subsequent z axes (e.g., parallel, intersect, or overlap) are handled similarly to the presented extension of the standard D-H convention. The only difference is that in the previously described convention one refers to the relative orientation of two subsequent joint axes (since it applies to manipulators), while for characterizing the modules the axes considered are the two subsequent z axes of the auxiliary frames. For example, when these two axes intersect,

Joint Module	Proximal	a^{pl}	α^{pl}	p^{pl}	n^{pl}	δ^{pl}	
	Distal	a^{dl}	α^{dl}	p^{dl}	n^{dl}	δ^{dl}	
	Joint	δ^{J}		Joint type			
Link Module		a^{l}	α^{l}	p^{l}	n^{l}	$\delta^{l,in}$	$\delta^{l,out}$

Table 2.1: Module data for kinematic modeling.

the auxiliary frames have the same origin. Consequently, the parameter $a_{pl}/a_{dl}/a_{l}$ is zero, and the x axes of the auxiliary frames are parallel and oriented as the cross product between the considered z axes.

Table 2.1 collects the parameters to characterize a joint or a link module for kinematics. These parameters constitute the part of the module data for kinematics that are required to be stored in each module or in a database. Once the robot is assembled and these data are collected, the kinematic model of the robot can be automatically obtained by means of the procedure described next. Please note that this approach can be straightforwardly applied to modules with multiple input and/or output connectors by obtaining a set of parameters for each possible combination of such connectors.

2.3.3 From the Module Data to the Kinematic Model

It is now possible to introduce the procedure for automatically obtaining the extended set of D-H parameters (a_i, α_i, γ_i, p_i, n_i) for every link of an assembled manipulator (hereafter denoted as *arm-link* for brevity), using the data collected from the modules that compose the arm-link itself. We stress that the standard D-H parameters (and therefore the forward kinematics) can be easily obtained once the extended set of parameters and the joint types are known for each link of the manipulator.

As it can be observed from Figure 2.2b, the parameters can be obtained by considering each pair of joint axes from the base to the end of the robot arm. The automatic procedure for obtaining these parameters from module data follows the same approach. Figure 2.5 shows a generic link of an assembled modular manipulator and is useful for describing the procedure in detail. This exemplary case shows the connection of two joint modules through a non-negative number $k \in \mathbb{N}$ of link modules. To automatically obtain the parameters of the arm-link, we first compute a homogeneous transformation matrix $\mathbf{F}_i$, which describes the pose of a frame parallel to the D-H one and located at PJ_i from a frame parallel to the first auxiliary frame of

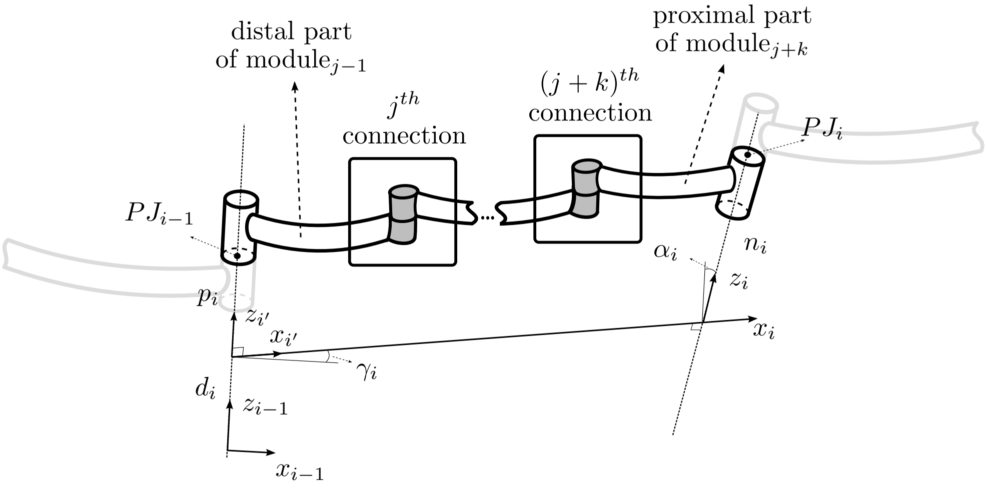


Figure 2.5: Synthesis of the extended D-H parameters for the *ith* constituted link with j being the number of modules connected to the arm starting from the basis.

the distal part of module$_{j-1}$ with origin PJ_{i-1} (see Figure 2.5). In the following, this matrix is referred to as a *synthesis matrix*. The synthesis matrix plays a crucial role in this procedure since it can be computed using module data, and its entries can be analytically related to the parameters of the extended D-H convention as shown next.

The synthesis matrix can be obtained in a two steps approach. First, an auxiliary homogeneous transformation matrix $\mathbf{F}'_i$ is computed. This matrix describes the pose of a frame parallel to the second auxiliary frame of the proximal part of module$_{j+k}$, with respect to the frame parallel to the first auxiliary frame of the distal part of module$_{j-1}$ and located at PJ_{i-1}. Second, an additional rotation ϕ_i about z is applied to align the frame with the one of the extended D-H convention.

The computation of $\mathbf{F}'_i$ is implemented using only the module data for kinematics of the modules that compose the arm-link under study. This is performed by means of homogeneous transformation matrices of elementary translations/rotations. In the following, the homogeneous transformation matrix of the elementary translation/rotation along/around the χ axis are compactly denoted by $T_\chi(\,\cdot\,)/R_\chi(\,\cdot\,)$. With reference to Figure 2.5, the homogeneous transformation matrices of the distal part $\mathbf{A}^{dl}_{j-1}$, of the k link modules $\mathbf{A}^{l}_{j,j+k}$, and of the proximal part of the module that completes the arm-link $\mathbf{A}^{pl}_{j+k}$, are obtained. These matrices can be directly used for computing $\mathbf{F}'_i$ as

$$\mathbf{F}'_i = \mathbf{A}^{dl}_{j-1}\,\mathbf{A}^{l}_{j,j+k}\,\mathbf{A}^{pl}_{j+k} = \begin{pmatrix} \mathbf{R}'_i & \mathbf{U}'_i \\ \mathbf{0}^T & 1 \end{pmatrix}, \tag{2.6}$$

where

$$\begin{aligned}\mathbf{A}^{dl}_{j-1} &= T_z(-p^{dl}_{j-1})T_x(a^{dl}_{j-1})R_x(\alpha^{dl}_{j-1})T_z(n^{dl}_{j-1})R_z(\delta^{dl}_{j-1}),\\ \mathbf{A}^{pl}_{j+k} &= R_z(-\delta^{pl}_{j+k})T_z(-p^{pl}_{j+k})T_x(a^{pl}_{j+k})R_x(\alpha^{pl}_{j+k})T_z(n^{pl}_{j+k}),\end{aligned}$$

and

$$\mathbf{A}^{l}_{j,j+k} = \begin{cases}\prod_{h=j}^{j+k-1} R_z(-\delta^{l,in}_h)T_z(-p^l_h)T_x(a^l_h)R_x(\alpha^l_h)T_z(n^l_h)R_z(\delta^{l,out}_h), & \text{if } k > 0,\\ \mathbf{I}_{4\times 4}, & \text{otherwise.}\end{cases}$$

The transformations of $\mathbf{F}'_i$ allow the pose of a frame parallel to the second auxiliary frame of the last joint module in the arm-link to be reached, but not the one of the D-H frame. As previously mentioned, to obtain $\mathbf{F}_i$ an additional rotation ϕ_i about z_i has to be considered. This angle is computed considering the previously discussed particular cases for the subsequent joint axes as follows:

i. When the consecutive joint axes overlap: $\phi_i = 0$.

ii. When the axes are parallel[1]:

$$\phi_i = \operatorname{atan2}(v_y,\, v_x), \text{ from } \mathbf{V} = [v_x,\, v_y,\, v_z]^T = \mathbf{R}'^T_i\,\mathbf{U}'_i. \tag{2.7}$$

iii. When the axes are skew or intersect:

$$\phi_i = \operatorname{atan2}(v_y,\, v_x), \text{ from } \mathbf{V} = [v_x,\, v_y,\, v_z]^T = \mathbf{R}'^T_i\,\mathbf{V}' \text{ and } \mathbf{V}' = \mathbf{z}_i \times \mathbf{z}_{i-1}. \tag{2.8}$$

The detection of these cases is performed using $\mathbf{U}'_i$ and the unit vector of the z axis in $\mathbf{R}'_i$ from (2.6), which contains information about the relative orientation of two consecutive joint axes (see e.g., Algorithm 1).

Once $\mathbf{F}'_i$ and ϕ_i have been obtained, the synthesis matrix is computed as follows:

$$\mathbf{F}_i = \mathbf{F}'_i R_z(\phi_i) = \begin{pmatrix} r_{x_i} & s_{x_i} & t_{x_i} & u_{x_i}\\ r_{y_i} & s_{y_i} & t_{y_i} & u_{y_i}\\ r_{z_i} & s_{z_i} & t_{z_i} & u_{z_i}\\ 0 & 0 & 0 & 1\end{pmatrix}, \tag{2.9}$$

from which the parameters of the extended D-H convention for the corresponding arm-link of the manipulator can be obtained. In fact, considering the extended D-H parameters of an

[1] atan2(a, b) is the function that gives the arc tangent of $\frac{a}{b}$, considering the proper quadrant of the point $[a,\, b]$.

arm-link and an auxiliary rotation ϕ_i'[1], the same transformation of $\mathbf{F}_i$ can be written as:

$$\begin{aligned}\mathbf{L}_i =& T_z(-p_i)R_z(\phi_i')T_x(a_i)R_x(\alpha_i)T_z(n_i)\\ =& \begin{pmatrix} C_{\phi_i'} & -C_{\alpha_i}S_{\phi_i'} & S_{\alpha_i}S_{\phi_i'} & a_iC_{\phi_i'}+n_iS_{\alpha_i}S_{\phi_i'}\\ S_{\phi_i'} & C_{\alpha_i}C_{\phi_i'} & -S_{\alpha_i}C_{\phi_i'} & a_iS_{\phi_i'}-n_iS_{\alpha_i}C_{\phi_i'}\\ 0 & S_{\alpha_i} & C_{\alpha_i} & n_iC_{\alpha_i}-p_i\\ 0 & 0 & 0 & 1\end{pmatrix}.\end{aligned} \tag{2.10}$$

By equating (2.10) and (2.9), we obtain the relations to infer the parameters of the extended D-H convention for the ith constituted link (a_i, α_i, γ_i, p_i, n_i):

$$a_i = u_{x_i}\,r_{x_i} + u_{y_i}\,r_{y_i},\quad \alpha_i = \operatorname{atan2}(s_{z_i},\, t_{z_i}), \tag{2.11}$$

$$\gamma_i = \overbrace{\operatorname{atan2}(r_{y_i},\, r_{x_i})}^{\phi_i'} + \delta_{i-1}^J - \phi_{i-1}. \tag{2.12}$$

If $s_{z_i} \neq 0$:

$$n_i = \frac{(u_{x_i}\,r_{y_i} - u_{y_i}\,r_{x_i})}{s_{z_i}},\quad p_i = \frac{(u_{x_i}\,t_{z_i}\,r_{y_i} - u_{z_i}\,s_{z_i} - u_{y_i}\,t_{z_i}\,r_{x_i})}{s_{z_i}}. \tag{2.13}$$

If $s_{z_i} = 0$:

$$n_i = 0,\quad p_i = -u_{z_i}. \tag{2.14}$$

While the obtainment of the analytical relations (2.11) and (2.13) follows directly from equating (2.10) and (2.9), the choice of (2.14) follows from the extended D-H convention for the case of parallel consecutive joint axes. Instead, (2.12) is obtained from a slightly more articulated reasoning that can be explained as follows. Starting from the axis x_{i-1} one can subtract the angle ϕ_{i-1} and add δ_{i-1}^J to reach a frame parallel to the first auxiliary frame of module$_{j-1}$. The relation for γ_i is obtained by additionally adding the rotation ϕ_i' that brings the x axis of that auxiliary frame to match the one of the extended D-H frame.

The procedure described above for processing the module data and automatically obtaining the table of the (extended) D-H parameters for an assembled arm is compactly presented in Algorithm 1. An implementation of this procedure has been made available with the function $ModRob2DHext(\dots)$ in the repository *OTFCtrlModRob* downloadable from [44].

[1] The additional auxiliary angle ϕ_i' is the rotation about z of the first auxiliary frame of module$_{j-1}$ that is required in order to let the x axis of this auxiliary frame match the one of the respective extended D-H frame.

Algorithm 1 Algorithm for obtaining the extended D-H parameters of each subsequent pair of joints of an assembled modular robot using module data.

```
Input: Module data for kinematics
Output: Extended D-H parameters
$\phi_0 \leftarrow 0$
for the $i$th pair of subsequent joint modules do
    $\mathbf{z}_{i-1} \leftarrow [0,\ 0,\ 1]^T$
    $\mathbf{F}'_i \leftarrow$ use (2.6) and the module data for kinematics
    $[\mathbf{R}'_i,\ \mathbf{U}'_i] \leftarrow$ extract from $\mathbf{F}'_i$ as in (2.6)
    $\mathbf{z}_i \leftarrow \mathbf{R}'_i(:,3)$
    if $\|\mathbf{z}_i \times \mathbf{z}_{i-1}\| == 0$ and $\|\mathbf{U}'_i(1:2,1)\| == 0$ then        ▷ joint axes are overlapped
        $\phi_i \leftarrow 0$
    else if $\|\mathbf{z}_i \times \mathbf{z}_{i-1}\| == 0$ then        ▷ joint axes are parallel
        $\mathbf{V} \leftarrow \mathbf{R}'^T_i \mathbf{U}'_i$
        $\phi_i \leftarrow \mathrm{atan2}(v_y, v_x)$
    else        ▷ joint axes are skew or intersect
        $\mathbf{V}' \leftarrow \mathbf{z}_i \times \mathbf{z}_{i-1}$
        $\mathbf{V} \leftarrow \mathbf{R}'^T_i \mathbf{V}'$
        $\phi_i \leftarrow \mathrm{atan2}(v_y, v_x)$
    end if
    $\mathbf{F}_i \leftarrow \mathbf{F}'_i R_z(\phi_i)$
    $[a_i,\ \alpha_i,\ \gamma_i,\ p_i,\ n_i] \leftarrow$ use equations (2.11)-(2.14) with elements of $\mathbf{F}_i$
end for
```

2.4 Dynamic Modeling using Module Data

The large body of research results on modeling classical fixed-configuration manipulators can be leveraged to model the dynamics of an assembled arm. In this section, the module data for the dynamics of each module that may compose an arm-link are first defined. Subsequently, the procedure to process the module data of an assembled arm and obtain the dynamical parameters of each arm-link is devised. Once the parameters of each arm-link are obtained, classical algorithms for the dynamics of fixed-configuration robots can be used. A comprehensive survey of robot dynamics can be found in [43]. Among the most popular approaches are the Lagrange and the Newton-Euler (N-E) formulations. While the first one is based on the study of the kinetic and potential energy of the arm, the second one is based on the balance of forces/torques of the fundamental equations of motion for each arm-link. A computationally efficient (recursive) variant of the Newton-Euler method has first been proposed in [71] with a complexity that grows in a linear fashion with the number of the joints. A comparative study between a recursive Lagrangian and a recursive N-E approach can be found in [48], in which the latter is superior in terms of computational efficiency. Enhanced versions of this algorithm have been presented: for example, in [94] the authors consider inertia effects of the rotors, and in [34] the authors provide a modified version of the algorithm for fault detection and passivity-based control. The latter is an important result for efficient implementation of passivity-based control methods that are among the most effective model-based trajectory tracking controllers.

The recursive N-E scheme can be directly used for numerically computing the dynamical model so that no software for symbolic variables manipulation is required. This characteristic makes this class of algorithms an ideal choice for the scope of this thesis. We briefly recall the basic idea of this algorithm to initialize the notation and because it is crucial for subsequent analysis.

Recalls on the Recursive N-E Formulation

The algorithm is composed of two recursions, one for kinematics (forward recursion) and one for the balance of forces and torques (backward recursion) for each link of the arm. Since the kinematics are required for the forward recursion, the kinematic description of the arm automatically obtained with the procedure described in Section 2.3 can be exploited. This recursion computes the linear acceleration of the center of mass of the links, starting from the base to the end effector. For the backward recursion, the algorithm is based on the balance of forces/torques of the fundamental equations of motion for translation and rotation of each ith arm-link, considered as a rigid body. In fact, the backward recursion is based on the following equations [96, Section 7.5]:

$$\mathbf{f}_i^i - \mathbf{f}_{i+1}^i = m_i \ddot{\mathbf{C}}_i^i, \tag{2.15}$$

$$\mathbf{n}_i^i + \mathbf{f}_i^i \times \left(\mathbf{r}_{D_{i-1},D_i}^i + \mathbf{r}_{D_i,C_i}^i\right) - \mathbf{n}_{i+1}^i + \mathbf{f}_{i+1}^i \times \mathbf{r}_{D_i,C_i}^i = \mathbf{I}_i^i \dot{\boldsymbol{\omega}}_i^i + \boldsymbol{\omega}_i^i \times \left(\mathbf{I}_i^i \boldsymbol{\omega}_i^i\right), \tag{2.16}$$

where (with reference to Figure 2.6a) the vectors $\mathbf{f}_i^i/\mathbf{f}_{i+1}^i$ and $\mathbf{n}_i^i/\mathbf{n}_{i+1}^i$ are action-reaction force and torque vectors exerted on the ith link, $\boldsymbol{\omega}_i^i$ is the angular velocity of the link, $\ddot{\mathbf{C}}_i^i$ is the linear acceleration of the center of mass, and $\mathbf{r}_{D_{i-1},C_i}^i/\mathbf{r}_{D_i,C_i}^i$ are the vectors from the application point of the forces $\mathbf{f}_i^i/\mathbf{f}_{i+1}^i$ to the center of mass. Further, m_i is the mass of the link, and $\mathbf{I}_i^i$ its inertia tensor. The superscript of vectors denotes in which frame they are defined. The force vector due to gravity is not present in (2.15), since it is assumed that the contribution of gravitational acceleration is included in the base frame's acceleration during initialization of the forward recursion. A complete description of the standard recursive N-E algorithm can be found in the Appendix A.1.

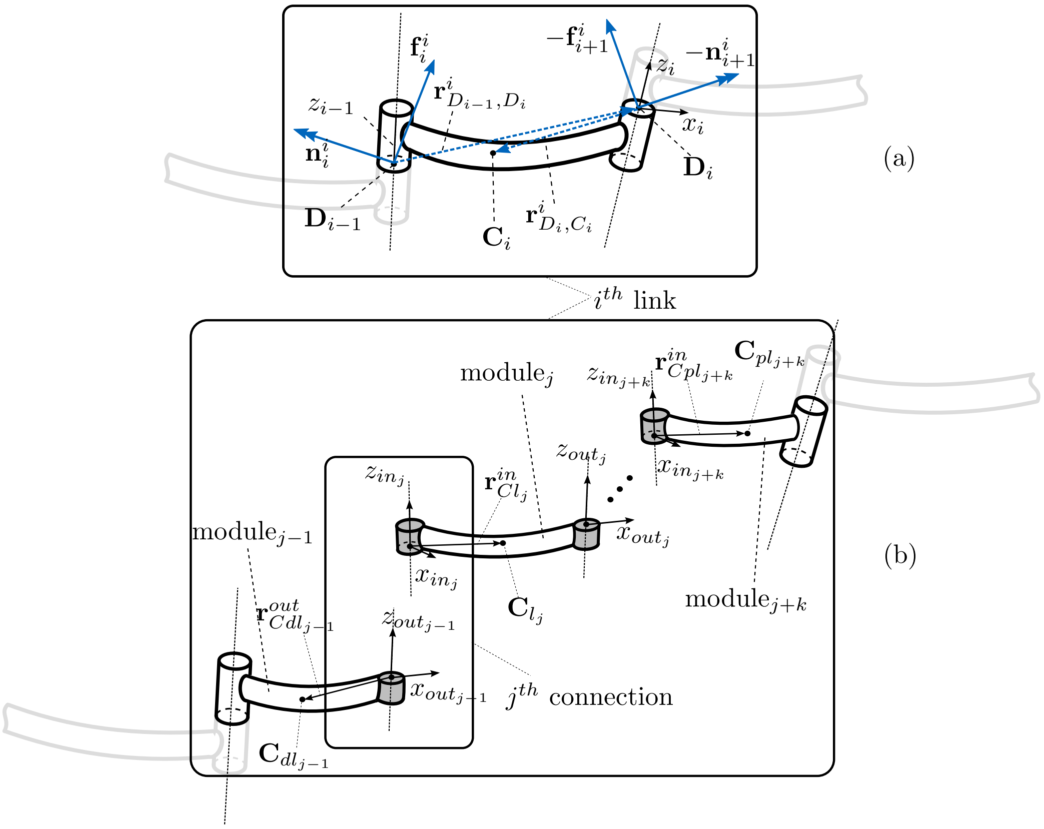


Figure 2.6: Illustration of a connection of modules (b) which constitutes the ith link of the manipulator (a), involving parameters for dynamics. Connectors are indicated in light-grey.

2.4.1 Definition of the Module Data for Dynamics

With the above recalls made, it is now evident that the algorithm considered for computing the arm dynamics require the mass of each link m_i, its inertia tensor $\mathbf{I}_i^i$, the coordinates of the center of gravity, and the coordinates of the application point of forces. More specifically, as shown in Figure 2.6a, a link-fixed frame with origin in $\mathbf{D}_i$ and assumed to be that of the resulting extended D-H frame is considered. With respect to this frame, the coordinates of the center of mass $\mathbf{r}_{D_i,C_i}^i$ and the coordinates of the connection point with the previous joint $\mathbf{r}_{D_{i-1},D_i}^i$ are required. The above-mentioned parameters for dynamics are those of an assembled link of the manipulator. Such a link, however, is composed of modules as shown in the generic connection of Figure 2.6b.

In the generic connection shown in the exploded-view of Figure 2.6b, module$_{j-1}$ (joint module), k link modules, and a subsequent module$_{j+k}$ (joint module) are involved for establishing

Joint Module	Proximal	m^{pl}	$\mathbf{I}^{in}_{pl}$	$\mathbf{r}^{in}_{Cpl}$	
	Distal	m^{dl}	$\mathbf{I}^{out}_{dl}$	$\mathbf{r}^{out}_{Cdl}$	
	Joint	$f(\dot{q})$	σ_m	I_m	σ_r
Link Module		m^l	$\mathbf{I}^{in}_{l}$	$\mathbf{r}^{in}_{Cl}$	

Table 2.2: Module data for dynamic modeling.

the connection of the ith arm-link. As explicated from (2.15) and (2.16), the dynamical parameters of each ith link of the manipulator are required. Therefore, the module data that are required for dynamics are the dynamical parameters of the rigid bodies that compose the link: mass, coordinates of the center of mass, and the inertia tensor of the distal part (m^{dl}, $\mathbf{r}^{out}_{Cdl}$, $\mathbf{I}^{out}_{dl}$), of the proximal part (m^{pl}, $\mathbf{r}^{in}_{Cpl}$, $\mathbf{I}^{in}_{pl}$), and of the link modules (m^{l}, $\mathbf{r}^{in}_{Cl}$, $\mathbf{I}^{in}_{l}$). As denoted by the superscripts, the coordinates of the center of mass and the inertia tensor for proximal parts and link modules are expressed with respect to the input frame. For distal parts, instead, they are expressed with respect to the output frame. This choice allows a simple implementation of a recursive procedure to synthesize the module data and obtain the parameters of each arm-link. This procedure is detailed in the following subsection.

The parameters described so far allow a description of the dynamics of the arm for its principal structure. However, this is typically not sufficient due to non-negligible contributions to the dynamics from the actuation unit components, such as friction and rotor inertia effects. In this thesis the actuation units are part of joint modules. Thus, parameters to model these dynamic effects have to be included as well in the module data for joint modules. Therefore, the friction model $f(\dot{q})$ (or the corresponding coefficients) should be included in the module data. Additionally, to capture the inertia effects of the rotor, it is common practice to include its inertia I_m by adding it to the diagonal of the robot inertia matrix through the square of the gear ratio σ_r [5, 94]. Further, since the typical control signal of electric motors is current, the respective torque constant σ_m is required. Finally, the complete set of data for each module required for dynamics are collected in Table 2.2.

For example, if all numerical entries of the modular parameters for both kinematics and dynamics were single format, the memory required for storing the module data of a simple joint module would be 148 bytes, or 64 bytes for a link module.

2.4.2 From the Module Data to the Dynamical Model

Using the module data for kinematics and dynamics previously defined, the automatic procedure for obtaining the dynamical parameters of each constituted link of a modular manipulator will now be devised. With reference to the generic connection of modules in Figure 2.6b, it is convenient to first consider that the module$_j$ (a link module) gets connected to the distal part of module$_{j-1}$ (a joint module). This is realized through the jth connection shown in Figure 2.6b. Once this connection is performed, the distal part of module$_{j-1}$ and the module$_j$ together create a new auxiliary distal part. Denoting with "io" the matched input-output frame of the connection, the dynamical parameters of the new auxiliary distal part can be computed as follows:

$$m_j^a = m_{j-1}^{dl} + m_j^l, \quad \mathbf{I}_{a_j}^{io} = \mathbf{I}_{dl_{j-1}}^{out} + \mathbf{I}_{l_j}^{in}, \quad \mathbf{r}_{Ca_j}^{io} = \frac{m_{j-1}^{dl}\,\mathbf{r}_{Cdl_{j-1}}^{out} + m_j^l\,\mathbf{r}_{Cl_j}^{in}}{m_j^a}. \tag{2.17}$$

These parameters are those of the new auxiliary distal part and are denoted with the label a. The above computed inertia tensor and coordinates of the center of mass are expressed in the matched input-output frame and should now be transformed, to be expressed in the output frame of the auxiliary distal part. This can be done by using homogeneous transformations and Steiner's theorem [96, Appendix B.2]. Given the homogeneous transformation matrix of the coordinate transformation of the output frame with respect to the matched "io" frame that is computed as follows:

$$\mathbf{A}_{out,a_j}^{io,a_j} = R_z(-\delta_j^{l,in})T_z(-p_j^l)T_x(a_j^l)R_x(\alpha_j^l)T_z(n_j^l)R_z(\delta_j^{l,out}) = \begin{pmatrix} \mathbf{R}_{out,a_j}^{io,a_j} & \mathbf{U}_{out,a_j}^{io,a_j} \\ \mathbf{0}^T & 1 \end{pmatrix}, \tag{2.18}$$

the coordinates on the center of mass with respect to the output frame can be obtained from

$$\begin{pmatrix} \mathbf{r}_{Ca_j}^{out} \\ 1 \end{pmatrix} = \left(\mathbf{A}_{out,a_j}^{io,a_j}\right)^{-1} \begin{pmatrix} \mathbf{r}_{Ca_j}^{io} \\ 1 \end{pmatrix}. \tag{2.19}$$

Further, the inertia tensor expressed in the output frame can be computed as follows:

$$\mathbf{I}_{a_j}^{out} = \left(\mathbf{R}_{out,a_j}^{io,a_j}\right)^T \left(\mathbf{I}_{a_j}^{io} - m_j^a\,\mathbf{S}^T(\mathbf{r}_{Ca_j}^{io})\mathbf{S}(\mathbf{r}_{Ca_j}^{io})\right)\mathbf{R}_{out,a_j}^{io,a_j} + m_j^a\,\mathbf{S}^T(\mathbf{r}_{Ca_j}^{out})\mathbf{S}(\mathbf{r}_{Ca_j}^{out}), \tag{2.20}$$

where $\mathbf{S}(\,\cdot\,)$ denotes a skew-symmetric matrix of the type

$$\mathbf{S}(\mathbf{U}) = \begin{pmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{pmatrix}, \ \mathbf{U} = \begin{pmatrix} u_x & u_y & u_z \end{pmatrix}^T.$$

Now, still referring to Figure 2.6b as a generic connection, an additional module$_{j+1}$ (link module) could be present. In this case it is now easy to see that a new auxiliary distal part is established, once this new link module is connected to the previously obtained auxiliary distal part. The same operations described above can be applied (*mutatis mutandis*) to obtain the dynamical parameters of this second auxiliary distal part. Following this approach, a forward recursion can be implemented for the k link modules that compose the arm-link under study when $k > 0$.

When the above mentioned recursion is completed, the last connection to consider is between the kth auxiliary distal part and the proximal part of module$_{j+k}$. This situation is equivalent to the case of $k = 0$, which implies that no link modules are involved in the assembly of the arm-link. In fact, the last connection to consider is between a distal part (which is an auxiliary one when $k > 0$) and a proximal part of two joint modules. This last connection establishes the ith arm-link under consideration. The last operations to perform aim at obtaining the dynamical parameters of the arm-link, required for running the recursive N-E algorithm as described at the beginning of Subsection 2.4.1: m_i, $\mathbf{I}^i_i$, $\mathbf{r}^i_{D_i,C_i}$, $\mathbf{r}^i_{D_{i-1},D_i}$. These parameters can now be computed as follows:

$$m_i = m^{dl}_{j-1+k} + m^{pl}_{j+k}, \quad \mathbf{I}^{io}_i = \mathbf{I}^{out}_{dl_{j-1+k}} + \mathbf{I}^{in}_{pl_{j+k}}, \quad \mathbf{r}^{io}_{C_i} = \frac{m^{dl}_{j-1+k}\,\mathbf{r}^{out}_{Cdl_{j-1+k}} + m^{pl}_{j+k}\,\mathbf{r}^{in}_{Cpl_j}}{m_i}, \tag{2.21}$$

$$\begin{pmatrix} \mathbf{r}^i_{D_i,C_i} \\ 1 \end{pmatrix} = \left(\mathbf{A}^{io}_i\right)^{-1} \begin{pmatrix} \mathbf{r}^{io}_{C_i} \\ 1 \end{pmatrix}, \tag{2.22}$$

where

$$\mathbf{A}^{io}_i = R_z(-\delta^{pl}_j)T_z(-p^{pl}_j)T_x(a^{pl}_j)R_x(\alpha^{pl}_j)T_z(n^{pl}_j)R_z(\phi_i)T_z(-n_i) = \begin{pmatrix} \mathbf{R}^{io}_i & \mathbf{U}^{io}_i \\ \mathbf{0}^T & 1 \end{pmatrix}, \tag{2.23}$$

and

$$\mathbf{I}^i_i = \mathbf{R}^{io^T}_i \Big(\mathbf{I}^{io}_i - m_i\,\mathbf{S}^T(\mathbf{r}^{io}_{C_i})\mathbf{S}(\mathbf{r}^{io}_{C_i})\Big)\mathbf{R}^{io}_i. \tag{2.24}$$

The vector $\mathbf{r}^i_{D_{i-1},D_i}$ can be computed with the synthesized parameters for kinematics of the arm-link. This information can be extracted from the homogenous transformation matrix describing the pose of the D-H frame of the arm-link $i-1$ (see Figure 2.6a) with respect to the D-H frame of the arm-link i. This homogenous transformation matrix is denoted by $\mathbf{A}^i_{i-1}$ and computed using (2.3) as follows:

$$\mathbf{A}^i_{i-1} = \left(\mathbf{A}^{i-1}_i\right)^{-1} = \begin{pmatrix} \mathbf{R}^i_{i-1} & \mathbf{U}^i_{i-1} \\ \mathbf{0}^T & 1 \end{pmatrix}. \tag{2.25}$$

Finally, from the last column of the matrix in (2.25), the last parameter is obtained as

$$\mathbf{r}^{i}_{D_{i-1},D_i} = -\mathbf{U}^{i}_{i-1} = \begin{pmatrix} a_i & d_i S_{\alpha_i} & d_i C_{\alpha_i} \end{pmatrix}^T . \tag{2.26}$$

Please note that the procedure described above for processing the module data and obtaining the dynamical parameters of the arm is simpler than the one for kinematics. Therefore, its presentation with an algorithm environment is not included in this thesis. However, an implementation of this procedure has been made available with the function $ModRob2Dynpar(\cdots)$ in the repository *OTFCtrlModRob* downloadable from [44].

Obtainment of the dynamical model follows directly. Starting from the module data, a table of the D-H parameters **DH** is computed (as described in Section 2.3). Second, the dynamical parameters of all arm-links are obtained with the procedures described in this section and are included in a table **DynPar**, which additionally contains the actuator-related parameters (see third row of Table 2.2). With this information at hand, the standard recursive N-E algorithm $NEA_g\big(\cdots\big)$ reported in Appendix A.1 can be directly used for obtaining the dynamical model:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{c}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{f}(\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) = NEA_g\big(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}},\mathbf{DH},\mathbf{DynPar}\big). \tag{2.27}$$

2.5 Motion Control

This section details how the motion control problem is solved in the presented framework. It is worth stressing that all the methods discussed next start from the module data and exploit the procedures described in the previous sections for obtaining the assembled robot description automatically. Therefore, the centralized controller is automatically deployed without user intervention starting from the module data. The control architecture of the framework of this chapter is illustrated in Figure 2.7. In addition to the blocks of kinematic and joint-space control as in a classical structure (see e.g., [95]), the blocks for processing the module data are added to deliver the assembled-robot data. This combination enables the automatic generation of the control of the arm, once the robot is assembled and a data structure containing the module data of the arm (**ModRob**) is created. This architecture is simple: it allows tracking of task space trajectories by solving the inverse kinematic problem online (using closed loop inverse kinematics schemes) to obtain the reference trajectories in joint space ($\mathbf{q}_d$, $\dot{\mathbf{q}}_d$, $\ddot{\mathbf{q}}_d$) that are tracked by means of a joint-space tracking controller. Both the kinematic control and joint-space control are automatically deployed using the automatically generated robot description. The following subsections detail how the kinematic and joint-space control can be implemented.

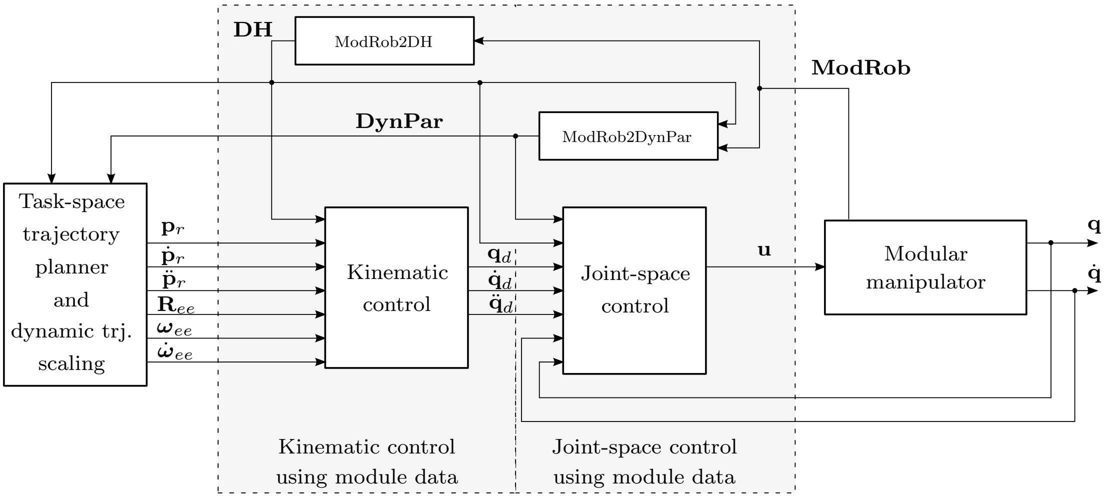


Figure 2.7: Architecture of the automatically deployable tracking controller.

2.5.1 Joint-Space Control

As mentioned in the problem statement of this thesis, the main aim is to have an automatically generated controller which can globally track sufficiently smooth trajectories. Thanks to the automatic obtainment of the robot description, this framework can directly consider two of the most effective model-based joint space controllers, which can be directly implemented: inverse-dynamics control (see e.g., [96, Section 8.5.2]) and passivity-based control (see e.g., [102, Section 8.4]). While the first one relies on the perfect knowledge of the overall system dynamics for compensating the nonlinear and coupling terms through feedback, the second one exploits the property in (2.2) and avoids the direct inversion of the dynamics with better robustness against model uncertainties [29]. The latter is also particularly suitable for adaptive control [98] and robust control schemes [104] [125]. To properly handle assembly dependent friction model uncertainties, a version of the passivity-based tracking controller with an adaptive friction compensation action is also fostered. Effectiveness of all these controllers has been verified with experiments, as presented in Section 2.6.

Inverse-Dynamics Control

The inverse dynamics control scheme is simple. It exploits the model knowledge to directly cancel couplings through feedback and obtain a linear and decoupled system from a new auxiliary control input variable. This new auxiliary input vector $\mathbf{y}$ is selected to provide asymptotically stable dynamics of the error in joint space. The inverse dynamics control law is implemented using:

$$\mathbf{u}_{ID} = \mathbf{M}\left(\mathbf{q}\right)\mathbf{y} + \mathbf{c}\left(\mathbf{q}, \dot{\mathbf{q}}\right) + \mathbf{f}\left(\dot{\mathbf{q}}\right) + \mathbf{g}\left(\mathbf{q}\right) \tag{2.28}$$

for the system in (2.1), which provides $\ddot{\mathbf{q}} = \mathbf{y}$. A typical choice of $\mathbf{y}$ is

$$\mathbf{y} = \ddot{\mathbf{q}}_d + \mathbf{K}_D\dot{\mathbf{e}} + \mathbf{K}_P\mathbf{e}, \tag{2.29}$$

which provides

$$\ddot{\mathbf{e}} + \mathbf{K}_D\dot{\mathbf{e}} + \mathbf{K}_P\mathbf{e} = \mathbf{0}, \tag{2.30}$$

where $\mathbf{K}_P$, $\mathbf{K}_D$ are diagonal positive definite gain matrices of proper dimensions. Please note that integral action to reject constant input disturbance can be included by simply adding $\mathbf{K}_I \int_0^t \mathbf{e}\, d\tau$ on the right-hand side of (2.29) with an additional gain matrix $\mathbf{K}_I$ [18, Section 2.4].

By using the automatically generated robot description as described in the previous section, this controller can be efficiently implemented as follows:

$$\mathbf{u}_{ID} = \mathbf{M}(\mathbf{q})\mathbf{y} + \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{f}(\mathbf{q}) + \mathbf{g}(\mathbf{q}) = NEA_g\big(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{y}, \mathbf{DH}, \mathbf{DynPar}\big). \tag{2.31}$$

In light of the fact that each coordinate of (2.30) is a second-order linear system, the tuning is simple and can be done by selecting a user-defined natural frequency ω_n and damping ratio ζ. This is performed by using e.g., $\mathbf{K}_P = \omega_n\mathbf{I}$ and $\mathbf{K}_D = 2\zeta\omega_n\mathbf{I}$. This scheme makes the tuning independent from the specific robot configuration so that, in principle, no user intervention is required after a reconfiguration.

Passivity-Based Control

Contrary to the inverse dynamics control scheme, passivity-based tracking controllers do not rely on the complete cancellation of all couplings through feedback. Instead, they exploit the property in (2.2), which can be obtained from the conservation of energy as shown in [98]. This control approach is expected to provide better robustness to model uncertainty with respect to the inverse dynamics controller since it does not rely on the perfect cancellation of the couplings [18, 29]. The classical passivity-based tracking controller is implemented using

$$\mathbf{u}_{PB} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}}_a + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}_a + \mathbf{f}(\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) + \mathbf{\Lambda}\mathbf{r}, \tag{2.32}$$

with $\dot{\mathbf{q}}_a = \dot{\mathbf{q}}_d + \mathbf{K}_r\mathbf{e}$, $\mathbf{r} = \dot{\mathbf{e}} + \mathbf{K}_r\mathbf{e}$, and where $\mathbf{\Lambda}$ and $\mathbf{K}_r$ are diagonal positive definite matrices of proper dimensions. By applying the passivity-based control law in (2.32) to the system in (2.1), the following closed loop system is obtained:

$$\mathbf{M}(\mathbf{q})\dot{\mathbf{r}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\mathbf{r} + \mathbf{\Lambda}\mathbf{r} = \mathbf{0}. \tag{2.33}$$

Global asymptotic stability follows by exploiting property (2.2) as described e.g., in [18, Section 2.3]. Integral action to reject constant input disturbance can be included by simply adding $\mathbf{K}_I \int_0^t \mathbf{r}\, d\tau$ on the right-hand side of (2.32) with an additional positive definite matrix $\mathbf{K}_I$ of proper dimensions as described in [18, Section 2.4].

Within the framework presented in this chapter, this controller is implemented numerically using a recursive Newton-Euler algorithm. The structure of (2.32) does not allow the direct use of the algorithm $NEA_g\big(\cdots\big)$ because of its inability to compute the term $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}_a$. A modified version of this algorithm that solves this shortcoming has been proposed in [34], where the authors consider robots with revolute joints. A small extension of it for considering prismatic joints is described in [128]. This modified algorithm that can handle both revolute and prismatic joints is detailed in Algorithm 2 of Appendix A.1, and is denoted in this thesis by $NEA_g^*\big(\cdots\big)$ to provide

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}}_a + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}_a + \mathbf{f}(\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) = NEA_g^*\big(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_a, \ddot{\mathbf{q}}_a, \mathbf{DH}, \mathbf{DynPar}\big). \tag{2.34}$$

With this algorithm at hand, **DH**, and **DynPar** automatically computed after assembly, the passivity based control is directly implemented as follows:

$$\mathbf{u}_{PB} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}}_a + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}_a + \mathbf{f}(\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) + \mathbf{\Lambda}\mathbf{r} = NEA_g^*\big(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_a, \ddot{\mathbf{q}}_a, \mathbf{DH}, \mathbf{DynPar}\big) + \mathbf{\Lambda}\mathbf{r}. \tag{2.35}$$

Passivity-Based Control with Adaptive Friction Compensation

In light of the fact that the load at the joints of a robot may affect the friction model (as reported e.g., in [110] and shown in Figure 2.12), special attention should be paid in the framework of this chapter. This becomes clear when considering that the parameters of the friction model (that are part of the module data) enter directly in **DynPar** and are usually obtained when the intended robot configurations are not yet known. After the modules are characterized singularly, they can compose arbitrary assemblies. The load at the joints cannot be assumed to be known a priori. As a result, the use of the module data with friction parameters estimated for the modules alone during initial characterization may lead to a friction model that is significantly different from the real one after assembling the robot. This aspect can be addressed by introducing an adaptive term to the passivity-based feedback control law.

In order to introduce a term to the feedback control law for having an adaptive friction compensation, the work of [98] can be tailored to this application by assuming that the model uncertainty is significant only with respect to the friction model parameters. By considering the following friction model

$$\mathbf{f}(\dot{\mathbf{q}}) = \boldsymbol{\beta}_v \dot{\mathbf{q}} + \boldsymbol{\beta}_c \text{sign}(\dot{\mathbf{q}}), \tag{2.36}$$

with viscous and static friction coefficients ($\boldsymbol{\beta}_v$ and $\boldsymbol{\beta}_c$, respectively), and assuming that only nominal parameters $\beta_{0v,i}$ and $\beta_{0c,i}$ for each ith joint are available for control, the passivity-based control command with adaptive friction compensation is computed as

$$\mathbf{u}_{PBAFC} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}}_a + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}_a + \hat{\mathbf{f}}(\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) + \boldsymbol{\Lambda}\,\mathbf{r}, \tag{2.37}$$

where

$$\hat{f}_i(\dot{q}_i) = \hat{\beta}_{v,i}(t)\dot{q}_i + \hat{\beta}_{c,i}(t)\text{sign}(\dot{q}_i), \quad \forall i \in \{1, \ldots, N\} \tag{2.38}$$

and $\hat{\beta}_{v,i}(0) = \beta_{0v,i}$, $\hat{\beta}_{c,i}(0) = \beta_{0c,i}$. Using (2.37) in (2.1), after rearrangement it can be shown that the following relation holds:

$$\mathbf{M}(\mathbf{q})\dot{\mathbf{r}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\mathbf{r} + \boldsymbol{\Lambda}\,\mathbf{r} = \hat{\mathbf{f}}(\dot{\mathbf{q}}) - \mathbf{f}(\dot{\mathbf{q}}) = \mathbf{Y}(\dot{\mathbf{q}})\boldsymbol{\Delta}_\beta, \tag{2.39}$$

where

$$\mathbf{Y}(\dot{\mathbf{q}}) = \begin{pmatrix} \dot{q}_1 & \text{sign}(\dot{q}_1) & \ldots & 0 & 0 \\ 0 & 0 & \ddots & 0 & 0 \\ 0 & 0 & \ldots & \dot{q}_N & \text{sign}(\dot{q}_N) \end{pmatrix}, \quad \boldsymbol{\Delta}_\beta = \hat{\boldsymbol{\beta}}(t) - \boldsymbol{\beta} = \begin{pmatrix} \hat{\beta}_{v,1}(t) - \beta_{v,1} \\ \hat{\beta}_{c,1}(t) - \beta_{c,1} \\ \vdots \\ \hat{\beta}_{v,N}(t) - \beta_{v,N} \\ \hat{\beta}_{c,N}(t) - \beta_{c,N} \end{pmatrix}.$$

Global asymptotic stability follows from a similar argument of [96, Section 8.5.4], where the main difference is in the choice of $\mathbf{Y}(\dot{\mathbf{q}})$, provided that the parameters $\hat{\beta}_{v,i}(t)$ and $\hat{\beta}_{c,i}(t)$ are computed using the following adaptive law:

$$\dot{\boldsymbol{\Delta}}_\beta = \dot{\hat{\boldsymbol{\beta}}}(t) = \mathbf{K}_{\Delta_\beta}^{-1}\mathbf{Y}(\dot{\mathbf{q}})^T\mathbf{r}.$$

2.5.2 Kinematic Control

The only missing component for generating a complete controller is the consideration of the solution of the inverse kinematic problem. This problem is typically solved by means of analytical and numerical solutions. As it has been analyzed in [88] in detail, an analytical solution of the inverse kinematic problem can be found only for manipulators that are sufficiently simple and have specific geometries. Since a modular robot can assume, in principle, an arbitrary geometry and can become redundant, analytical solutions for the inverse kinematic problem are not suitable. Numerical approaches can instead be employed by exploiting the differential kinematics, and are not limited to specific robotic structures or non-redundant manipulators.

As shown in Figure 2.7, the kinematic control aims at solving online the inverse kinematic problem, given time varying trajectories of the desired pose of the end effector, to provide the reference trajectories for the joints. The latter can then be tracked by one of the previously mentioned joint-space controllers [95]. A large body of works addressed this problem and many classical effective approaches are well documented in books e.g., [96]. Further, a description of the most effective methods for kinematically redundant robots can be found e.g., in [92].

A clear distinction among inverse kinematics algorithms can be made based on the type of description they employ for orientation. They can be divided into those that use a minimal description of the orientation (Euler angles), those based on the axis-angle representation, and those based on the unit quaternion (Euler parameters). A detailed comparison of these approaches can be found in [24]. From that comparison, the use of the unit quaternion is superior when considering computational efficiency and representation singularities. These singularities arise from the incapability of uniquely mapping a rotation to the minimal representation chosen for some particular orientations. The same issue can also appear when using an axis-angle representation [96, Sections 2.4 and 2.5].

In practice, the use of the unit quaterion for describing the end effector orientation and for computing the orientation error feedback of the kinematic control schemes has unique advantages with respect to Euler angles and axis-angle representation. In fact, it removes the risk of encountering representation singularities, and can be implemented by directly using the geometric Jacobian in place of the analytical one whose obtainment is computationally more expensive [28]. These features make it an ideal choice for the automatic controller generation. Indeed, as it will be presented next, the approaches considered in this framework only require the computation of the $6 \times N$ geometric Jacobian $\mathbf{J}(\mathbf{q}) = [\mathbf{J}_p(\mathbf{q})^T,\ \mathbf{J}_\omega(\mathbf{q})^T]^T$ and its derivative $\dot{\mathbf{J}}(\mathbf{q}, \dot{\mathbf{q}})$. These computations can be done numerically online by using $\mathbf{q}$, $\dot{\mathbf{q}}$, and the automatically obtained kinematic description of the robot as described in Appendix A.3. An implementation of the algorithm can be found in the repository *OTFCtrlModRob* [44] with the function $dJacobian(\dots)$.

Closed-loop inverse kinematics schemes are usually separated into first and second-order ones. First-order schemes provide the joint velocities and positions which let the end effector track a task space trajectory. Drift is counteracted by employing a feedback control loop on joint positions. A description and experimental application of the latter using the unit quaternion can be found in [28]. However, for the purpose of this framework, the computation of the reference joint acceleration is required. In the case that a first order scheme is employed, a practical way for computing the joint acceleration is via numerical differentiation. In contrast, second order methods allow direct computation of the joint reference accelerations. Joint velocities and joint positions are then obtained through integration. Drift is counteracted by employing a feedback control loop on both joint velocities and positions.

Hereafter, a unit quaternion is denoted as a vector of four components e.g., $\mathbf{Q} = [\eta,\, \boldsymbol{\epsilon}^T]^T = [\eta,\, \epsilon_x,\, \epsilon_y,\, \epsilon_z]^T$, where η is the scalar part of the quaternion and $\boldsymbol{\epsilon} = [\epsilon_x,\, \epsilon_y,\, \epsilon_z]^T$ its vectorial part. By considering the required trajectory of the end effector frame specified for positions $\mathbf{p}_r \in \mathbb{R}^3$, orientations $\mathbf{Q}_r = [\eta_r,\, \boldsymbol{\epsilon}_r]^T \in \mathbb{R}^4$, linear velocities and accelerations $(\dot{\mathbf{p}}_r,\, \ddot{\mathbf{p}}_r)$, and angular velocities and accelerations $(\boldsymbol{\omega}_r,\, \dot{\boldsymbol{\omega}}_r)$, the following closed loop inverse kinematic scheme can be implemented:

$$\ddot{\mathbf{q}}_d = \mathbf{J}^{\dagger}(\mathbf{q}_d)\Big(\boldsymbol{\nu} - \dot{\mathbf{J}}(\mathbf{q}_d, \dot{\mathbf{q}}_d)\mathbf{q}_d\Big) - \kappa\Big(\mathbf{I} - \mathbf{J}^{\dagger}(\mathbf{q}_d)\mathbf{J}(\mathbf{q}_d)\Big)\dot{\mathbf{q}}_d, \tag{2.40}$$

with

$$\boldsymbol{\nu} = \begin{bmatrix} \ddot{\mathbf{p}}_r + K_v(\dot{\mathbf{p}}_r - \mathbf{J}_p(\mathbf{q}_d)\dot{\mathbf{q}}_d) + K_p(\mathbf{p}_r - \mathbf{p}_{fk}(\mathbf{q}_d)) \\ \dot{\boldsymbol{\omega}}_r + K_\omega(\boldsymbol{\omega}_r - \mathbf{J}_\omega(\mathbf{q}_d)\dot{\mathbf{q}}_d) + K_o\mathbf{e}_o(\mathbf{q}_d) \end{bmatrix}. \tag{2.41}$$

In (2.40), $\mathbf{J}^{\dagger}$ is the Jacobian pseudo-inverse (or damped least-squares inverse [23] near the kinematic singularities). In (2.40) and (2.41), κ, K_v, K_p, K_ω, and K_o are positive gains, $\mathbf{p}_{fk}(\mathbf{q}_d)$ is the position of the end effector computed with the forward kinematics, and finally,

$$\mathbf{e}_o = \eta_{fk}(\mathbf{q}_d)\boldsymbol{\epsilon}_r - \eta_r\boldsymbol{\epsilon}_{fk}(\mathbf{q}_d) - \mathbf{S}(\boldsymbol{\epsilon}_r)\boldsymbol{\epsilon}_{fk}(\mathbf{q}_d) \tag{2.42}$$

is the quaternion-based orientation error feedback vector [118]. In (2.42), $\eta_{fk}(\mathbf{q}_d)$ and $\boldsymbol{\epsilon}_{fk}(\mathbf{q}_d)$ are the components of the unit quaternion for the orientation of the end effector computed using the forward kinematic function and the current joint variables of the inverse kinematic solution. The feedback variables $\mathbf{q}_d$, $\dot{\mathbf{q}}_d$ are obtained by integrating $\ddot{\mathbf{q}}_d$. In the event that the assembled robot is redundant, second order schemes suffer from floating null-space motions [50]. To account for these cases, the last term on the right-hand side of (2.40) is added to introduce damping in the null space as in [37]. For the considered scheme, asymptotic stability of both positional and orientational error dynamics can be shown by Lyapunov arguments (see e.g., [24, 118]). It is worth stressing that in principle no user intervention is required after reassembling the robot with this scheme, since the kinematic description of the robot

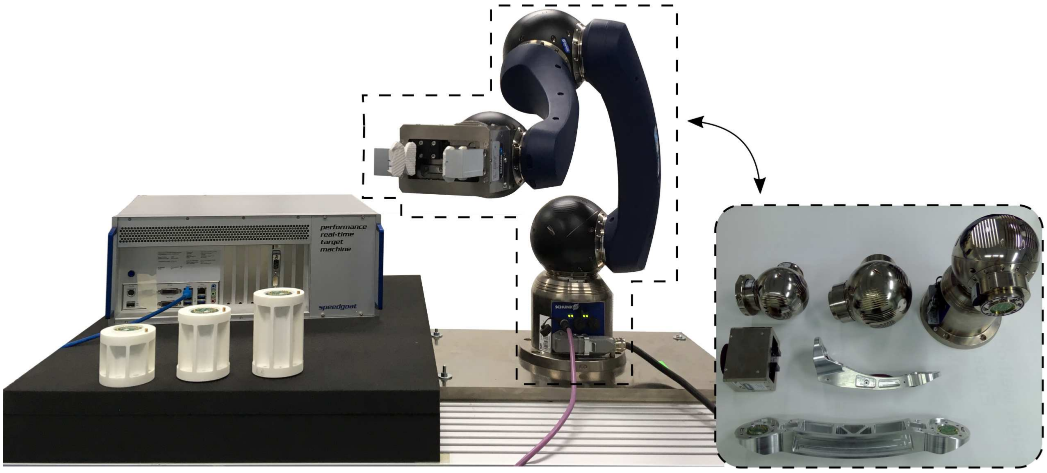


Figure 2.8: Modular robot test bed used in the experiments.

can be automatically obtained and directly used for computing the geometric Jacobian. An implementation of (2.40) has been made available in the repository *OTFCtrlModRob* [44] with the function *invkin_2d*(...).

Even though no saturation limits of the actuators have been considered so far, this is clearly a crucial aspect to consider for a practical implementation. One possibility is to make sure that the required joint-space trajectories are sufficiently slow to let the robot comply with the actuation limits while tracking the trajectories. For this purpose, the scheme presented here can be used in conjunction with the recursive N-E algorithm to find the appropriate time-scaling factor for the task-space trajectories exploiting the results in [49].

2.6 Experimental Application

This section presents experiments of the proposed framework using a modular robot manipulator. In particular, after a description of the modular robot test bed used in the experiments, this section explains how the characterization of the modules from the available set can be performed to obtain the module data according to the notation described in Subsections 2.4.1 and 2.3.2. Once the module data are obtained, for each different assembly, the model and model-based controllers of the assembled arm can be automatically computed. An experimental validation of these models is then presented. The section is concluded with experiments that show the tracking performance obtained with the proposed framework.

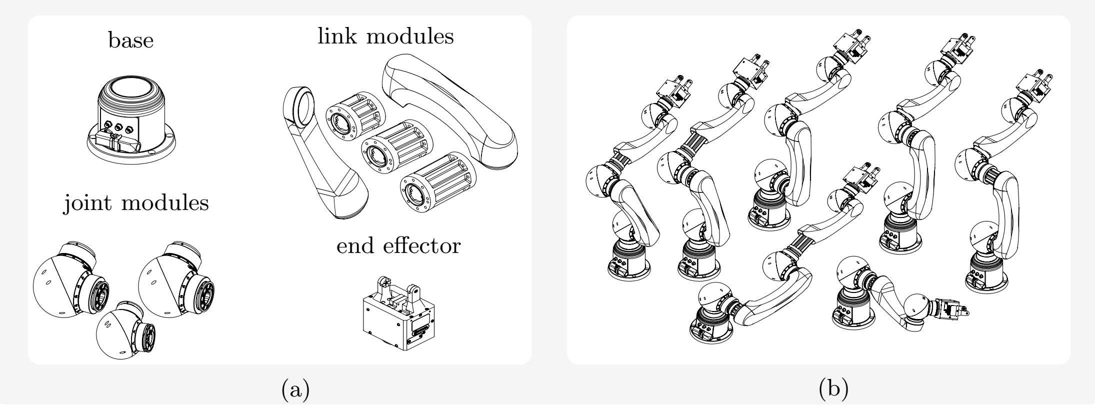


Figure 2.9: Illustration of the modules available for experiemnts (a) and a subset of the possible resulting robot assemblies (b).

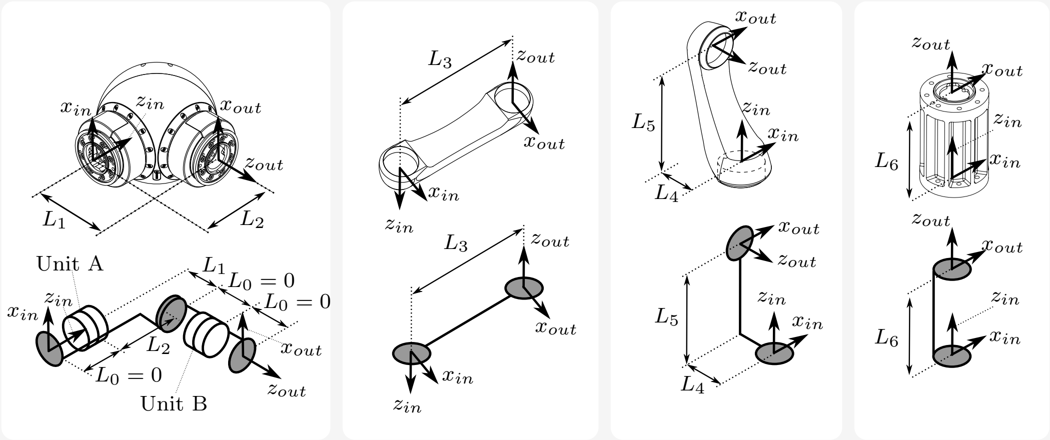
2.6.1 Description of the Modular Robot Test Bed

The test bed considered for this application is shown in Figure 2.8. It is composed of a Schunk LWA 4P modular robot manipulator with an end effector module and three additional 3D printed modules. These additional modules have been designed for enhancing the reconfiguration capabilities of the original, commercially available arm. The set of modules available is shown in Figure 2.9a. This set is composed of a base, five link modules, three complex joint modules, and one end-effector. These joint modules are complex because they introduce two joint axes. Particular considerations are necessary for characterizing such modules, as detailed next in 2.6.2. In Figure 2.9b, a subset of all possible robot assemblies with the available set of modules is illustrated.

The central control unit that implements the presented framework is a Speedgoat Real-Time target PC, equipped with an Intel Core i7-3770K clocked at 3.5 GHz and 4 GB of RAM. The communication with the robot is done via CAN-bus. The communication bus limits the sampling rate that can be used for centralized control at $500Hz$ when assembling a six-axes robot. The controller has been developed using Matlab/Simulink 2015b and implemented on the target machine via automatic code generation.

2.6.2 Derivation of the Module Data

It is now possible to describe how the notation presented in Subsections 2.4.1 and 2.3.2 can be applied to the real scenario with the modules in Figure 2.9a. Even though the complex joint modules differ from the module definiton assumed so far (that has only one joint axis), the



Complex joint module	Unit A	Unit B
a^{pl}	0	0
α^{pl}	0	0
p^{pl}	0	0
n^{pl}	0	0
δ^{pl}	0	0
a^{dl}	0	0
α^{dl}	$-\pi/2$	0
p^{dl}	$-L_2$	0
n^{dl}	L_1	0
δ^{dl}	0	0

Link module	LA
a^l	L_3
α^l	π
p^l	0
n^l	0
$\delta^{l,in}$	$\pi/2$
$\delta^{l,out}$	$-\pi/2$

Link module	LB
a^l	0
α^l	$-\pi/2$
p^l	$-L_5$
n^l	$-L_4$
$\delta^{l,in}$	π
$\delta^{l,out}$	$-\pi$

Link module	LC
a^l	0
α^l	0
p^l	$-L_6$
n^l	0
$\delta^{l,in}$	0
$\delta^{l,out}$	0

Figure 2.10: Characterization of the modules using simple modular units. The models of the components have been derived using the CAD data available from the website of Schunk GmbH & Co. KG.

notation proposed is still applicable since multiple simple joint module units (like the one in Figure 2.3(a)) can be used in sequence for characterizing more complex modules. In this way, the proposed notation can be used to characterize modules with an arbitrary number of joints.

By considering first the complex joint modules, as shown in the top-left of Figure 2.10, one can decompose it into two simple joint model units (unit A and unit B). To parametrize these two units, the corresponding input and output frames should first be placed. Then, the relevant parameters can be extracted according to the notation described in Subsection 2.3.2 for each proximal and distal part. It is important to notice that unit B in Figure 2.10 can be considered fictitious and is introduced with relevant parameters for kinematics at zero to properly describe the geometry of the physical module (the sphere-like body), complementing the use of unit A.

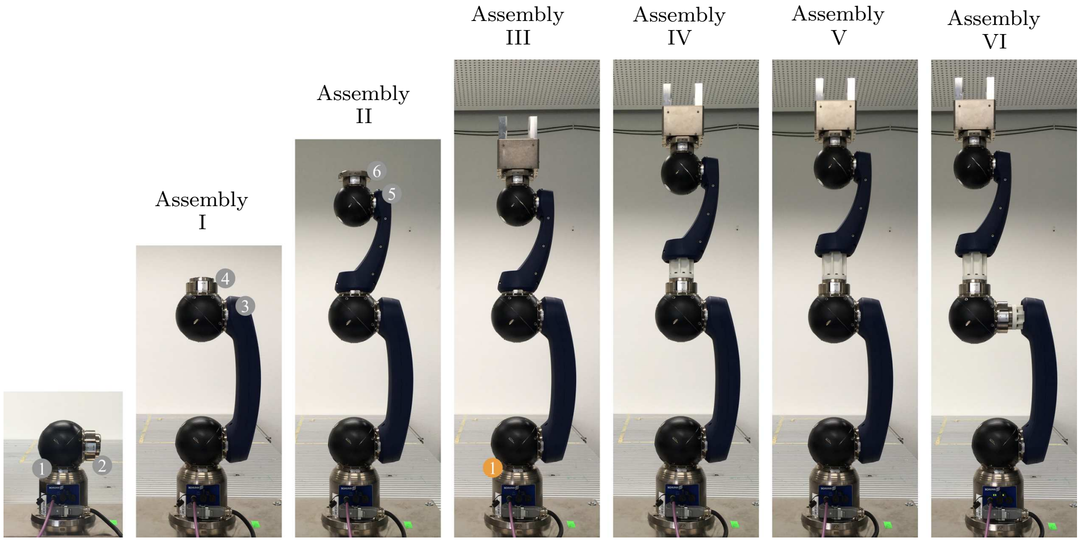


Figure 2.11: Assemblies used for tests.

For characterizing the link modules, the procedure described in Subsection 2.3.2 can be directly applied: the input and output frames are first set, then the auxiliary frames are identified and the kinematic parameters are extracted.

The required module data can be obtained from CAD software and data sheets from the robot manufacturer. It is important to notice that parameters for dynamics of fictitious units can also be set to zero. For example, according to the modeling of the available joint modules using simple joint units (see top-left of Figure 2.10), the mass, the coordinates of the center of mass, and the inertia tensor of the sphere-like body is associated to the distal part of the unit A. The other parts of the modular units (the proximal and distal parts of unit B and the proximal part of unit A) are considered to be fictitious with zero mass and inertia. Please note that even though the mass properties of the sphere-like body have been concentrated in the simple modular unit A, the unit B must still carry the data for kinematics and dynamics of its joint (i.e., type of joint, friction parameters, torque mapping, gear ratio, and rotor inertia). The data relative to the actuators such as friction parameters and rotor inertia can be estimated by performing simple identification of each joint module. Ideally, this can be performed at the time of the module development without applying the distal part on the module. For this simple identification procedure, the following model can be considered:

$$\overbrace{I_m\,\sigma_r^2}^{:=I_{eq}}\ddot{q}_i + \beta_{v,i}\mathrm{sign}(\dot{q}_i) + \beta_{c,i}\mathrm{sign}(\dot{q}_i) = u_i.$$

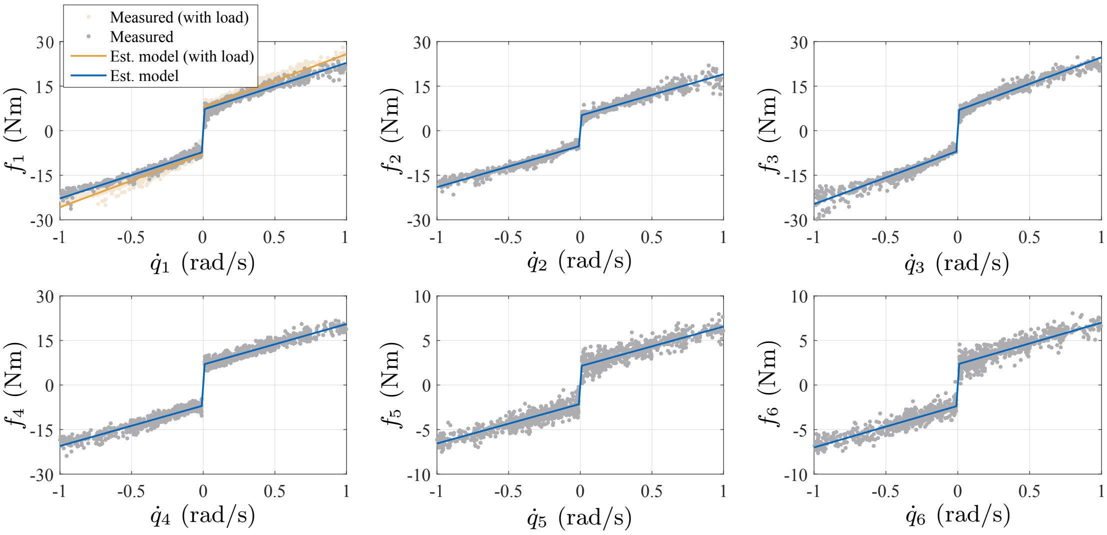


Figure 2.12: Results of the friction identification procedure for all joint modules. This figure additionally shows the influence of the load for the first joint axis. Joint axes numbering is referred to in Figure 2.11.

In light of this model, the parameters can simply be estimated by using the following regression offline when k samples are obtained from a test motion as follows:

$$[I_{eq},\, \beta_{v,i},\, \beta_{c,i}]^T = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{b},$$

where[1]

$$\mathbf{\Phi} = \begin{pmatrix} \ddot{q}_i(1) & \dot{q}_i(1) & \operatorname{sign}(\dot{q}_i(0)) \\ \vdots & \vdots & \vdots \\ \ddot{q}_i(k) & \dot{q}_i(k) & \operatorname{sign}(\dot{q}_i(k)) \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} u_i(1) \\ \vdots \\ u_i(k) \end{pmatrix}.$$

The results of the identification for the joint modules available in this experimental evaluation are shown in Figure 2.12, where the numbering refers to the joint numbers in Figure 2.11. Please note that the same approach can be used with a more complex friction model provided that linearity in the parameters is maintained. Even if the equivalent inertia of the rotor can be obtained from motor data-sheets, the info of auxiliary components e.g., mounting flanges could be missing. This motivated the additional inclusion of the estimation of the equivalent inertia in the identification scheme presented above.

[1] The acceleration can be obtained offline from zero-phase-shift digital filtering [79].

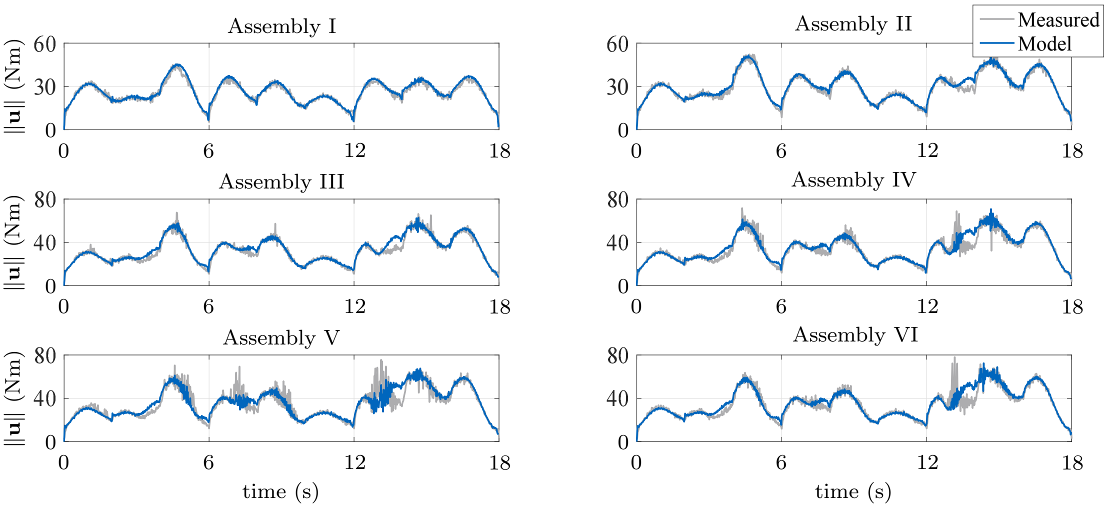


Figure 2.13: Experimental verification of the automatically generated models. Assembly numbering is referred to Figure 2.11.

2.6.3 Validation of the Automatically-Generated Models

The models automatically obtained with the approaches presented in this framework can now be verified. This is possible by letting the robot follow a trajectory and by processing offline the measured joint positions, velocities, and applied control torques. In fact, by comparing the torque applied by the actuators with respect to the predicted torque obtained by using the automatically generated models, the model quality can be observed. The predicted torque can be computed using the recursive N-E algorithm. This algorithm takes as input the automatically generated kinematic/dynamic description of the assembled robot, the measured joint position, velocity, and acceleration vector. Even though the joint accelaration vector is typically not available, it can be computed offline by processing the measured joint position data through numerical differentiation and zero-phase-shift digital filtering [79]. The details on the trajectories used for this test can be found in Table B.1 of Appendix B and have been selected using fifth order polynomials, such that the maximum joint velocities are approached for all axes.

The assemblies from I to VI shown in Figure 2.11 have been tested. The results of these tests are collected in Figure 2.13, in which the comparison between the measured applied torque and the predicted one is shown. These plots show a significantly good model matching and verify the effectiveness of the automatic modeling approach proposed in this framework.

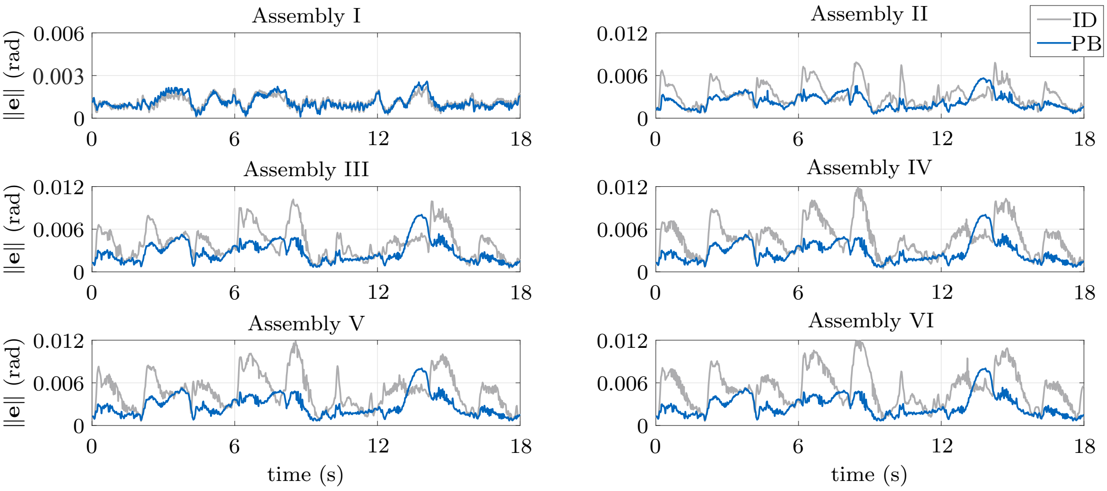


Figure 2.14: Tracking performance comparison when using inverse dynamics control (ID) and passivity-based control (PB) for different assemblies.

2.6.4 Evaluation of the Control Performance

The considered approaches for motion control that are part of this framework are now compared. For the same trajectory used in Subsection 2.6.3 and reported in Appendix B, the tracking performance of different joint space controllers are evaluated. The results of these experiments are collected in Figure 2.14, which shows remarkable insensitivity of the tracking error with respect to changes in the assembly. These plots also show that passivity-based control performs better than inverse dynamics control. Even though a fair comparison among nonlinear controllers for robots is usually difficult due to different nonlinear mapping, tuning parameters have been selected in a fair way by maintaining a similar measurement noise amplification observed in the torque commands as shown in Figure 2.15. The gains used for the inverse-dynamics controller have been selected such that the error dynamics of each axis has $\omega_n = 35$ and $\zeta = 0.65$. The gains used for the passivity-based controller have been chosen as $\mathbf{K}_r = \mathbf{\Lambda} = 42\, diag([1,\, 1,\, 1,\, 1,\, 0.5,\, 0.5])$. Please note that no user intervention has been required when changing the assembly.

The action of the adaptive friction compensation is shown by performing a test with a roughly-tuned version of the automatically generated passivity-based controller and a wrong initial guess for the friction parameters. This is executed by setting them to zero, which is surely not the case in reality. Figure 2.16 shows the results of this test. From this figure, the beneficial action of the adaptive friction compensation can be observed by noting the reduction

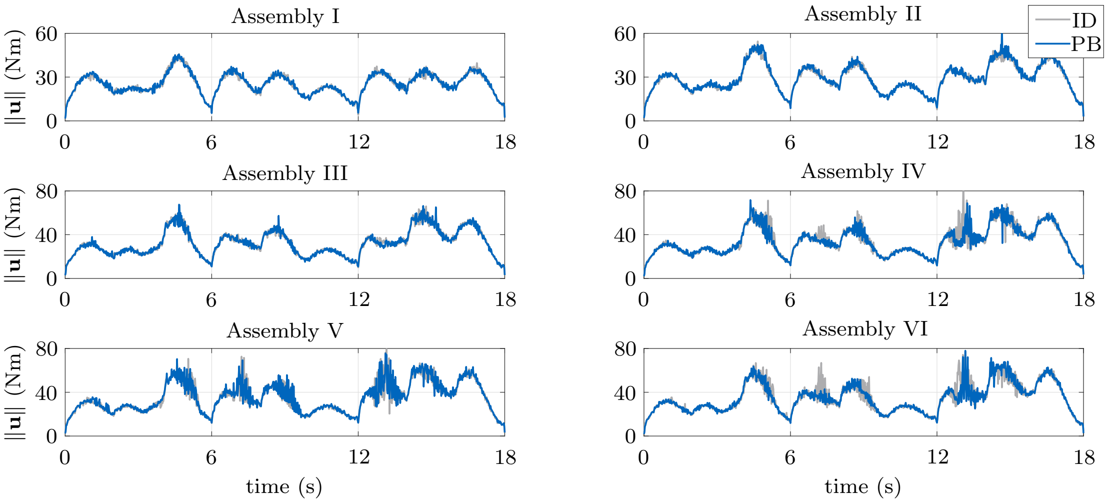


Figure 2.15: Control torque commands required when using inverse dynamics control (ID) and passivity-based control (PB) for the trajectory tracking test of Figure 2.14.

of the norm of the tracking error over time. It is worth mentioning that the event of having a wrong initial guess for the friction coefficients, and a roughly-tuned controller, could happen in practice when quick commissioning of the robot with a new assembly is required. In this experiment the following gains have been used: $\mathbf{K}_r = \mathbf{\Lambda} = 30\mathbf{I}$, $\mathbf{K}_{\Delta_\beta} = \mathbf{I}$.

An additional performance comparison is presented in this thesis to show the benefit of using automatically generated models for control, with respect to the use of a simple model-free approach. The simple approach considered is based on distributed proportional-integral-derivative schemes with decentralized feedforward actions at the joints (PIDs). These decentralized controllers are designed independently for each axis. They allow the closed-loop for each axis to be considered as a second-order linear system with user-defined natural frequency ω_n and damping ratio ζ. This is typically done by considering the actuator dynamics only and by leaving the couplings as disturbances to be rejected (see e.g., in [96, Section 8.3.1]). This performance comparison has been executed for assemblies I, II, and III of Figure 2.11. The gains of the controllers have been fairly selected by setting $\omega_n = 50\,(rad/s)$ and $\zeta = 0.65$ for both inverse-dynamics control and the PIDs. The passivity-based control law with adaptive friction compensation has been tuned such that the tracking performance with assembly I are comparable with the other controllers: $\mathbf{K}_r = \mathbf{\Lambda} = 50\mathbf{I}$, $\mathbf{K}_{\Delta_\beta} = \mathbf{I}$. The results of these tests are collected in Figure 2.17. While the first column of plots of this figure shows a comparable development of the tracking error over time when using assembly I, the control performance of the PIDs dramatically decreases when changing the robot assembly without re-tuning. This

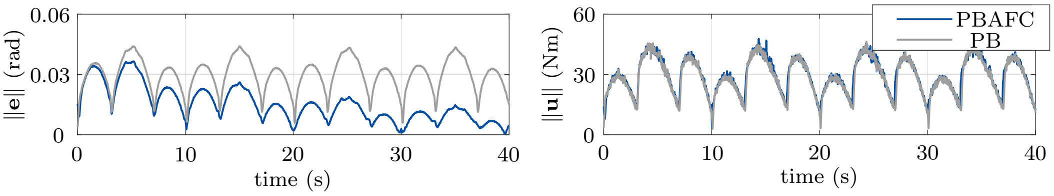


Figure 2.16: Demonstration of the adaptive friction compensation. PB denotes the standard passivity-based controller, and PBAFC denotes the passivity-based control law with adaptive friction compensation.

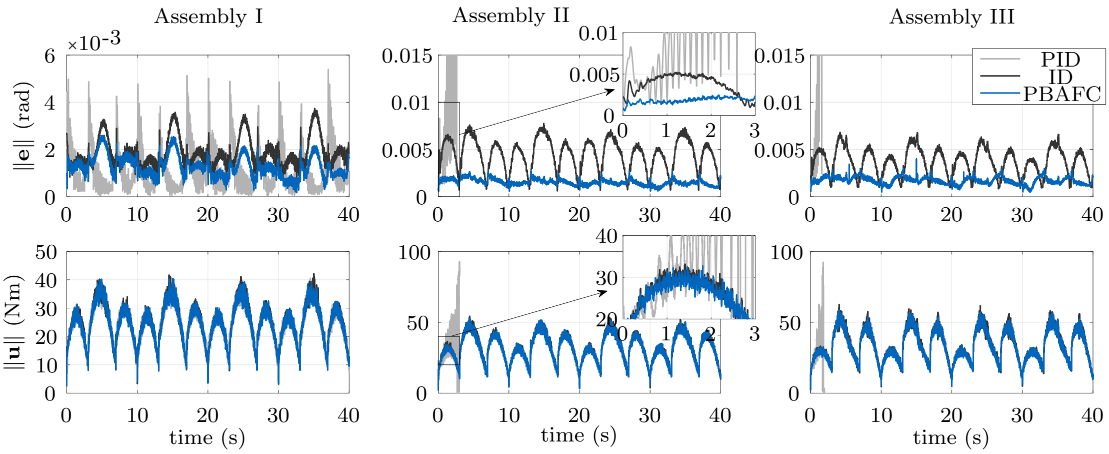


Figure 2.17: Experimental comparison of the tracking performance of different controllers when changing the assembly of the robot. ID denotes the inverse dynamics controller, PBAFC the passivity-based control with adaptive friction compensation, and PID the scheme with decentralized PID controllers.

may frequently happen for the modular reconfigurable robotic setting considered in this thesis. While instabilities are observed for the decentralized PIDs when increasing the complexity of the robotic assembly, the automatically generated model-based controllers show a remarkable insensitivity. It is worth mentioning that the PIDs could be made stable again by manual re-tuning after assembly. However, this would require additional user intervention contrary to the proposed automatic controller design approach.

Now, experimental results of tracking task-space trajectories when using the complete architecture of Figure 2.7 are presented. The task of writing on a white-board has been required for the arm with different assemblies. In particular, for the same task, the robot has been reconfigured from assembly III to VI (with reference to Figure 2.11). For each different assembly the task-space trajectory tracking controller has been automatically generated. The results of these experiments are collected in Figures 2.18, 2.19, 2.20, and 2.21. The figures show the position and orientation error of the inverse kinematic solver on the left. These plots show that the inverse kinematic problem is solved with high precision for all the different assemblies used. On the right of these figures, the task-space tracking performance can be observed. The main limiting factor here is the limited precision of the joint-space controller. Improvements are expected when using a higher sampling rate (which was not possible with the available setup). In light of these results, we can conclude that the tracking performances (both task-space and joint-space) are not significantly affected when changing the assembly of the robot, thanks to the use of the proposed control framework. Finally, it is worth reporting that the maximum total execution time experienced for computing the model-based control commands in these experiments was less than 50 microseconds. The gains used for the experiments of Figures 2.18, 2.19, 2.20, and 2.21 are the following: $\mathbf{K}_r = \mathbf{\Lambda} = 55\, diag([1,\, 1,\, 1,\, 1,\, 0.5,\, 0.5])$, $\mathbf{K}_I = 20\mathbf{I}$, $\kappa = 200$, $K_v = 200$, $K_p = 100^2$, $K_\omega = 10$, and $K_o = 200$.

2.7 Summary

A new framework for controlling modular robot manipulators has been presented in this chapter. Contrary to existing methods that unnecessarily force modular robots to require decentralized control schemes, we show both theoretically and with experiments that centralized control is possible instead. The approach that has been proposed to achieve this is systematic and allows the automatic obtainment of the robot description after arbitrary assembly of modules. Thanks to this approach, the large body of results proposed so far for controlling classical robots can also be used for modular reconfigurable robots. Thus, the criticism typically associated with modular robot manipulators regarding their difficult control problem is removed at the root. In fact, the experimental results lead to the conclusion that the motion-control performance of a modular robot manipulator can be made assembly independent with the proposed framework.

The proposed framework is complete since it covers everything from modeling the single modules to the automatic deployment of motion-control schemes in joint-space and task-space. In particular, it has been shown that minimal assumptions on the module geometries have been considered (basically only that they have standardized connectors). This aspect is particularly important because it makes the framework suitable for modular robot setups with heterogeneous modules, contrary to other existing approaches.

Besides the implementation of existing approaches for optimal exploitation of possible null-space motions or impedance/admittance control, this framework would certainly benefit from the development of automatically deployable kinematic control schemes with hard constraints on obstacle avoidance and joint limits. Another interesting extension of this framework could be the consideration of joint modules with multiple input and output connectors that would enable the realization of assemblies with kinematic loops.

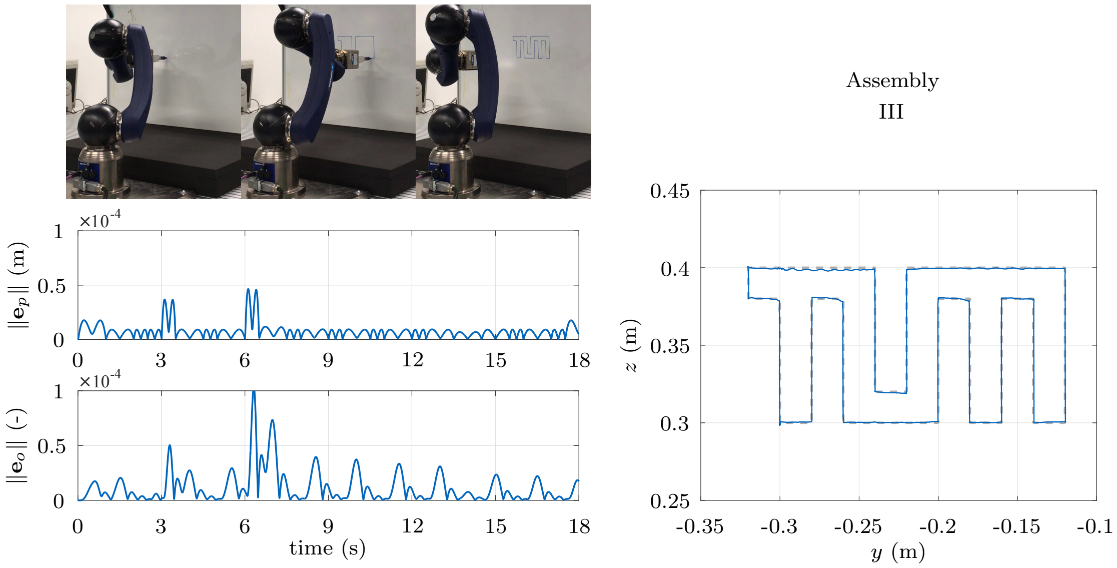


Figure 2.18: Task-space trajectory tracking experiment for assembly III.

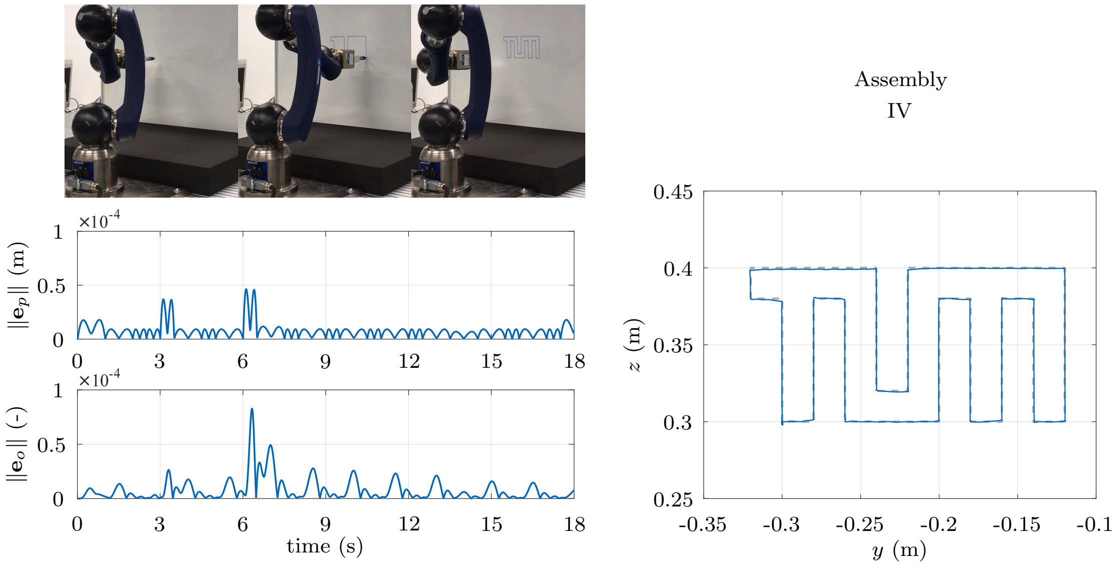


Figure 2.19: Task-space trajectory tracking experiment for assembly IV.

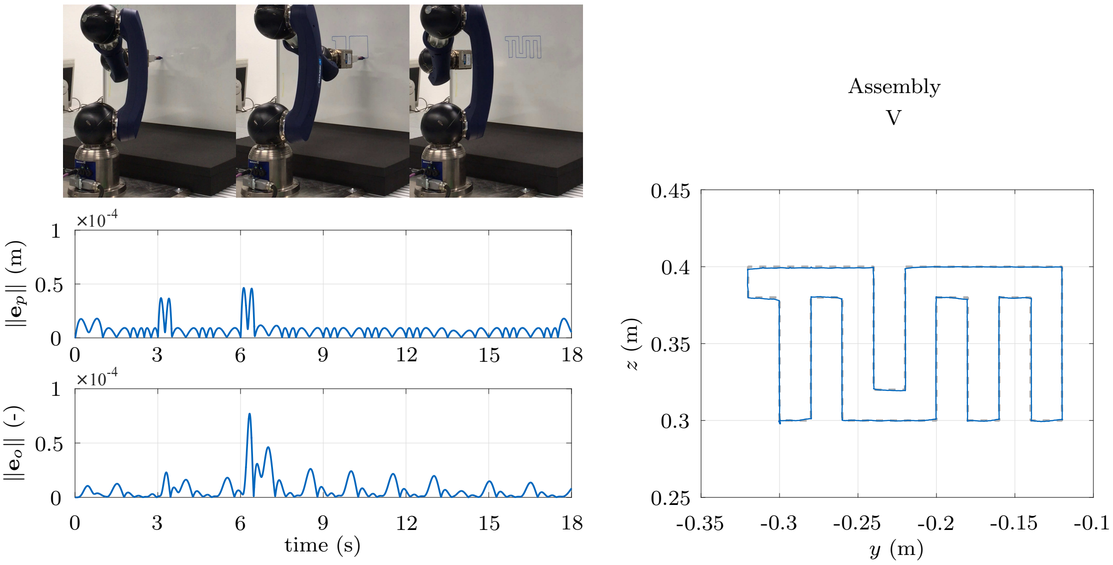


Figure 2.20: Task-space trajectory tracking experiment for assembly V.

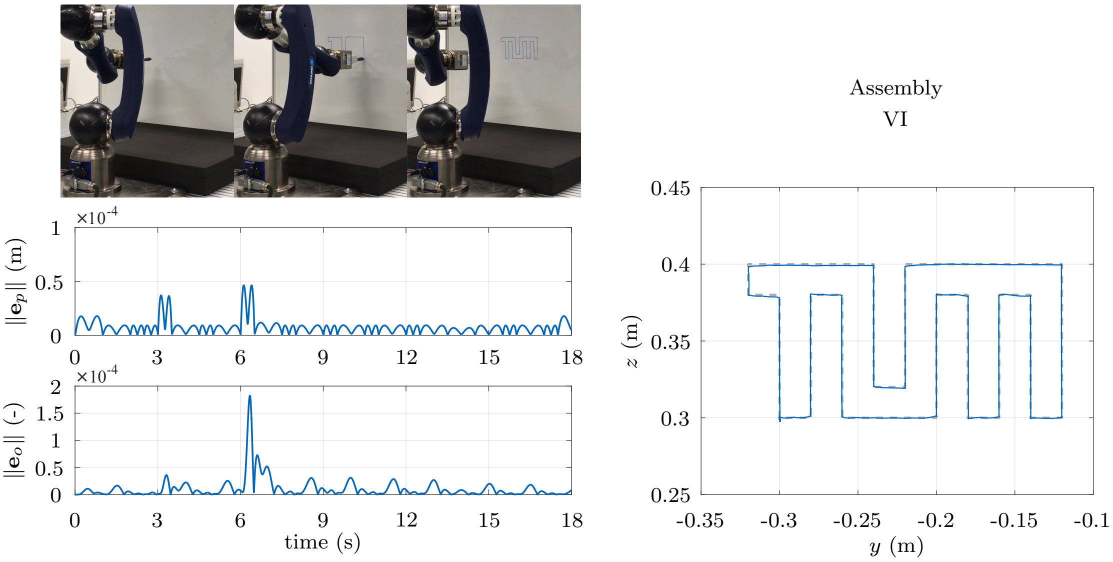


Figure 2.21: Task-space trajectory tracking experiment for assembly VI.

Chapter 3

Automatic Design of Robust Controllers

This chapter focuses on the design of controllers that can guarantee robust stability and performance under model mismatches and input disturbances. To this end, the state-of-the-art works that approach this problem are first reviewed, paying particular attention to their applicability for robust control of modular arms. Then, this chapter presents a new approach for robust control using interval arithmetic, which allows one to automatically deploy robust controllers. Simulation and experimental results which verify effectiveness and applicability of this approach are also presented. This chapter is largely based on, and extend, the author's work in [125,126].

3.1 Introduction and State of the Art

In light of the results presented in the previous chapter, the automatic deployment of model-based controllers for modular arms has shown to be a practically viable option. As is typical for model-based controllers in general, the achievable tracking performance strongly depends on the quality of the model available. For modular manipulators this is clearly related to the accuracy of the module data with respect to the real module parameters. In practice, the matching between the model that can be obtained from module data and the real arm dynamics is always imperfect. In particular, the mismatch can easily become significant when unknown payloads are involved. Modifications to the control laws (e.g., after performing system identification procedures or several tuning iterations), are a possible solution for classical fixed-configuration robots in order to achieve and maintain good motion-control performance. However, when considering modular robot arms, these procedures are time consuming and would clearly limit their swift reconfigurability.

The problem of designing controllers for robot arms with uncertain dynamics has attracted numerous researchers since the late '70s. A survey that provides an overview of early works on this subject can be found in [1], and classical approaches can be found in textbooks such as [40, 96, 102]. Among existing methods, the authors in [105] consider the use of a simplified model for feedback linearization and complete the control law with a robustifying linear compensator that ensures a bounded norm of the trajectory tracking error. This controller relies on the assumption that model mismatches can be bounded. A completely linear state feedback approach, where gains are selected on the base of initial conditions and a suitable polynomial function that bounds the perturbations on the closed-loop system, is proposed in [90] and ensures boundedness of the tracking error norm. Methods that exploit an optimal control framework for solving the robust control problem can be found e.g., in [63, 87].

3.1.1 Discontinuous, Smoothed, and Continuous Schemes

Robust control methods that can theoretically guarantee asymptotic convergence of the trajectory tracking error to zero are based on discontinuous control laws [30, 45]. The discontinuity makes these schemes difficult to implement in practice because they introduce undesired chattering behavior [97]. Chattering is usually a highly undesired effect because it may excite unmodeled dynamics, such as those due to the elasticity of the transmissions and the flexibility of the links e.g., when lightweight components are used. Classical robust control schemes have recently been revised in [10], where the model uncertainty has been represented in a way such that it avoids the influence of the controller gains. In practice, with classical approaches, increasing the gains would also increase the bounds on the perturbations from model uncertainty which would clearly make tuning more complicated. However, in the description of [10], the control law presented is discontinuous, thus exposing the closed-loop system to the previously-mentioned risk of chattering. Continuity of these control laws can be provided by smoothing the discontinuous robust control action as proposed e.g., in [30, 97]. However, this procedure has the consequence that asymptotic convergence to zero of the tracking error can no longer be ensured. For this aspect, the authors in [30] defined a practical stability notion of uncertain systems, by introducing the concept of uniform ultimate boundedness of the trajectories. This stability notion has subsequently become widely adopted.

Among the most effective methods that adopt the smoothing concept is the work in [104], which is also particularly easy to implement. In fact, by exploiting the property that the dynamical model of a manipulator can be made linear in the dynamical parameters (see e.g., [96, Section 7.2.2]), the perturbations due to model uncertainty can be directly considered by the dynamical parameters instead of considering bounds on a highly nonlinear state-dependent perturbation function. The effectivenenss of this approach has been experimentally evaluated in [54] on a directly-actuated planar arm with two degrees of freedom. In that paper, an experimental comparison is provided where another theoretically interesting controller is also tested, which is intrinsically continuous since it does not rely on smoothing a discontiuous control law. Such a controller is called a "r-α" tracker and has been proposed in [120]. The name r-α tracker comes from its capability of ensuring user defined bounds of the tracking error norm r, which can be reached with a convergence rate α. The approaches in [104] and [120] have unique benefits in principle: the former does not require the computation of bounds on nonlinear state-dependent perturbation terms, while the latter does. However, the latter can explicitly guarantee that a user-defined robust performance on the tracking error can be met. The r-α tracker may require very high control gains and introduce chattering when considering realistic sampling effects, as shown in the numerical simulations presented next and in [125]. It is important to mention the scheme in [18, Section 2.4.2] that is inspired by [104]. This controller exploits the property of linearity in the dynamical parameters and is intrinsically continuous without the need of resorting to the smoothing approach.

3.1.2 Removal of Implementation Difficulties

The estimation of bounds of perturbations, which many proposed robust control techniques require, is not easy to perform in practice. One possible practical shortcut is to consider these bounds as parameters to be adjusted during tuning procedures. This aspect is particularly critical for modular robots that can be frequently and arbitrarily reconfigured, since the change of the model due to reconfiguration may change the perturbations from model uncertainties as well. To address this issue, a novel approach for robust control that does not require the estimation of bounds of nonlinear state-dependent perturbation terms has been introduced in the author's work [125]. This paper proposes the idea of using interval arithmetic for automatic online computation of worst-case perturbations that has the additional benefit of providing guaranteed overapproximative results. Such an interval-arithmetic-based robust control approach, contrary to [104] and [18, Section 2.4.2], does not exploit the property of linearity in the robot model's parameters and therefore does not require the computation of the regressor matrix. Even though in [125] the computation of worst-case perturbation using interval arithmetic was still performed in a computationally expensive way, the author's subsequent results in [126] removed this issue. In fact, this last work proposes a method for highly efficient computability of the perturbations with interval arithmetic. This novel control approach is suitable for automatically deploying robust controllers for classical fixed configuration and for modular reconfigurable robot manipulators. It has also been recently applied with success for addressing the robust control problem of more exotic continuum robot arms, as presented in the author's contribution in [131]. After the formulation of the control problem and fundamental preliminaries, a complete description of the interval-arithmetic-based robust control approach follows in the next sections.

3.2 Formulation of the Control Problem

A robot arm assembled from modules with N serially connected links with uncertain dynamical parameters is considered. Hereafter, the real unknown dynamical parameters of the arm are considered to be collected in a vector as

$$\begin{aligned}\mathbf{\Delta} = (m_1\, \ldots,\, m_N,\, c_{x,1},\, \ldots,\, c_{z,N},\, I_{xx,1},\, \ldots, I_{zz,N},\, \beta_{v,1}\, \ldots,\, \beta_{v,N},\\ \beta_{c,1}\, \ldots,\, \beta_{c,N},\, I_{m,1}\, \ldots,\, I_{m,N})^T\,,\end{aligned}$$

where the mass, the coordinates of the center of mass, and the inertia tensor of the ith link are respectively denoted by

$$m_i,\;\; \mathbf{r}_{D_i,C_i} = \begin{pmatrix} c_{x,i} \\ c_{y,i} \\ c_{z,i} \end{pmatrix} \;\text{and}\; \mathbf{I}_i = \begin{pmatrix} I_{xx,i} & -I_{xy,i} & -I_{xz,i} \\ * & I_{yy,i} & -I_{yz,i} \\ * & * & I_{zz,i} \end{pmatrix}.$$

Further, $I_{m,i}$ is the moment of inertia of the rotor, $\beta_{v,i}$ the viscous friction coefficient, and $\beta_{c,i}$ the static one.

The dependency on the dynamical parameters of the assembled arm is now made explicit for the sake of clarity of the subsequent description. The mathematical model that describes the dynamics of the considered system can be rewritten as

$$\mathbf{M}(\mathbf{q},\mathbf{\Delta})\ddot{\mathbf{q}} + \overbrace{\mathbf{C}(\mathbf{q},\dot{\mathbf{q}},\mathbf{\Delta})\dot{\mathbf{q}} + \mathbf{f}(\dot{\mathbf{q}},\mathbf{\Delta}) + \mathbf{g}(\mathbf{q},\mathbf{\Delta})}^{:=\mathbf{n}(\mathbf{q},\dot{\mathbf{q}},\mathbf{\Delta})} = \mathbf{u} + \mathbf{d}, \tag{3.1}$$

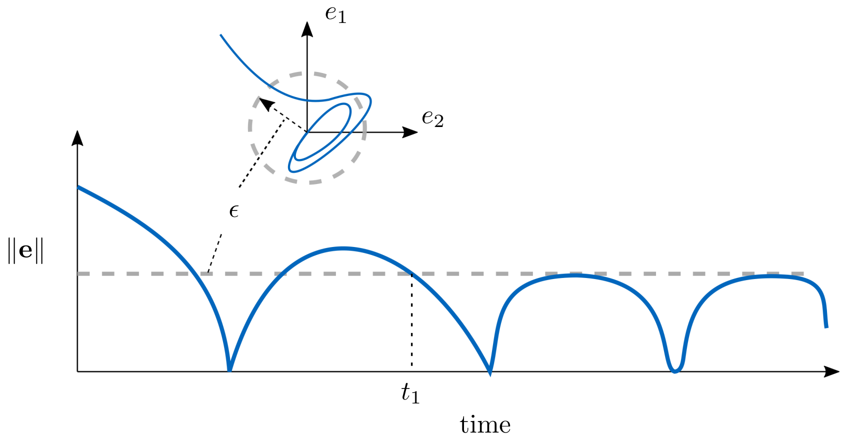


Figure 3.1: Illustration of error trajectories for a closed-loop system with ultimate robust performance/stability.

where $\mathbf{d} \in \mathbb{R}^N$ is a bounded input disturbance vector. For control design purposes, it is assumed that the model structure and a nominal guess $\mathbf{\Delta}_0$ of the real unknown vector $\mathbf{\Delta}$ are available. This is possible to assume considering the results of the framework presented in Chapter 2, which provides the automatic generation of a (nominal) model. In that case, the model structure is defined by the kinematics of the assembled robot arm, whose description can be obtained automatically by processing module data. A vector of dynamical parameters can also be obtained. Clearly, the outcome of this procedure can only provide some nominal values. An additional practical assumption is that the amount of uncertainty for each element of $\mathbf{\Delta}$ is known, so that certain bounds for each parameter can be considered. Finally, the input disturbance is assumed to have bounded norm β_d. Both the cases that β_d is known a priori and that it is unknown are considered.

This chapter focuses on the problem of ensuring that an automatically generated controller for a modular robot provides

$$\|\mathbf{e}\| < \epsilon, \quad \forall\, t \geq t_1, \tag{3.2}$$

for a finite time $t_1 \geq 0$ and a finite $\epsilon > 0$. We denote this result as ultimate robust performance if ϵ is defined by the user; otherwise, we simply refer to ultimate robust stability. An illustration of the targeted evolution of the tracking error over time is shown in Figure 3.1.

3.3 Preliminaries on Interval Arithmetic

Interval arithmetic [55, 75, 76] is a tool from applied mathematics that enables one to bound possible solutions of mathematical problems involving uncertain parameters, with absolute certainty. In fact, with interval arithmetic the operations are not limited to real numbers only, but instead can involve sets of real numbers. This is especially useful for analyzing physical systems whose parameters are uncertain. Software packages are currently available for performing computations with interval arithmetic e.g., in [4, 91].

The author's proposed approach for robust control in [125, 126] exploits the automatic computation of closed-loop perturbations from highly-nonlinear uncertain robot dynamics, using interval arithmetic. This control approach is detailed in the upcoming sections. To maintain clarity of the subsequent description, the following definitions are introduced.

Definition 1 (Multidimensional interval)**.** *A multidimensional interval is defined as a set of real numbers:*

$$[\mathbf{x}] := [\underline{\mathbf{x}}, \overline{\mathbf{x}}], \quad \underline{\mathbf{x}} \in \mathbb{R}^n, \ \ \overline{\mathbf{x}} \in \mathbb{R}^n, \ \ \underline{x}_i \leq \overline{x}_i, \ \ \forall \ \ i \in \{1, \ldots, n\}.$$

The scalar case is simply denoted by $[x]$ *instead of* $[\mathbf{x}]$*, with* $\underline{x}$ *for denoting its infimum and* $\overline{x}$ *its supremum.*

Definition 2 (Degenerate interval)**.** *A degenerate interval is an interval* $[x]$ *whose infimum* $\underline{x}$ *and supremum* $\overline{x}$ *are equal. This case is simply denoted by* x*.*

Definition 3 (Interval-valued function)**.** *An interval-valued function can be seen as an extension of a real-valued function evaluated with one or more interval arguments. More precisely, given a generic real-valued function* $\mathbf{z} : \mathbb{R}^n \rightarrow \mathbb{R}^m$*, its interval evaluation over a set* $[\mathbf{x}]$ *is defined as*

$$\mathbf{z}([\mathbf{x}]) := \{\mathbf{z}(\mathbf{x}) \ \mid \ \mathbf{x} \in [\mathbf{x}]\}.$$

The operations between intervals that are used in this thesis are now defined as set-based addition, subtraction, multiplication, and division as follows.

Definition 4 (Set-based operations)**.** *Let* $\mathbb{IR}$ *be the set of all scalar intervals. For* $[x] \in \mathbb{IR}$ *and* $[y] \in \mathbb{IR}$*, the result of the binary operations* $* \in \{+, -, \ \cdot \ , /\}$ *is defined as*

$$[x] \circledast [y] := \{x * y \ \mid \ x \in [x], y \in [y]\}.$$

The above-mentioned operations can be straightforwardly implemented as follows (see e.g., [76, Appendix B]):

$$
\begin{aligned}
[x] \oplus [y] &= [\underline{x} + \underline{y},\, \overline{x} + \overline{y}],\\
[x] \ominus [y] &= [\underline{x} - \overline{y},\, \overline{x} + \underline{y}],\\
[x] \odot [y] &= [\min(\underline{xy},\, \underline{x}\overline{y},\, \overline{x}\underline{y},\, \overline{xy}),\, \max(\underline{xy},\, \underline{x}\overline{y},\, \overline{x}\underline{y},\, \overline{xy})],\\
[x] \oslash [y] &= [x] \odot [iy], \text{ where } [iy] = [1/\overline{y},\, 1/\underline{y}], \quad 0 \notin [y].
\end{aligned}
$$

When multidimensional intervals are involved, set-based addition and subtraction are applied element-wise. Contrary, interval matrix/scalar-matrix multiplication requires a definition.

Definition 5 (Interval matrix/scalar-matrix multiplication)**.** *Given* $[\mathbf{X}] \in \mathbb{IR}^{n\times m}$*,* $[\mathbf{Y}] \in \mathbb{IR}^{m\times p}$ *and* $[a] \in \mathbb{IR}$*, the results of a matrix and scalar-matrix multiplication are defined respectively as*

$$
([\mathbf{X}] \odot [\mathbf{Y}])_{ij} = \bigoplus_{k=1}^{n} \Big([\mathbf{X}]_{ik} \odot [\mathbf{Y}]_{kj} \Big), \quad ([a] \odot [\mathbf{X}])_{ij} = [a] \odot [\mathbf{X}]_{ij},
$$

where $\bigoplus_{k=1}^{n}$ *denotes the interval version of the summation symbol involving set-based additions.*

Finally, the set-based cross-product is defined, as it will be required for robot dynamics computations. Please note that set-based multiplications and divisions bind more strongly than additions and subtractions.

Definition 6 (Set-based cross-product)**.** *Given two interval vectors* $[\mathbf{x}] \in \mathbb{IR}^{3\times 1}$ *and* $[\mathbf{y}] \in \mathbb{IR}^{3\times 1}$*, the result of the set-based cross product between them is defined as*

$$
[\mathbf{x}] \otimes [\mathbf{y}] = \begin{pmatrix} [\mathbf{x}]_2 \odot [\mathbf{y}]_3 \ominus [\mathbf{x}]_3 \odot [\mathbf{y}]_2 \\ [\mathbf{x}]_3 \odot [\mathbf{y}]_1 \ominus [\mathbf{x}]_1 \odot [\mathbf{y}]_3 \\ [\mathbf{x}]_1 \odot [\mathbf{y}]_2 \ominus [\mathbf{x}]_2 \odot [\mathbf{y}]_1 \end{pmatrix}.
$$

3.4 Interval-Arithmetic-Based Robust Control

The use of interval arithmetic for robust control of robot manipulators, as proposed in the author's work [125], has the benefit of removing the need for nontrivial, time-consuming, and often empiric procedures for estimating bounds of closed-loop perturbations arising from model uncertainties. The basic idea is that, starting from known uncertain bounds for the physical parameters of the arm-links (e.g., masses, inertia tensors, and coordinates of the centers of mass), the effect of the propagation of the parametric uncertainty through the nonlinear dynamical model (which creates the closed-loop perturbations) can be directly computed with interval arithmetic in the form of certain bounds. This has the clear advantage of making the robust control quickly deployable, with positive implications for the automatic generation of the robust control of modular manipulators.

In the following subsections, the theoretical derivation of Interval-Arithmetic-Based (IAB) robust controllers is first presented. Then, a method to enable the efficient (online) computation of IAB robust controllers is described, followed by a description of the application of this control approach for modular reconfigurable arms. The two closing sections of this chapter present numerical comparative simulations and experimental results that verify the approach's applicability.

3.4.1 Derivation of the Controllers

The problem of achieving robust stability and robust performance is addressed in this subsection using interval arithmetic. Even if some uncertainty in the knowledge of the dynamical parameters of the arm-links is present, model-based trajectory tracking controllers can be implemented by using nominal model data. Nominal model parameters are denoted hereafter with 0 as a subscript. A nominal vector of dynamical parameters Δ_0 can now be considered, which collects the nominal dynamical parameters of the arm-links as:

$$\begin{aligned}\Delta_0 = (&m_{01}\, \ldots,\, m_{0N},\, c_{x,01},\, \ldots,\, c_{z,0N},\, I_{xx,01},\, \ldots, I_{zz,0N},\, \beta_{v,01}\, \ldots,\, \beta_{v,0N},\\ &\qquad\qquad\qquad\qquad \beta_{c,01}\, \ldots,\, \beta_{c,0N},\, I_{m,01}\, \ldots,\, I_{m,0N})^T\,.\end{aligned}$$

Hereafter, bounds of uncertainty of the parameters for each ith arm-link are considered as available. Using these bounds, the interval vector of the uncertain dynamical parameters can be defined as

$$\begin{aligned}[\Delta] = ([&m_1]\, \ldots,\, [m_N],\, [c_{x,1}],\, \ldots,\, [c_{z,N}],\, [I_{xx,1}],\, \ldots, [I_{zz,N}],\, [\beta_{v,1}]\, \ldots,\, [\beta_{v,N}],\\ &\qquad\qquad\qquad\qquad [\beta_{c,1}]\, \ldots,\, [\beta_{c,N}],\, [I_{m,1}]\, \ldots,\, [I_{m,N}])^T\,,\end{aligned}$$

such that $\mathbf{\Delta}_0 \in [\mathbf{\Delta}]$.

Two different interval-arithmetic-based robust controllers are formulated in the following. The type depends on the nominal tracking control law employed. First, the case of using inverse-dynamics nominal control is described, followed by the case of using passivity-based nominal control. These two schemes will be denoted next by Interval-Arithmetic Inverse-Dynamics (IA-ID) control, and Interval-Arithmetic Passivity-Based (IA-PB) control, respectively. For compactness of the following description the model terms computed with nominal model parameters are also denoted with 0 as a subscript. For example, the nominal mass matrix $\mathbf{M}\left(\mathbf{q}, \mathbf{\Delta}_0\right)$ will be simply denoted by $\mathbf{M}_0\left(\mathbf{q}\right)$. The same applies for all other model terms.

Interval-Arithmetic Inverse-Dynamics Control

The inverse-dynamics control law with explicit dependence of the nominal parameters can be written as

$$\mathbf{u} = \mathbf{M}_0\left(\mathbf{q}\right)\overbrace{\left(\ddot{\mathbf{q}}_d + \mathbf{K}_D\dot{\mathbf{e}} + \mathbf{K}_P\mathbf{e}\right)}^{=\mathbf{y}} + \overbrace{\mathbf{C}_0\left(\mathbf{q}, \dot{\mathbf{q}}\right)\dot{\mathbf{q}} + \mathbf{f}_0\left(\dot{\mathbf{q}}\right) + \mathbf{g}_0\left(\mathbf{q}\right)}^{:=\mathbf{n}_0(\mathbf{q},\dot{\mathbf{q}})} - \boldsymbol{\nu}, \tag{3.3}$$

where $\mathbf{n}_0$ contains the nominal contributions due to centrifugal, Coriolis, friction, and gravity terms. Further, in (3.3) the vector $\mathbf{y}$ is the same as in (2.29), and $\boldsymbol{\nu}$ is an auxiliary input vector that allows the introduction of the interval-arithmetic-based robust control action. By applying this control law to (3.1), the system

$$\mathbf{M}(\mathbf{q}, \mathbf{\Delta})\ddot{\mathbf{q}} = \mathbf{M}_0(\mathbf{q})\mathbf{y} + \mathbf{n}_0(\mathbf{q}) - \mathbf{n}(\mathbf{q}, \mathbf{\Delta}) - \boldsymbol{\nu} + \mathbf{d} \tag{3.4}$$

is obtained. Now, the subtraction from both sides of (3.4) of the term $\mathbf{M}(\mathbf{q}, \mathbf{\Delta})(\ddot{\mathbf{q}}_d + \mathbf{K}_D\dot{\mathbf{e}} + \mathbf{K}_P\mathbf{e})$, yields

$$\mathbf{M}(\mathbf{q}, \mathbf{\Delta})(\ddot{\mathbf{e}} + \mathbf{K}_D\dot{\mathbf{e}} + \mathbf{K}_P\mathbf{e}) = \mathbf{w}_{ID}(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{y}, \mathbf{\Delta}_0, \mathbf{\Delta}, \mathbf{d}) + \boldsymbol{\nu}, \tag{3.5}$$

where

$$\mathbf{w}_{ID}(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{y}, \mathbf{\Delta}_0, \mathbf{\Delta}, \mathbf{d}) = \big(\mathbf{M}(\mathbf{q}, \mathbf{\Delta}) - \mathbf{M}_0(\mathbf{q})\big)\mathbf{y} + \mathbf{n}(\mathbf{q}, \mathbf{\Delta}) - \mathbf{n}_0(\mathbf{q}) - \mathbf{d} \tag{3.6}$$

is a perturbation term arising from imperfect knowledge of the dynamical model parameters and external disturbance.

From the position of (3.5), the interval-arithmetic-based robust control action can be introduced through $\boldsymbol{\nu}$. To this end, a function that always maximizes the worst-case perturbation can be defined as

$$\rho([\mathbf{\Phi}_{ID}]) = \max\big(|\underline{\mathbf{\Phi}}_{ID}|, |\overline{\mathbf{\Phi}}_{ID}|\big), \tag{3.7}$$

in which the max operator is applied element-wise and where

$$[\mathbf{\Phi}_{ID}] = \mathbf{w}_{ID}(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{y}, \mathbf{\Delta}_0, [\mathbf{d}], [\mathbf{\Delta}]),$$

from the interval-valued function $\mathbf{w}_{ID}(\dots, [\mathbf{d}], [\mathbf{\Delta}])$. By considering Definition 3, it is not difficult to see that

$$\mathbf{w}_{ID}(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{y}, \mathbf{\Delta}_0, \mathbf{\Delta}, \mathbf{d}) \in [\mathbf{\Phi}_{ID}] = \mathbf{w}_{ID}(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{y}, \mathbf{\Delta}_0, [\mathbf{\Delta}], [\mathbf{d}]),\ \ \forall\, \mathbf{q},\ \dot{\mathbf{q}},\ \mathbf{y},\ \mathbf{\Delta}_0 \text{ and } \mathbf{\Delta} \in [\mathbf{\Delta}],\ \mathbf{d} \in [\mathbf{d}]$$

holds, which provides

$$\rho_i([\mathbf{\Phi}_{ID}]) \geq |w_{ID,i}(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{y}, \mathbf{\Delta}_0, \mathbf{\Delta}, \mathbf{d})|, \;\; \forall\, \mathbf{q},\, \dot{\mathbf{q}},\, \mathbf{y},\, \mathbf{\Delta}_0 \text{ and } \mathbf{\Delta} \in [\mathbf{\Delta}],\, \mathbf{d} \in [\mathbf{d}]. \tag{3.8}$$

The result in (3.8) can be simply shown by contradiction. For example, suppose instead that $\rho_i([\mathbf{\Phi}_{ID}]) < |w_{ID,i}(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{y}, \mathbf{\Delta}_0, \mathbf{\Delta}, \mathbf{d})|$. Then, from (3.7)

$$\max\left(|\underline{\Phi}_{ID,i}|, |\overline{\Phi}_{ID,i}|\right) < |w_{ID,i}(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{y}, \mathbf{\Delta}_0, \mathbf{\Delta}, \mathbf{d})|$$

would also follow. However, this contradicts the definition of the interval-valued function that provides $[\mathbf{\Phi}_{ID}]$ and implies that

$$\underline{\Phi}_{ID,i} \leq w_{ID,i}(\ldots, \mathbf{\Delta}, \mathbf{d}) \leq \overline{\Phi}_{ID,i}. \tag{3.9}$$

The completion of the Interval-Arithmetic Inverse-Dynamics (IA-ID) control law follows by directly using $\rho([\mathbf{\Phi}_{ID}])$ for feedback control. To describe this step, the following closed-loop error-dynamics (which is derived from (3.5)) can be considered for

$$\boldsymbol{\xi} = (\mathbf{e}^T, \;\; \dot{\mathbf{e}}^T)^T, \tag{3.10}$$

$$\dot{\boldsymbol{\xi}} = \mathbf{A}\,\boldsymbol{\xi} + \mathbf{B}\mathbf{M}^{-1}\boldsymbol{\nu} + \mathbf{B}\mathbf{M}^{-1}\mathbf{w}, \tag{3.11}$$

where

$$\mathbf{A} = \begin{pmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{K}_P & -\mathbf{K}_D \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} \mathbf{0} \\ \mathbf{I} \end{pmatrix}. \tag{3.12}$$

From the position of (3.11), the task is now to exploit both the interval arithmetic tools introduced earlier and $\boldsymbol{\nu}$, for counteracting the perturbation term $\mathbf{w}$. The following derivation is inspired by [18, Section 2.4.2] and introduces the use of interval arithmetic for feedback control inspired by the author's work [125]. In the following, $\lambda_{\min}(\boldsymbol{\chi})$ and $\lambda_{\max}(\boldsymbol{\chi})$ are used to denote the minimum and the maximum eigenvalue of a matrix $\boldsymbol{\chi}$, respectively.

By considering a symmetric positive definite matrix $\mathbf{P}$ such that

$$\mathbf{A}^T\mathbf{P} + \mathbf{P}\mathbf{A} = -\mathbf{Q} \tag{3.13}$$

with $\mathbf{Q}$ positive definite, the following Lyapunov function candidate can be used:

$$\mathbf{V} = \boldsymbol{\xi}^T\mathbf{P}\,\boldsymbol{\xi}. \tag{3.14}$$

Taking the derivative over time of (3.14), along the trajectories of the system in (3.11), yields

$$\dot{\mathbf{V}} = -\,\boldsymbol{\xi}^T\mathbf{Q}\,\boldsymbol{\xi} + 2\,\boldsymbol{\xi}^T\mathbf{P}\mathbf{B}\mathbf{M}^{-1}\big(\boldsymbol{\nu} + \mathbf{w}\big) \tag{3.15}$$

$$\overset{(3.8)}{\leq} \; -\,\boldsymbol{\xi}^T\mathbf{Q}\,\boldsymbol{\xi} + 2\,\boldsymbol{\xi}^T\mathbf{P}\mathbf{B}\mathbf{M}^{-1}\boldsymbol{\nu} + 2\,\|\boldsymbol{\xi}^T\mathbf{P}\mathbf{B}\|\|\mathbf{M}^{-1}\|\|\boldsymbol{\rho}\big([\mathbf{\Phi}_{ID}]\big)\|. \tag{3.16}$$

By selecting the robustifying term of the controller as

$$\boldsymbol{\nu} = -\kappa_P \, \|\boldsymbol{\rho}\big([\boldsymbol{\Phi}_{ID}]\big)\|^2 \, \mathbf{B}^T \mathbf{P} \boldsymbol{\xi}, \tag{3.17}$$

where $\kappa_P > 0$ is a tuning parameter, (3.16) becomes

$$\dot{\mathbf{V}} \leq -\boldsymbol{\xi}^T \mathbf{Q} \, \boldsymbol{\xi} - 2 \, \kappa_P \, \|\boldsymbol{\rho}\big([\boldsymbol{\Phi}_{ID}]\big)\|^2 \, \boldsymbol{\xi}^T \mathbf{P} \mathbf{B} \mathbf{M}^{-1} \mathbf{B}^T \mathbf{P} \boldsymbol{\xi} + 2 \, \|\boldsymbol{\xi}^T \mathbf{P} \mathbf{B}\| \|\mathbf{M}^{-1}\| \|\boldsymbol{\rho}\big([\boldsymbol{\Phi}_{ID}]\big)\|. \tag{3.18}$$

Now, by considering that

$$\lambda_{\min}\big(\mathbf{M}^{-1}\big) \|\mathbf{x}\|^2 \leq \mathbf{x}^T \mathbf{M}^{-1} \mathbf{x} \quad \forall \, \mathbf{x} \in \mathbb{R}^N,$$

$$\lambda_{\min}\big(\mathbf{M}^{-1}\big) = \frac{1}{\lambda_{\max}\big(\mathbf{M}\big)} \geq \frac{1}{\lambda_M} \quad \forall \, \mathbf{q} \in \mathbb{R}^N,$$

and that

$$\|\mathbf{M}^{-1}\| = \frac{1}{\lambda_{\min}\big(\mathbf{M}\big)} \leq \frac{1}{\lambda_m} \quad \forall \, \mathbf{q} \in \mathbb{R}^N,$$

for some finite positive constants λ_m, λ_M, the inequality (3.18) can be rewritten as

$$\begin{aligned} \dot{\mathbf{V}} \leq & -\boldsymbol{\xi}^T \mathbf{Q} \, \boldsymbol{\xi} - \frac{2\kappa_P}{\lambda_M} \|\boldsymbol{\xi}^T \mathbf{P} \mathbf{B}\|^2 \|\boldsymbol{\rho}\big([\boldsymbol{\Phi}_{ID}]\big)\|^2 + \frac{2}{\lambda_m} \|\boldsymbol{\xi}^T \mathbf{P} \mathbf{B}\| \|\boldsymbol{\rho}\big([\boldsymbol{\Phi}_{ID}]\big)\| \\ \leq & -\boldsymbol{\xi}^T \mathbf{Q} \, \boldsymbol{\xi} + \frac{2\kappa_P}{\lambda_M} \, \|\boldsymbol{\xi}^T \mathbf{P} \mathbf{B}\| \|\boldsymbol{\rho}\big([\boldsymbol{\Phi}_{ID}]\big)\| \Big(\frac{\lambda_M}{\kappa_P \lambda_m} - \|\boldsymbol{\xi}^T \mathbf{P} \mathbf{B}\| \|\boldsymbol{\rho}\big([\boldsymbol{\Phi}_{ID}]\big)\| \Big). \end{aligned} \tag{3.19}$$

It is now possible to observe from (3.19) that when $\|\boldsymbol{\xi}^T \mathbf{P} \mathbf{B}\| \|\boldsymbol{\rho}\big([\boldsymbol{\Phi}_{ID}]\big)\| \geq \frac{\lambda_M}{\kappa_P \lambda_m}$, $\dot{\mathbf{V}} < -\boldsymbol{\xi}^T \mathbf{Q} \, \boldsymbol{\xi}$, while for $\|\boldsymbol{\xi}^T \mathbf{P} \mathbf{B}\| \|\boldsymbol{\rho}\big([\boldsymbol{\Phi}_{ID}]\big)\| < \frac{\lambda_M}{\kappa_P \lambda_m}$ the following inequalities are obtained:

$$\begin{aligned} \dot{\mathbf{V}} \leq & -\boldsymbol{\xi}^T \mathbf{Q} \, \boldsymbol{\xi} + \frac{2}{\lambda_m} \, \|\boldsymbol{\xi}^T \mathbf{P} \mathbf{B}\| \|\boldsymbol{\rho}\big([\boldsymbol{\Phi}_{ID}]\big)\| \\ \leq & -\lambda_{\min}(\mathbf{Q}) \|\boldsymbol{\xi}\|^2 + \frac{2\lambda_M}{\kappa_P \lambda_m^2}. \end{aligned} \tag{3.20}$$

Since (3.20) always holds, uniform ultimate boundedness of the error trajectories follows. Further, the bound that is ultimately reached can be computed as

$$\|\boldsymbol{\xi}\| \leq \sqrt{\frac{\lambda_{\max}(\mathbf{P})}{\lambda_{\min}(\mathbf{P})} \frac{2\lambda_M}{\kappa_P \lambda_m^2}}. \tag{3.21}$$

Both these results can be shown *mutatis mutandis* from e.g., [60, Theorem 4.18].

In principle, from the relation (3.21) the designer can infer the minimum value of κ_P required to achieve a specific user defined tracking performances, provided that λ_M and λ_m are known. If these values are not known a priori, specific desired tracking performance can be reached by tuning κ_P. The block diagram of the complete control scheme is presented in Figure 3.2.

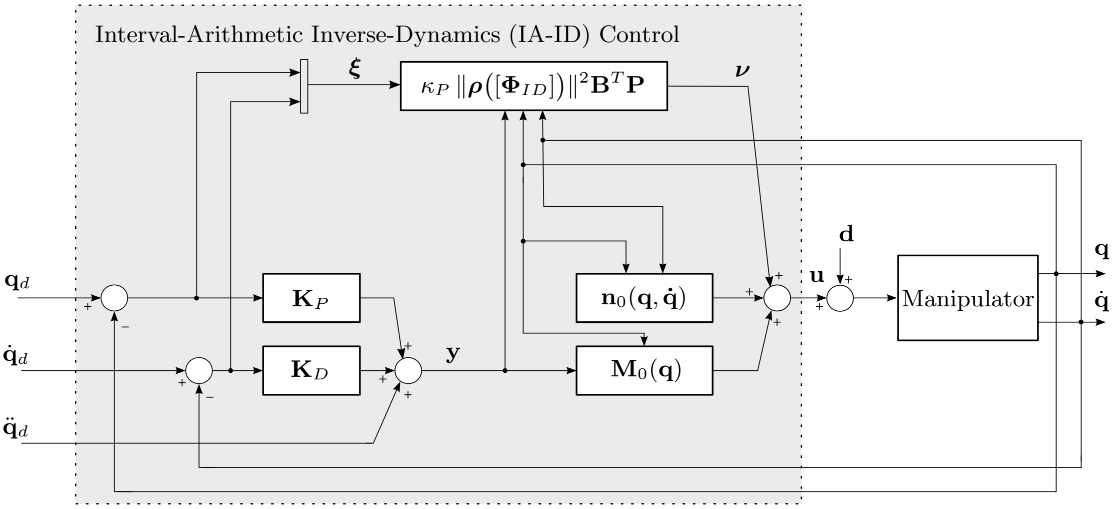

Interval-Arithmetic Inverse-Dynamics (IA-ID) Control
$\boldsymbol{\xi}$
$\kappa_P \, \|\boldsymbol{\rho}([\boldsymbol{\Phi}_{ID}])\|^2 \mathbf{B}^T\mathbf{P}$
$\boldsymbol{\nu}$
$\mathbf{q}_d$
$\dot{\mathbf{q}}_d$
$\ddot{\mathbf{q}}_d$
$\mathbf{K}_P$
$\mathbf{K}_D$
$\mathbf{y}$
$\mathbf{n}_0(\mathbf{q},\dot{\mathbf{q}})$
$\mathbf{M}_0(\mathbf{q})$
$\mathbf{u}$
$\mathbf{d}$
Manipulator
$\mathbf{q}$
$\dot{\mathbf{q}}$
+
−

Figure 3.2: Block diagram of Interval-Arithmetic Inverse-Dynamics (IA-ID) control.

An important remark on (3.17) is that it is continuous, and it does not approach a discontinuous control action when the required tracking performances (and thus κ_P in this case) is increased, contrary to e.g., [104]. Continuity follows from the fact that (3.8) is continuous, as the same argument of [125, Proposition 1] applies here. In particular, given that $[\boldsymbol{\Phi}_{ID}] = \mathbf{w}(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{y}, \boldsymbol{\Delta}_0, [\boldsymbol{\Delta}], [\mathbf{d}])$, for given $\mathbf{q}$, $\dot{\mathbf{q}}$, $\mathbf{y}$, one can choose $\mathbf{d}^* \in [\mathbf{d}]$ and $\boldsymbol{\Delta}^* \in [\boldsymbol{\Delta}]$ such that $\overline{\boldsymbol{\Phi}}_{ID}$ is maximal (or $\underline{\boldsymbol{\Phi}}_{ID}$ is minimal). Since $\mathbf{w}(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{y}, \boldsymbol{\Delta}_0, \boldsymbol{\Delta}^*, \mathbf{d}^*)$ is continuous, it guarantees in turn continuity of $|\overline{\boldsymbol{\Phi}}_{ID}|$ (or $|\underline{\boldsymbol{\Phi}}_{ID}|$). Then, since the max operator between two continuous functions preserves continuity, $\boldsymbol{\rho}([\boldsymbol{\Phi}_{ID}])$ is continuous.

Interval-Arithmetic Passivity-Based Control

An alternative interval-arithmetic-based robust controller can be obtained using the nominal passivity-based tracking scheme. This control law is first derived for the case of known bounds of the external disturbance vector $\mathbf{d}$. Subsequently, this assumption is relaxed. The following control scheme is employed:

$$\mathbf{u} = \mathbf{M}_0(\mathbf{q})\ddot{\mathbf{q}}_a + \mathbf{C}_0(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}_a + \mathbf{f}_0(\dot{\mathbf{q}}) + \mathbf{g}_0(\mathbf{q}) - \boldsymbol{\nu}, \tag{3.22}$$

in which the term $\boldsymbol{\nu} \in \mathbb{R}^N$ is exploited to enhance robustness. For the sake of clarity, it is recalled that

$$\dot{\mathbf{q}}_a = \dot{\mathbf{q}}_d + \mathbf{K}_r\mathbf{e}, \quad \mathbf{e} = \mathbf{q}_d - \mathbf{q}, \tag{3.23}$$

with $\mathbf{K}_r$ being a diagonal positive definite matrix of proper dimensions.

Applying the control law of (3.22) to (3.1) yields

$$\mathbf{M}(\mathbf{q},\Delta)\dot{\mathbf{r}}+\mathbf{C}(\mathbf{q},\dot{\mathbf{q}},\Delta)\mathbf{r}=\boldsymbol{\nu}+\mathbf{w}_{PB}(\mathbf{q},\dot{\mathbf{q}},\dot{\mathbf{q}}_a,\ddot{\mathbf{q}}_a,\boldsymbol{\Delta}_0,\boldsymbol{\Delta},\mathbf{d}), \tag{3.24}$$

where

$$\mathbf{r}=\dot{\mathbf{e}}+\mathbf{K}_r\mathbf{e}, \tag{3.25}$$

and $\mathbf{w}_{PB}(\mathbf{q},\dot{\mathbf{q}},\dot{\mathbf{q}}_a,\ddot{\mathbf{q}}_a,\boldsymbol{\Delta}_0,\boldsymbol{\Delta},\mathbf{d})$ is a disturbance vector due the external disturbance input and the imperfect knowledge of the system dynamics. This term can be expressed as

$$\mathbf{w}_{PB}(\mathbf{q},\dot{\mathbf{q}},\dot{\mathbf{q}}_a,\ddot{\mathbf{q}}_a,\boldsymbol{\Delta}_0,\boldsymbol{\Delta},\mathbf{d})=\tilde{\mathbf{M}}(\mathbf{q},\Delta)\ddot{\mathbf{q}}_a+\tilde{\mathbf{C}}(\mathbf{q},\dot{\mathbf{q}},\Delta)\dot{\mathbf{q}}_a+\tilde{\mathbf{g}}(\mathbf{q},\Delta)+\tilde{\mathbf{f}}(\dot{\mathbf{q}},\Delta)-\mathbf{d}, \tag{3.26}$$

where

$$\tilde{\mathbf{M}}(\mathbf{q},\Delta)=\mathbf{M}(\mathbf{q},\Delta)-\mathbf{M}_0(\mathbf{q}),\ \tilde{\mathbf{C}}(\mathbf{q},\dot{\mathbf{q}},\Delta)=\mathbf{C}(\mathbf{q},\dot{\mathbf{q}},\Delta)-\mathbf{C}_0(\mathbf{q},\dot{\mathbf{q}}),$$
$$\tilde{\mathbf{g}}(\mathbf{q},\Delta)=\mathbf{g}(\mathbf{q},\Delta)-\mathbf{g}_0(\mathbf{q}),\ \tilde{\mathbf{f}}(\dot{\mathbf{q}},\Delta)=\mathbf{f}(\mathbf{q},\Delta)-\mathbf{f}_0(\dot{\mathbf{q}}).$$

By proceeding in a similar way as for the inverse-dynamics version, the following relation can be written:

$$\mathbf{w}_{PB}(\mathbf{q},\dot{\mathbf{q}},\dot{\mathbf{q}}_a,\ddot{\mathbf{q}}_a,\boldsymbol{\Delta}_0,\boldsymbol{\Delta},\mathbf{d})\in[\boldsymbol{\Phi}_{PB}]=\mathbf{w}_{PB}(\mathbf{q},\dot{\mathbf{q}},\dot{\mathbf{q}}_a,\ddot{\mathbf{q}}_a,\boldsymbol{\Delta}_0,[\boldsymbol{\Delta}],[\mathbf{d}]), \tag{3.27}$$

where the set membership relation of (3.27) can be straightforwardly inferred from Definition 3. The worst-case perturbation is then measured with $[\boldsymbol{\Phi}_{PB}]$ as

$$\boldsymbol{\rho}([\boldsymbol{\Phi}_{PB}])=\max\left(|\underline{\boldsymbol{\Phi}}_{PB}|,|\overline{\boldsymbol{\Phi}}_{PB}|\right). \tag{3.28}$$

From the same arguments as for the inverse-dynamics variant, it holds that

$$\rho_i([\boldsymbol{\Phi}_{PB}])\geq|w_{PB,i}(\mathbf{q},\dot{\mathbf{q}},\dot{\mathbf{q}}_a,\ddot{\mathbf{q}}_a,\boldsymbol{\Delta}_0,\boldsymbol{\Delta},\mathbf{d})|,\ \ \forall\,\mathbf{q},\ \dot{\mathbf{q}},\ \dot{\mathbf{q}}_a,\ \ddot{\mathbf{q}}_a,\ \boldsymbol{\Delta}_0 \text{ and } \boldsymbol{\Delta}\in[\boldsymbol{\Delta}],\ \mathbf{d}\in[\mathbf{d}]. \tag{3.29}$$

In this case as well, $\boldsymbol{\rho}([\boldsymbol{\Phi}_{PB}])$ can be directly used for feedback control. To formulate the derivation of such a control law, it is convenient to start from the storage function

$$V(\mathbf{r})=\frac{1}{2}\mathbf{r}^T\mathbf{M}(\mathbf{q},\Delta)\mathbf{r}, \tag{3.30}$$

whose derivative along the coordinates of (3.24) can be written as

$$\begin{aligned}\dot{V}(\mathbf{r})\ &=\mathbf{r}^T\mathbf{M}(\mathbf{q},\Delta)\dot{\mathbf{r}}+\frac{1}{2}\mathbf{r}^T\dot{\mathbf{M}}(\mathbf{q},\Delta)\mathbf{r}\\ &\overset{(3.24)}{=}\mathbf{r}^T\Big(\boldsymbol{\nu}+\mathbf{w}_{PB}(\mathbf{q},\dot{\mathbf{q}},\dot{\mathbf{q}}_a,\ddot{\mathbf{q}}_a,\boldsymbol{\Delta}_0,\boldsymbol{\Delta},\mathbf{d})\Big)+\frac{1}{2}\mathbf{r}^T\Big(\dot{\mathbf{M}}(\mathbf{q},\Delta)-2\mathbf{C}(\mathbf{q},\dot{\mathbf{q}},\Delta)\Big)\mathbf{r}\\ &\overset{(2.2)}{=}\mathbf{r}^T\boldsymbol{\nu}+\mathbf{r}^T\mathbf{w}_{PB}(\mathbf{q},\dot{\mathbf{q}},\dot{\mathbf{q}}_a,\ddot{\mathbf{q}}_a,\boldsymbol{\Delta}_0,\boldsymbol{\Delta},\mathbf{d}).\end{aligned} \tag{3.31}$$

From this relation, the term $\boldsymbol{\nu}$ can now be selected as

$$\boldsymbol{\nu} = -\Big(\kappa(t)\,\|\boldsymbol{\rho}\big([\boldsymbol{\Phi}_{PB}]\big)\| + \varphi(t)\Big)\,\mathbf{r}. \tag{3.32}$$

where $\kappa(t)$ and $\varphi(t)$ are two positive increasing functions with $\kappa_P \geq 1$ and $\varphi_P \geq 1$ as their respective minimum. This choice in (3.31) yields

$$\begin{aligned}\dot V(\mathbf{r}) \; &= -\varphi(t)\,\|\mathbf{r}\|^2 - \kappa(t)\,\|\boldsymbol{\rho}\big([\boldsymbol{\Phi}_{PB}]\big)\|\,\|\mathbf{r}\|^2 + \mathbf{r}^T\mathbf{w}(\mathbf{q},\dot{\mathbf{q}},\dot{\mathbf{q}}_a,\ddot{\mathbf{q}}_a,\boldsymbol{\Delta}_0,\boldsymbol{\Delta},\mathbf{d})\\ &\leq -\varphi_P\,\|\mathbf{r}\|^2 - \kappa_P\,\|\boldsymbol{\rho}\big([\boldsymbol{\Phi}_{PB}]\big)\|\,\|\mathbf{r}\|^2 + \|\mathbf{r}\|\|\mathbf{w}(\mathbf{q},\dot{\mathbf{q}},\dot{\mathbf{q}}_a,\ddot{\mathbf{q}}_a,\boldsymbol{\Delta}_0,\boldsymbol{\Delta},\mathbf{d})\|\\ &\overset{(3.29)}{\leq} -\varphi_P\,\|\mathbf{r}\|^2 \underbrace{-\kappa_P\|\boldsymbol{\rho}\big([\boldsymbol{\Phi}_{PB}]\big)\|\,\|\mathbf{r}\|^2 + \|\mathbf{r}\|\|\boldsymbol{\rho}\big([\boldsymbol{\Phi}_{PB}]\big)\|}_{=:\,h_1(\mathbf{r})}.\end{aligned} \tag{3.33}$$

By factoring out $\|\mathbf{r}\|$ in $h_1(\mathbf{r})$, it follows that for $\|\mathbf{r}\| \geq \frac{1}{\kappa_P}$, $\dot V(\mathbf{r}) < 0$ since $h_1(\mathbf{r}) \leq 0$. From this result, uniform ultimate boundedness of the error trajectories is proven in a similar way as in [40, Section 2.4.2]. To show this, without loss of generality, one can consider that the state lies outside a ball $\mathcal{B}_{\kappa_P^{-1}}$ of radius $\frac{1}{\kappa_P}$, at $t = 0$. Subsequently, since $\dot V(\mathbf{r}) < 0$, the trajectories will converge to the ball $\mathcal{B}_{\kappa_P^{-1}}$ and there will be a finite time t_1 such that $\|\mathbf{r}(t_1)\| = \frac{1}{\kappa_P}$. Until the trajectories reach the edge of the ball, $h_1(\mathbf{r}) \leq 0$ and thus $\dot V(\mathbf{r}) \leq -\varphi_P\,\|\mathbf{r}\|^2$ from (3.33). Consequently,

$$V(\mathbf{r}(t_1)) - V(\mathbf{r}(0)) \leq \int_0^{t_1} -\frac{\varphi_P}{\kappa_P^2}\,dt = -t_1\,\frac{\varphi_P}{\kappa_P^2}. \tag{3.34}$$

Now, given that the norm of the inertia matrix always has upper and lower bounds we can write

$$\lambda_m\,\|\mathbf{x}\|^2 \leq \mathbf{x}^T\,\mathbf{M}(\mathbf{q},\Delta)\mathbf{x} \leq \lambda_M\,\|\mathbf{x}\|^2,\;\; \forall \mathbf{x} \in \mathbb{R}^N,$$

where $\lambda_m = \lambda_{min}(\mathbf{M}(\mathbf{q},\Delta)) > 0$ and $\lambda_M = \lambda_{max}(\mathbf{M}(\mathbf{q},\Delta)) < \;\infty$ represent the minimum and maximum eigenvalue of the matrix $\mathbf{M}(\mathbf{q},\Delta)$ $\forall$ $\mathbf{q}$, respectively. Recalling (3.30), it can now be shown that t_1 is finite. Given

$$\forall t:\;\; \gamma_1(\mathbf{r}) \leq V(\mathbf{r}) \leq \gamma_2(\mathbf{r}), \tag{3.35}$$

where $\gamma_1(\mathbf{r}) = \frac{1}{2}\,\lambda_m\,\|\mathbf{r}\|^2$ and $\gamma_2(\mathbf{r}) = \frac{1}{2}\,\lambda_M\,\|\mathbf{r}\|^2$. Since $V(\mathbf{r}(0)) \leq \gamma_2(\mathbf{r}(0))$ and $V(\mathbf{r}(t_1)) \geq \gamma_1(\mathbf{r}(t_1))$, using (3.34) yields

$$\gamma_1(\kappa_P^{-1}) \leq V(\mathbf{r}(t_1)) \leq \gamma_2(\mathbf{r}(0)) - t_1\,\frac{\varphi_P}{\kappa_P^2},$$

Therefore, t_1 is finite, given that κ_P is finite, as

$$t_1 \leq \frac{1}{2}\,\frac{\lambda_M\,\kappa_P^2\,\|\mathbf{r}(0)\|^2 - \lambda_m}{\varphi_P}.$$

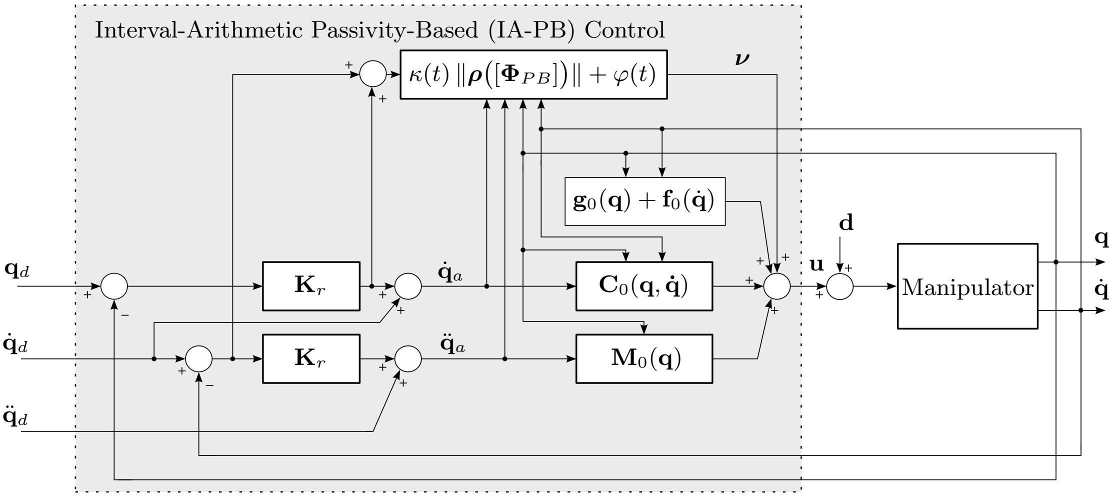

Interval-Arithmetic Passivity-Based (IA-PB) Control
$\kappa(t)\,\|\boldsymbol{\rho}([\boldsymbol{\Phi}_{PB}])\| + \varphi(t)$
$\boldsymbol{\nu}$
$\mathbf{g}_0(\mathbf{q}) + \mathbf{f}_0(\dot{\mathbf{q}})$
$\mathbf{q}_d$
$\mathbf{K}_r$
$\dot{\mathbf{q}}_a$
$\mathbf{C}_0(\mathbf{q}, \dot{\mathbf{q}})$
$\mathbf{u}$
$\mathbf{d}$
Manipulator
$\mathbf{q}$
$\dot{\mathbf{q}}$
$\dot{\mathbf{q}}_d$
$\mathbf{K}_r$
$\ddot{\mathbf{q}}_a$
$\mathbf{M}_0(\mathbf{q})$
$\ddot{\mathbf{q}}_d$

Figure 3.3: Block diagram of Interval-Arithmetic Passivity-Based (IA-PB) control.

Once the trajectories enter the ball $\mathcal{B}_{\kappa_P^{-1}}$, they could leave it at a finite time t_2, since there is no guarantee that $\dot{V}(\mathbf{r}) < 0$. Assuming now that they do leave the ball, the same reasoning of the time frame $0 \leq t \leq t_1$ applies for $t > t_2$ and therefore there will be a finite t_3 at which trajectories re-enter $\mathcal{B}_{\kappa_P^{-1}}$. Considering (3.35) and the time interval for $t_2 < t \leq t_3$,

$$\frac{1}{2}\,\lambda_m \,\|\mathbf{r}\|^2 \leq V(\mathbf{r}(t)) < V(\mathbf{r}(t_2)) \leq \frac{1}{2}\,\frac{\lambda_M}{\kappa_P^2},$$

which leads to the conclusion that the trajectories $\mathbf{r}$ are ultimately bounded by

$$\|\mathbf{r}\| \leq \frac{1}{\kappa_P}\sqrt{\frac{\lambda_M}{\lambda_m}}.$$

It is now important to highlight that the boundedness of the trajectories $\mathbf{r}$ imply boundedness of the tracking error trajectories $\mathbf{e}$ as well, with $\mathbf{K}_r$ properly chosen. This can be seen by considering the system in (3.25), with $\mathbf{r}$ as its bounded input. In fact, with $\mathbf{K}_r$ being diagonal and positive definite, this system is a set of first-order linear systems that asymptotically reach $|e_i| \leq \frac{|r_i|}{K_{r,i}}$ for each coordinate i. It is worth mentioning that the overall controller can ultimately reach any desired tracking performance with the selection of large enough gains of the matrix $\mathbf{K}_r$ and κ_P. In addition, ultimate robust performance can be achieved analytically once λ_M and λ_m are known, as it is the case for the inverse-dynamics variant. With respect to the inverse-dynamics version, however, tuning is simpler in this case since there is no need for computing the matrix $\mathbf{P}$. The block diagram of this control law is presented in Figure 3.3.

Contrary to the robust controllers based on [30] (such as [104]), in this approach, the tracking performance can be increased without approaching a discontinuous feedback control law by increasing φ_P, κ_P and/or the gains of $\mathbf{K}_r$. Increasing these gains, the tracking error can be made arbitrarily close to zero in principle. However, in practice there will be an upper limit of these gains due to the finite sampling rate of real applications and measurement noise.

From the above description, it follows that robust controllers which provide ultimate uniform boundedness of the error trajectories can be automatically deployed thanks to the use of interval arithmetic. Very practical tuning knobs for directly increasing the tracking performance have also been given. Now, by properly selecting the functions $\kappa(t)$ and $\varphi(t)$, a controller which ultimately guarantees robust performance can be devised. In particular, this is possible without the need for finding specific large enough gains of $\mathbf{K}_r$ and the knowledge of λ_M, λ_m. Further, the assumption of the knowledge of the bound on the norm of the external disturbance vector β_d is now relaxed.

Assuming that the bound of the external disturbance vector norm is not known, it cannot be included in $\rho([\mathbf{\Phi}_{PB}])$. In this case, for computing $\rho([\mathbf{\Phi}_{PB}])$ based on (3.27) one would use $[\Phi_{PB}] = \mathbf{w}_{PB}(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_a, \ddot{\mathbf{q}}_a, \mathbf{\Delta}_0, [\mathbf{\Delta}], \mathbf{0})$. Therefore, the same argument used previously for ultimate boundedness of the trajectories $\mathbf{r}$ does not directly hold. Considering (3.31) and noticing that $\mathbf{w}_{PB}(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_a, \ddot{\mathbf{q}}_a, \mathbf{\Delta}_0, \mathbf{\Delta}, \mathbf{d}) = \mathbf{w}_{PB}(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_a, \ddot{\mathbf{q}}_a, \mathbf{\Delta}_0, \mathbf{\Delta}, \mathbf{0}) - \mathbf{d}$, the derivative of (3.30) can now be written in a slightly different form:

$$\begin{aligned}
\dot{V}(\mathbf{r}) &= -\varphi(t)\,\|\mathbf{r}\|^2 - \kappa(t)\,\|\boldsymbol{\rho}\big([\mathbf{\Phi}_{PB}]\big)\|\,\|\mathbf{r}\|^2 + \mathbf{r}^T\mathbf{w}_{PB}(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_a, \ddot{\mathbf{q}}_a, \mathbf{\Delta}_0, \mathbf{\Delta}, \mathbf{0}) - \mathbf{r}^T\,\mathbf{d} \\
&\le -\varphi(t)\,\|\mathbf{r}\|^2 - \kappa(t)\,\|\boldsymbol{\rho}\big([\mathbf{\Phi}_{PB}]\big)\|\,\|\mathbf{r}\|^2 + \|\mathbf{r}\|\|\mathbf{w}_{PB}(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_a, \ddot{\mathbf{q}}_a, \mathbf{\Delta}_0, \mathbf{\Delta}, \mathbf{0})\| + \|\mathbf{r}\|\beta_d \\
&\overset{(3.29)}{\le} -\varphi(t)\,(1-\delta)\,\|\mathbf{r}\|^2 \underbrace{-\Big(\varphi(t)\,\delta + \kappa(t)\|\boldsymbol{\rho}\big([\mathbf{\Phi}_{PB}]\big)\|\Big)\|\mathbf{r}\|^2 + \Big(\|\boldsymbol{\rho}\big([\mathbf{\Phi}_{PB}]\big)\| + \beta_d\Big)\,\|\mathbf{r}\|}_{=:\,h_2(\mathbf{r})},
\end{aligned} \tag{3.36}$$

where δ is a scalar such that $0 < \delta < 1$, which has been introduced for simplifying the subsequent description. By factoring out $\|\mathbf{r}\|$ in $h_2(\mathbf{r})$, it is not difficult to see that $h_2(\mathbf{r}) \le 0$ for

$$\|\mathbf{r}\| \ge \frac{\|\boldsymbol{\rho}\big([\mathbf{\Phi}_{PB}]\big)\| + \beta_d}{\varphi(t)\,\delta + \kappa(t)\|\boldsymbol{\rho}\big([\mathbf{\Phi}_{PB}]\big)\|}. \tag{3.37}$$

From the right-hand side of (3.37), the following inequalities can be written:

$$\forall t: \;\; \frac{\|\boldsymbol{\rho}\big([\mathbf{\Phi}_{PB}]\big)\| + \beta_d}{\varphi(t)\,\delta + \kappa(t)\|\boldsymbol{\rho}\big([\mathbf{\Phi}_{PB}]\big)\|} \le \max\Big(\frac{\beta_d}{\varphi(t)\,\delta}, \frac{1}{\kappa(t)}\Big) \le \max\Big(\frac{\beta_d}{\varphi_P\,\delta}, \frac{1}{\kappa_P}\Big).$$

Assuming that β_d is finite, it is now possible to claim that ultimate uniform boundedness of the trajectories $\mathbf{r}$ follows from the same arguments used previously, when β_d is known. In particular, following that same approach and provided that a ball of radius $\max\left(\frac{\beta_d}{\varphi_P\,\delta}, \frac{1}{\kappa_P}\right)$ is now considered, the trajectories are ultimately bounded by

$$\|\mathbf{r}\| \leq \max\left(\frac{\beta_d}{\varphi_P\,\delta}, \frac{1}{\kappa_P}\right)\sqrt{\frac{\lambda_M}{\lambda_m}}.$$

Interestingly, by properly choosing the functions $\varphi(t)$ and $\kappa(t)$, this control scheme can also guarantee that a user defined tracking performance is ultimately met (robust performance). In fact, with the use of the complete feedback control law composed of (3.22), (3.32), and

$$\begin{aligned}\varphi(t) &= \Big(\varphi_P + \varphi_I \int_0^t f(\|\mathbf{e}\|)\,d\tau\Big),\\ \kappa(t) &= \Big(\kappa_P + \kappa_I \int_0^t f(\|\mathbf{e}\|)\,d\tau\Big),\end{aligned}$$

where κ_I, $\varphi_I > 0$, $\mathbf{K}_r$ diagonal and positive definite, and

$$f(\|\mathbf{e}\|) = \begin{cases} 0 & \text{if } \|\mathbf{e}\| < \epsilon, \\ \|\mathbf{e}\| & \text{otherwise,} \end{cases}$$

any user-defined tracking precision $\epsilon > 0$ can ultimately be met. This result can be shown by contradiction. For this purpose one can suppose that

$$\nexists\, t_1, \;\; \forall\, t, \quad t > t_1 \; : \; \|\mathbf{e}\| < \epsilon. \tag{3.38}$$

With this supposition, the increasing functions $\kappa(t)$ and $\varphi(t)$ grow for all t due to their integral action. As a consequence, the maximum value of $\|\mathbf{r}\|$ such that $\dot{V}(\mathbf{r})$ is surely negative decreases because in (3.36) the term $h_2(\mathbf{r})$ is non-positive for $\|\mathbf{r}\| \geq \max\left(\frac{\beta_d}{\varphi(t)\,\delta}, \frac{1}{\kappa(t)}\right)$. Then, there will be a large enough $t_2 > t_1$ such that $\max\left(\frac{\beta_d}{\varphi(t_2)\,\delta}, \frac{1}{\kappa(t_2)}\right)$ is small enough, since

$$\|\mathbf{r}\| \leq \max\left(\frac{\beta_d}{\varphi(t_2)\,\delta}, \frac{1}{\kappa(t_2)}\right)\sqrt{\frac{\lambda_M}{\lambda_m}} < \max\left(\frac{\beta_d}{\varphi(t_1)\,\delta}, \frac{1}{\kappa(t_1)}\right)\sqrt{\frac{\lambda_M}{\lambda_m}},$$

to guarantee that for $t \geq t_2$ the trajectories $\mathbf{r}$ ultimately stay within a ball small enough such that $\|\mathbf{e}\| < \epsilon$ for any selection of the gains $\mathbf{K}_r$. This contradicts the supposition (3.38) and proves the result.

The proposed robust control schemes have important implications for the robust control of modular robot manipulators, thanks to the fact that the need for time-consuming and non-formal procedures for estimating bounds of perturbations from model uncertainties is removed. With the proposed approaches, a measure of the worst-case closed-loop perturbation is computed automatically using interval arithmetic. This measure is then directly used online for feedback control. Additionally, thanks to the last result presented of robust performance control, accurate tuning of $\mathbf{K}_r$ is not required since the increase of $\kappa(t)$ and $\varphi(t)$ automatically meets the tracking performance. Particular attention should be paid when implementing the integral actions of $\kappa(t)/\varphi(t)$ in practice. This aspect is detailed in Section 3.6, which is devoted to experimental application.

3.4.2 Efficient Computation of the Controllers

The price to pay for the benefits introduced by the interval-arithmetic-based robust controllers is the increase of computational complexity with respect to conventional schemes. For both the inverse-dynamics and the passivity-based version, a simple way to implement the controller is to use a software with symbolic manipulation capabilities and obtain the perturbation functions $\mathbf{w}_{ID}/\mathbf{w}_{PB}$ analytically. Then, software packages which support interval arithmetic computations can be used for evaluating these functions with the interval arguments. This approach has been employed in [125] for the comparison using simulations. In that case, only a planar two degrees of freedom robot has been considered.

With the increase of the complexity of the robotic structure (e.g., when more degrees of freedom are present), the computational complexity for analytically obtaining and for evaluating the perturbation functions $\boldsymbol{\rho}([\mathbf{\Phi}_{ID}])/\boldsymbol{\rho}([\mathbf{\Phi}_{PB}])$ increases significantly. In fact, the increase of computational complexity for the evaluation of the perturbation functions as interval valued functions endangers the online applicability of the interval-arithmetic-based robust controllers, especially for robots with a large number of degrees of freedom. In addition, considering the desired automatic deployment of the control for the modular robotic setting considered in this thesis, the central control unit would require expensive computational resources, such as those for handling symbolic algebraic manipulation.

In this subsection, an algorithm that removes the above mentioned threats is described. This approach has been first presented in [126] and allows one to perform the computations required for obtaining $\boldsymbol{\rho}([\mathbf{\Phi}_{ID}])/\boldsymbol{\rho}([\mathbf{\Phi}_{PB}])$ efficiently. They can be carried out numerically online for robots with many degrees of freedom as well, without the need for software with symbolic variable manipulation capabilities.

The proposed approach allows the computation of formally guaranteed over-approximative sets of perturbing torques/forces, arising from imperfect knowledge of dynamic model parameters. This is obtained by simply combining the use of the recursive Newton-Euler algorithm with the set-based operations from Definitions 4, 5, and 6. Starting from Algorithm 3 in Appendix A.1, set-based operations are included. The resulting interval-arithmetic-based Newton-Euler algorithm denoted by $IANEA_g^*(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_a, \ddot{\mathbf{q}}, \mathbf{DH}, [\boldsymbol{\Delta}])$ is presented in Appendix A.2. The use of set-based operations in this algorithm allows one to handle multidimensional interval vectors of the dynamical parameters to directly compute over-approximative sets of joint torques/forces as

$$IANEA_g^*(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbf{DH}, [\boldsymbol{\Delta}]) = [\mathbf{u}].$$

It is now possible to show in detail how $\boldsymbol{\rho}([\boldsymbol{\Phi}_{ID}])/\boldsymbol{\rho}([\boldsymbol{\Phi}_{PB}])$ can be computed efficiently by using the proposed algorithm. The worst-case perturbation for the IA-ID controller can be computed by considering that

$$\begin{aligned}\mathbf{w}_{ID}(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{y}, \boldsymbol{\Delta}_0, \boldsymbol{\Delta}, \mathbf{d}) &\overset{(3.6)}{=} \mathbf{M}(\mathbf{q}, \boldsymbol{\Delta})\mathbf{y} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\Delta}) - \mathbf{M}_0(\mathbf{q})\mathbf{y} + \mathbf{n}_0(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{d} \\ &= IANEA_g^*(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}, \mathbf{y}, \mathbf{DH}, \boldsymbol{\Delta}) \\ &\qquad - NEA_g^*(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}, \mathbf{y}, \mathbf{DH}, \boldsymbol{\Delta}_0) - \mathbf{d}.\end{aligned}$$

By introducing the non-degenerate interval for uncertain dynamic parameters, the following holds $\forall\, \mathbf{d} \in [\mathbf{d}]$, $\boldsymbol{\Delta} \in [\boldsymbol{\Delta}]$:

$$\begin{aligned}\mathbf{w}_{ID}(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}, \mathbf{y}, \boldsymbol{\Delta}_0, \boldsymbol{\Delta}, \mathbf{d}) \in [\Phi_{ID}] = IANEA_g^*(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}, \mathbf{y}, \mathbf{DH}, [\boldsymbol{\Delta}]) \\ \ominus NEA_g^*(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}, \mathbf{y}, \mathbf{DH}, \boldsymbol{\Delta}_0) \ominus [\mathbf{d}].\end{aligned} \tag{3.39}$$

Similarly, the worst-case perturbation of the passivity-based variant is computed by considering the following relation

$$\begin{aligned}\mathbf{w}_{PB}(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_a, \ddot{\mathbf{q}}_a, \boldsymbol{\Delta}_0, \boldsymbol{\Delta}, \mathbf{d}) &\overset{(3.26)}{=} \mathbf{M}(\mathbf{q}, \boldsymbol{\Delta})\ddot{\mathbf{q}}_a + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\Delta})\dot{\mathbf{q}}_a + \mathbf{g}(\mathbf{q}, \boldsymbol{\Delta}) \\ &\qquad - \big(\mathbf{M}_0(\mathbf{q})\ddot{\mathbf{q}}_a + \mathbf{C}_0(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}_a + \mathbf{g}_0(\mathbf{q})\big) - \mathbf{d} \\ &= IANEA_g^*(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_a, \ddot{\mathbf{q}}_a, \mathbf{DH}, \boldsymbol{\Delta}) \\ &\qquad - NEA_g^*(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_a, \ddot{\mathbf{q}}_a, \mathbf{DH}, \boldsymbol{\Delta}_0) - \mathbf{d}.\end{aligned}$$

Introducing $[\boldsymbol{\Delta}]$ yields $\forall\, \mathbf{d} \in [\mathbf{d}]$, $\boldsymbol{\Delta} \in [\boldsymbol{\Delta}]$:

$$\begin{aligned}\mathbf{w}_{PB}(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_a, \ddot{\mathbf{q}}_a, \boldsymbol{\Delta}_0, \boldsymbol{\Delta}, \mathbf{d}) \in [\Phi_{PB}] = IANEA_g^*(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_a, \ddot{\mathbf{q}}_a, \mathbf{DH}, [\boldsymbol{\Delta}]) \\ \ominus NEA_g^*(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_a, \ddot{\mathbf{q}}_a, \mathbf{DH}, \boldsymbol{\Delta}_0) \ominus [\mathbf{d}].\end{aligned} \tag{3.40}$$

The linear computational complexity of $IANEA_g^*$ and NEA_g^* in the number of joints is carried over to (3.39) and (3.40). The proposed approach qualifies for online use, due to the

Joint Module	Proximal	$[m^{pl}]$	$[\mathbf{I}_{pl}^{in}]$	$[\mathbf{r}_{Cpl}^{in}]$
	Distal	$[m^{dl}]$	$[\mathbf{I}_{dl}^{out}]$	$[\mathbf{r}_{Cdl}^{out}]$
	Joint	$[\beta_v]$	$[\beta_c]$	$[I_m]$
Link Module		$[m^l]$	$[\mathbf{I}_l^{in}]$	$[\mathbf{r}_{Cl}^{in}]$

Table 3.1: Additional module data for robust control design.

intrinsic computational efficiency and because there is no need for symbolic variable manipulation. With $[\boldsymbol{\Phi}_{ID}]$ (or $[\boldsymbol{\Phi}_{PB}]$) efficiently computed, the measure of the worst-case perturbation $\boldsymbol{\rho}([\boldsymbol{\Phi}_{ID}])$ (or $\boldsymbol{\rho}([\boldsymbol{\Phi}_{PB}])$) that can be used for feedback control is straightforwardly obtained using (3.7) (or (3.28)).

3.4.3 Application to Modular Robot Manipulators

A small extension of the framework presented in Chapter 2 and the interval-arithmetic-based robust control approach enables the automatic deployment of robust controllers for modular robot manipulators. This is possible by simply enlarging the set of parameters of the modules to be stored within them or in a central database.

The additional data required are the lower and upper bounds of each parameter of the module data for dynamics. This additional set of required parameters is compactly collected in Table 3.1. For each assembled arm, when the module data are processed for obtaining the arm-link parameters as described in Subsection 2.4.2, an equivalent procedure can be performed to compute the interval in which these values can lie. This can be done by employing the set-based operations in the Definitions 4 and 5. For the ith assembled link (with reference to the procedure described in Subsection 2.4.2 and Figure 2.6b), subsequent auxiliary distal parts are recursively considered first if one or more link modules are present. By considering interval variables, the operations in (2.17), (2.19), and (2.20) simply become

$$[m_j^a] = [m_{j-1}^{dl}] \oplus [m_j^l], \quad [\mathbf{I}_{a_j}^{io}] = [\mathbf{I}_{dl_{j-1}}^{out}] \oplus [\mathbf{I}_{l_j}^{in}],$$

$$[\mathbf{r}_{Ca_j}^{io}] = \big([m_{j-1}^{dl}] \odot [\mathbf{r}_{Cdl_{j-1}}^{out}] \oplus [m_j^l] \odot [\mathbf{r}_{Cl_j}^{in}]\big) \oslash [m_j^a],$$

$$\begin{pmatrix}[\mathbf{r}_{Ca_j}^{out}]\\ 1\end{pmatrix} = \left(\mathbf{A}_{out,a_j}^{io,a_j}\right)^{-1} \odot \begin{pmatrix}[\mathbf{r}_{Ca_j}^{io}]\\ 1\end{pmatrix},$$

and

$$
\begin{aligned}
[\mathbf{I}^{out}_{a_j}] = \left(\mathbf{R}^{io,a_j}_{out,a_j}\right)^T \odot \Big([\mathbf{I}^{io}_{a_j}] \ominus [m^a_j] \odot \mathbf{S}^T([\mathbf{r}^{io}_{Ca_j}]) \odot \mathbf{S}([\mathbf{r}^{io}_{Ca_j}])\Big) \odot \mathbf{R}^{io,a_j}_{out,a_j}& \\
\oplus [m^a_j] \odot \mathbf{S}^T([\mathbf{r}^{out}_{Ca_j}]) \odot \mathbf{S}([\mathbf{r}^{out}_{Ca_j}])&,
\end{aligned}
$$

where $\mathbf{A}^{io,a_j}_{out,a_j}$, $\mathbf{R}^{io,a_j}_{out,a_j}$ are computed as in (2.18). Once this recursion is completed, the remaining connection to be considered is between the last auxiliary distal part obtained and the proximal part of the joint modules that concludes the assembled arm-link. In this case, with reference to Figure 2.6b, an equivalent approach described in Subsection 2.4.2 can be followed using (2.21), (2.23), and (2.24) with interval variables, which provide the following relations:

$$
\begin{gathered}
[m_i] = [m^{dl}_{j-1+k}] \oplus [m^{pl}_{j+k}], \quad [\mathbf{I}^{io}_i] = [\mathbf{I}^{out}_{dl_{j-1+k}}] \oplus [\mathbf{I}^{in}_{pl_{j+k}}], \\
[\mathbf{r}^{io}_{C_i}] = \left([m^{dl}_{j-1+k}] \odot [\mathbf{r}^{out}_{Cdl_{j-1+k}}] \oplus [m^{pl}_{j+k}] \odot [\mathbf{r}^{in}_{Cpl_j}]\right) \oslash [m_i], \\
\begin{pmatrix} [\mathbf{r}^i_{D_i,C_i}] \\ 1 \end{pmatrix} = \left(\mathbf{A}^{io}_i\right)^{-1} \odot \begin{pmatrix} [\mathbf{r}^{io}_{C_i}] \\ 1 \end{pmatrix},
\end{gathered}
$$

and finally

$$
[\mathbf{I}^i_i] = \mathbf{R}^{io^T}_i \odot \Big([\mathbf{I}^{io}_i] \ominus [m_i] \odot \mathbf{S}^T([\mathbf{r}^{io}_{C_i}]) \odot \mathbf{S}([\mathbf{r}^{io}_{C_i}])\Big) \odot \mathbf{R}^{io}_i,
$$

where $\mathbf{A}^{io}_i$, $\mathbf{R}^{io}_i$ are computed as in (2.22).

The interval parameters obtained ($[m_i]$, $[\mathbf{I}^i_i]$, $[\mathbf{r}^i_{D_i,C_i}]$) can be directly used to compose $[\boldsymbol{\Delta}]$. This vector, together with the vector of parameters $\boldsymbol{\Delta}_0$ that can be computed using the procedure in Subsection 2.4.2 and nominal module data, can be directly used to automatically deploy the interval-arithmetic-based robust control.

3.5 Performance Evaluation using Simulations

In this section, simulation results of the implementation of the interval-arithmetic-based control schemes are shown. The simulations involve a comparison with existing robust control approaches: a classical method for robust control of rigid robots and the method of Zenieh and Corless [120], which can provide ultimate robust performance control. The simulations have been performed using MATLAB and Simulink R2015b. The Interval Laboratory [91] has been used for performing computations with interval arithmetic due to the simplicity of the considered case study. After a description of the simulation test bed, the essence of both the classical approach for robust control of robot manipulators and the method of Zenieh and Corless are

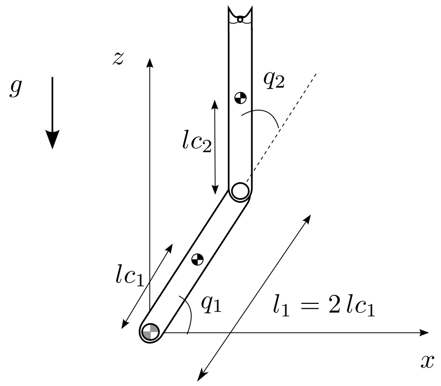


Figure 3.4: Illustration of the two links robot manipulator example.

recalled. Then, the simulation results are presented and discussed.

Simulation Test Bed

A simple two links planar robot manipulator with revolute joints that moves in the vertical plane (and thus is subject to gravity) is considered. Such a system is a popular case study for illustrating and testing control laws for robots. A representation of this arm is shown in Figure 3.4, and the terms of its dynamical model can be found in (3.41), (3.42), and (3.43). For brevity, friction is not considered in the simulations. Perfect knowledge of the dynamical parameters is assumed for the first link, while uncertainty is considered for the second one. The dynamical parameters of the links, and the considered amount of the uncertainty, are collected in Table 3.2.

$$\mathbf{M}(\mathbf{q},\Delta) = \begin{pmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{pmatrix}, \tag{3.41}$$

$$M_{11} = I_1 + I_2 + m_1\,lc_1 + m_2\,(l_1^2 + lc_2^2 + 2\,l_1\,lc_2\,cos(q_2)),$$

$$M_{12} = M_{21} = I_2 + m_2\,(lc_2^2 + l_1\,lc_2\,cos(q_2)),\;\; M_{22} = I_2 + m_2\,lc_2^2.$$

$$\mathbf{C}(\mathbf{q},\dot{\mathbf{q}},\Delta) = \begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix} \tag{3.42}$$

$$C_{11} = -m_2\,l_1\,lc_2\,sin(q_2)\,\dot{q}_2,\;\; C_{12} = -m_2\,l_1\,lc_2\,sin(q_2)\,(\dot{q}_1 + \dot{q}_2),$$

$$C_{21} = m_2\,l_1\,lc_2\,sin(q_2)\,\dot{q}_1,\;\; C_{22} = 0.$$

	Link 1	Link 2	Uncertainty
Mass (kg)	10	$5+\delta_m$	$0 \leq \delta_m \leq 5$
Inertia $(kg\,m^2)$	10/12	$10/12+\delta_I$	$0 \leq \delta_I \leq 15/12$
Center of mass (m)	0.5	$0.5+\delta_c$	$0 \leq \delta_c \leq 0.025$

Table 3.2: Dynamical parameters of the links for the simulation test bed.

$$\mathbf{g}(\mathbf{q},\Delta) = \begin{pmatrix} g_1 \\ g_2 \end{pmatrix}, \tag{3.43}$$

$$g_1 = (m_1\, lc_1 + m_2\, l_1)\, g\, cos(q_1) + m_2\, lc_2\, g\, cos(q_1+q_2),$$

$$g_2 = m_2\, lc_2\, g\, cos(q_1+q_2).$$

The simulations include one ideal case (Scenario 1) where no sampling effects and no input disturbances are considered. Then, a more realistic case (Scenario 2) is assumed, which includes the sampling rate at $1\,kHz$, as well as additive Gaussian noise on the velocity readings. Scenarios 1 and 2 are used to test the controllers that provide robust stability: a classical robust control scheme (recalled next in Subsection 3.5.1) and the IA-ID control. Two additional scenarios are considered. Scenario 3 includes uniformly distributed input disturbances and a sampling rate at $10\,kHz$, while Scenario 4 includes additive Gaussian noise on the velocity readings and a sampling rate of $1\,kHz$. Sampling effects have been included by using zero-order hold blocks. Scenarios 3 and 4 are used for testing controllers that provide robust performance: the r-α tracker (recalled next in Subsection 3.5.2) and the IA-PB control. The details of all the considered scenarios are collected in Table 3.3. In this table, μ denotes the mean of the additive Gaussian noise considered, and σ its standard deviation. For simulating all scenarios, the real (unknown for control design purposes) dynamical parameters of the second link are set to be $m_2 = 10\ kg$, $I_2 = 25/12\ kg\,m^2$, and $lc_2 = 0.525\ m$. In all the simulations shown, the following initial conditions have been used: $\mathbf{q}(0) = 2\cdot 10^{-2}\,(1 \quad -1)^T$, $\dot{\mathbf{q}}(0) = (0 \quad 0)^T$. The bounds on the input disturbance are assumed unknown for the deployment of the IA-PB control. The simulations are performed using the test trajectory

$$\mathbf{q}_d(t) = \left(\frac{\pi}{4}\,\sin\left(\pi\, t/2\right)\, \sin\left(\pi\, t\right) \quad -\frac{\pi}{8}\,\sin\left(\pi\, t/2\right)\, \sin\left(\pi\, t\right)\right)^T, \tag{3.44}$$

which is represented in Figure 3.5, and has $\mathbf{q}_d(0) = \mathbf{0}$ and $\dot{\mathbf{q}}_d(0) = \ \mathbf{0}$.

	Input disturb. (Nm)	Sampling time (s)	Required performance (rad)	Velocity meas. noise (rad/s)
Scenario 1 (ideal)	none	none	$\|\mathbf{e}\| < 10^{-2}$	none
Scenario 2	none	10^{-3}	$\|\mathbf{e}\| < 10^{-2}$	$\mu = 0$, $\sigma \approx 3 \cdot 10^{-3}$
Scenario 3	$\|\mathbf{d}\| \leq 10$	10^{-4}	$\|\mathbf{e}\| < 10^{-2}$	none
Scenario 4	$\|\mathbf{d}\| \leq 10$	10^{-3}	$\|\mathbf{e}\| < 10^{-2}$	$\mu = 0$, $\sigma \approx 10^{-2}$

Table 3.3: Simulation scenarios.

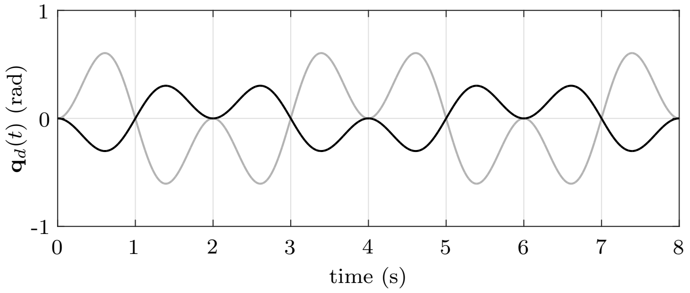


Figure 3.5: Test trajectory used in the simulations. The first component of $\mathbf{q}_d(t)$ is shown in light-gray and the second in black.

3.5.1 Classical Robust Control

Among the most popular approaches for designing the robust control of robot manipulators is the enhancement of a nominal inverse-dynamics control scheme by means of a robust control term obtained from Lyapunv's second method [96]. Such schemes typically rely on the following assumptions:

$$0 < B_m \leq \|\mathbf{M}(\mathbf{q}, \mathbf{\Delta})^{-1}\| \leq B_M < \infty \quad \forall \mathbf{q},\ \mathbf{\Delta} \in [\mathbf{\Delta}],$$

$$\|\mathbf{I} - \mathbf{M}(\mathbf{q}, \mathbf{\Delta})^{-1}\mathbf{M}_0(\mathbf{q})\| \leq \alpha_{cr} \leq 1 \quad \forall \mathbf{q},\ \mathbf{\Delta} \in [\mathbf{\Delta}],$$

$$\|\mathbf{n}_0(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{\Delta})\| < \eta_{cr} \quad \forall \mathbf{q},\ \dot{\mathbf{q}},\ \mathbf{\Delta} \in [\mathbf{\Delta}],$$

$$\max(\|\ddot{\mathbf{q}}_d\|) < Q_M < \infty \quad \forall \ddot{\mathbf{q}}_d,$$

for some positive constants B_m, B_M, α_{cr}, η_{cr}, Q_M. A classical robust control law that can be employed is

$$\mathbf{u} = \mathbf{M}_0\left(\mathbf{q}\right)\left(\ddot{\mathbf{q}}_d + \mathbf{K}_D\dot{\mathbf{e}} + \mathbf{K}_P\mathbf{e} + \boldsymbol{\nu}\right) + \mathbf{n}_0\left(\mathbf{q}, \dot{\mathbf{q}}\right).$$

From this position, the inverse dynamics control law can be enhanced by introducing the following robustifying action:

$$\boldsymbol{\nu} = \begin{cases} \frac{\rho_{cr}(\boldsymbol{\xi})}{\|\mathbf{z}\|}\mathbf{z}, & \|\mathbf{z}\| \geq \delta, \\ \frac{\rho_{cr}(\boldsymbol{\xi})}{\delta}\mathbf{z}, & \|\mathbf{z}\| < \delta, \end{cases} \tag{3.45}$$

with $\delta > 0$, $\mathbf{z} = \mathbf{B}^T\mathbf{P}\boldsymbol{\xi}$ and

$$\rho_{cr}(\boldsymbol{\xi}) \geq \frac{1}{1-\alpha_{cr}} \left(\alpha_{cr} Q_M + \alpha_{cr}\| \left(\mathbf{K}_P, \mathbf{K}_D\right) \| \|\boldsymbol{\xi}\| + B_M \eta + \beta_d\right).$$

Please note that in the above, the terms ξ, $\mathbf{P}$, and $\mathbf{B}$ are the same as in (3.10), (3.12) and (3.13). Additionally, it is worth stressing that none of the interval-arithmetic-based controllers introduced earlier require a switching function like (3.45).

With this robust control approach, it can be shown that the error trajectories are uniformly ultimately bounded and that the bounds depend on δ, i.e., the larger δ is, the larger the resulting bounds on the tracking error norm [96]. While this implies that such a control scheme achieves robust *stability*, when robust *performance* is desired, the phenomenon of chattering may arise. This becomes clear when considering that the increase of the tracking performance is achieved by reducing δ, but by reducing δ a discontinuous control law is approached. Even though global asymptotic stability can theoretically be shown for $\delta = 0$, the chattering phenomenon may be problematic for real world deployment. Additionally, this robust control approach is typically conservative due to the need for obtaining the considered bounds for the entire, practically reachable state space.

It is worth mentioning that a more recent revision of this approach has been presented in [10], which aims at reducing conservativity of the robustifying term. However, this approach has not been described in this thesis because of a practical inconsistency experienced in its deployment. In fact, during the attempt to implement it, it was difficult to find meaningful constants α_0, α_1, and α_2 used in [10] that should guarantee

$$\|\mathbf{n}_0(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\Delta})\| \leq \alpha_0 + \alpha_1\|\boldsymbol{\xi}\| + \alpha_2\|\boldsymbol{\xi}\|^2,$$

since the function $\mathbf{n}_0(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{n}_0(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\Delta})$ does not depend on the tracking error.

This brief description of the classical robust control approach for robot manipulators may have now shed additional light on the difficulty that such an approach would introduce when considering a required quick robot deployment. In practice, the required procedures for estimating the bounds of uncertainties, and the selection of the smoothing parameter to avoid chattering, may result in a significant limitation especially when automatic deployment is desired, e.g., for a modular robot manipulator right after assembly.

To apply this scheme to the considered simulation scenarios, a procedure for estimating the required bounds has been implemented. This estimation has been performed for $0 \leq q_i \leq 2\,\pi$, $0 \leq \dot{q}_i \leq \pi$ for $i = 1, 2$, $0 \leq \delta_m \leq 5$, $0 \leq \delta_I \leq 15/12$, and $0 \leq \delta_c \leq 0.025$ using 10^5 samples (units are SI) and provides the following bounds:

$$B_m = 0.029, \;\; B_M = 1.158, \;\; \alpha_{cr} = 0.977, \;\; \eta_{cr} = 128.35, \;\; Q_M = 10.3.$$

3.5.2 r-α Tracking Control

The r-α tracker is an approach proposed by Zenieh and Corless in [120], in which the resulting controller is remarkably simple. After a study of the uncertain model components for obtaining specific uncertainty bounds, the control law can be implemented without requiring model-based computations online. Such a control scheme guarantees tracking of a desired trajectory with a prescribed rate of convergence α within a user-defined bound r. Since the bound r can be directly selected by the user, the approach qualifies for providing control with robust performance. The controller design process starts from the computation of the uncertainty bounds β_i for $i \in \{0, \ldots, 3\}$, for all $\mathbf{q}$, $\dot{\mathbf{q}}$, $\mathbf{d} \in [\mathbf{d}]$ and $\Delta \in [\Delta]$ such that

$$\lambda_{min}\left(\mathbf{M}(\mathbf{q}, \Delta)\right) \geq \beta_0 \geq 0, \;\; \lambda_{max}\left(\mathbf{M}(\mathbf{q}, \Delta)\right) \leq \beta_1,$$
$$\|\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}, \Delta)\| \leq \beta_2 \|\dot{\mathbf{q}}\|, \;\; \|\mathbf{g}(\mathbf{q}, \Delta) - \mathbf{d}\| \leq \beta_3.$$

For the estimation of β_2, as proposed in [120], one can consider that $\beta_2 \geq \sqrt{\sum_{i=1}^{N} \|\mathbf{L}_i(\mathbf{q}, \Delta)\|^2}$, where $\mathbf{L}_i(\mathbf{q}, \Delta)$ is a square matrix such that for $\mathbf{y} := \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}, \Delta)\mathbf{v}$, $y_i = \dot{\mathbf{q}}^T \mathbf{L}_i(\mathbf{q}, \Delta)\mathbf{v}$. Once these bounds are computed, an additional tuning parameter δ_{zc} and two symmetric positive definite matrices (Λ, $\mathbf{Q}$) are selected such that

$$\lambda_{min}\left(\Lambda\right) \geq \alpha, \;\; \lambda_{min}\left(\mathbf{Q}\right) \geq \alpha\,\beta_1, \;\; \delta_{zc} \leq (\alpha\, r)^2 \lambda_{min}\left(\mathbf{Q}\right)\, \beta_0/\beta_1.$$

The r-α tracking controller can then be easily implemented as in [120]:

$$\mathbf{u} = \mathbf{Q}\mathbf{r} + \left(\|\rho_{zc}\,\mathbf{r}\| + \delta_{zc}\right)^{-1} \rho_{zc}\mathbf{r},$$
$$\rho_{zc} = \beta_1 \|\ddot{\mathbf{q}}_a\| + \beta_2 \|\dot{\mathbf{q}}_a\| \|\dot{\mathbf{q}}\| + \beta_3.$$

In the above equation, the terms $\mathbf{r}$ and $\dot{\mathbf{q}}_a$ are the same as in (3.25) and (3.23).

Similar to the classical robust control case, a sampling procedure for estimating the required bounds has been implemented, for applying this approach to the considered simulation scenario. This estimation has been performed for $0 \leq q_i \leq 2\,\pi$, $-\beta_d \leq d_i \leq \beta_d$ for $i = 1, 2$, $0 \leq \delta_m \leq 5$,

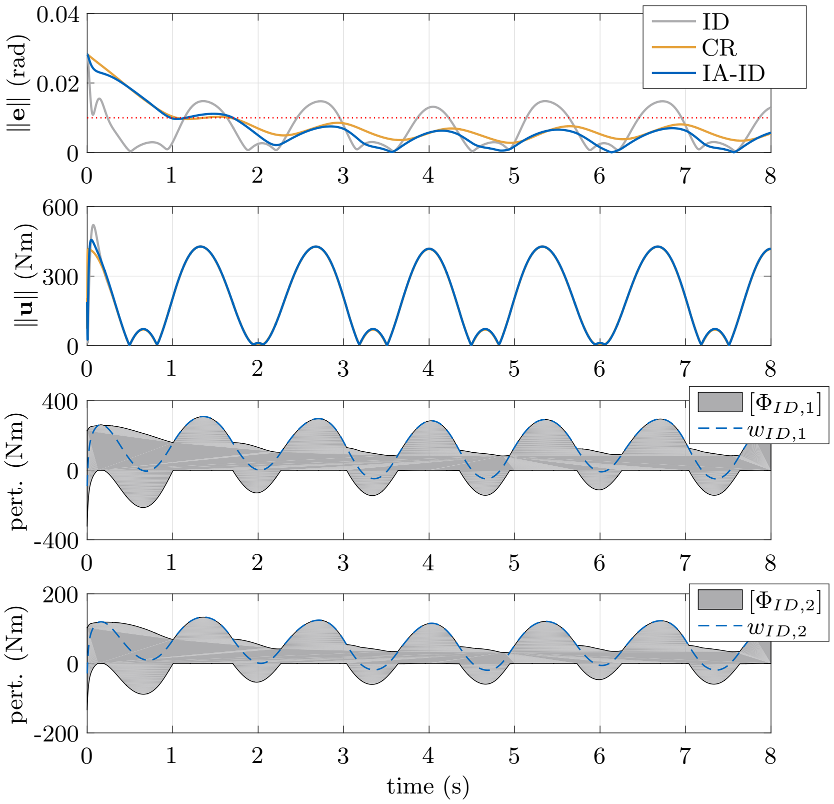


Figure 3.6: Simulation of the tracking performance of the interval-arithmetic inverse-dynamics controller (IA-ID), with respect to a classical robust control approach (CR) and the nominal inverse-dynamics controller (ID). No sampling effects have been considered in this test (Scenario 1 in Table 3.3).

$0 \leq \delta_I \leq 15/12$, and $0 \leq \delta_c \leq 0.025$ using 10^5 samples (units are SI). The procedure gives the following bounds:

$$\beta_0 = 0.86, \quad \beta_1 = 34.27, \quad \beta_2 = 9.95, \quad \beta_3 = 210.65.$$

3.5.3 Performance Comparison

In this subsection, the performance of the proposed and recalled controllers for the scenarios in Table 3.3 are shown and discussed. Controllers that are able to provide robust stability are first considered, followed by the comparison of those that can provide robust performance.

Controllers Providing Robust Stability

Among the presented controllers, those that do not directly take a user-defined tracking performance at the time of their deployment into account are the IA-ID and the classical

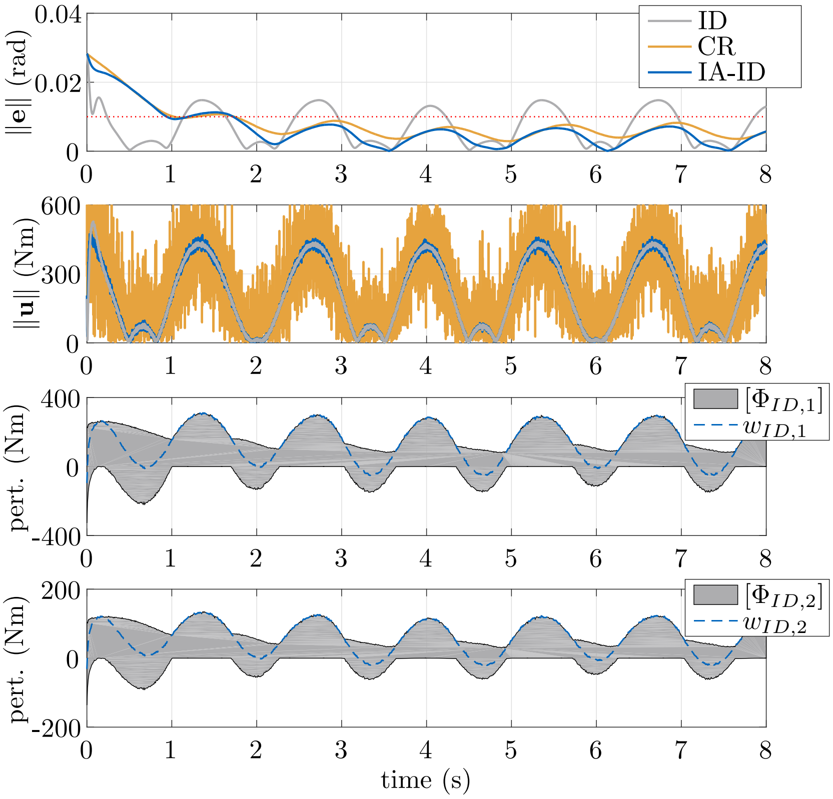


Figure 3.7: Simulation of the tracking performance of the interval-arithmetic inverse-dynamics controller (IA-ID), with respect to a classical robust control approach (CR) and the nominal inverse-dynamics controller (ID). Sampling effects (1ms sampling time) and additive noise on the joint velocity measurements have been considered in this test (Scenario 2 in Table 3.3).

robust control scheme. Theoretically, the parameter κ_P, which let the controller achieve a specific user defined performance, can be found for the former; this, however, would require knowing the inertia matrix bounds (ruining the benefit that no knowledge of bounds are required for IAB robust controllers). For the latter, the parameter δ needs to be selected so that it is small enough. However, the smaller this parameter, the higher the risk of introducing chattering, which makes it difficult to use a δ analytically found that is typically affected by large conservativity. In practice, these two controllers provide ultimate uniform boundedness of the trajectories whose bounds are reduced by increasing/reducing κ_P/δ as manual tuning parameters.

These two control schemes have been tested for Scenarios 1 and 2. Figure 3.6 and Figure 3.7 show the results for the simulations with Scenarios 1 and 2, respectively. In both figures, the norm of the tracking error and of the control command is shown for three cases: no robust control action is added to the nominal inverse-dynamics control (gray), a classical robust control scheme is employed (orange), and an IA-ID controller in used (blue). By observing first Figure 3.6, the performance of the controller with no robust control action are shown to exceed the desired tracking performance bounds (set to 0.01 rad). With the use of the robust control terms, both the classical robust control and the interval-arithmetic-based one can achieve the required performance, after tuning of κ_P/δ. In fact, these two parameters have been manually tuned such that the controllers provide similar performance in this ideal case (Scenario 1). Thanks to the ideal scenario considered and the simulation environment, this tuning procedure has not been difficult. However, as soon as the scenario becomes more realistic as in Scenario 2, the problem of high amplification of the measurement noise becomes clear, e.g., by observing Figure 3.7. Even though the tracking performance provided by the two robust controllers are still similar, Figure 3.7 shows that the classical robust control scheme is much more conservative with respect to the IA-ID scheme. Fair tuning of κ_P/δ has been ensured since they have been selected to provide similar tracking performance for the ideal case. At the bottom of Figure 3.6 and Figure 3.7 the evolution over time of the perturbation vector's components are shown to be bounded by those computed on-line using interval arithmetic. The gains of the IA-ID controller and the classical robust control scheme have been selected as

$$\omega_n = 40,\ \ \zeta = 0.9,\ \ \mathbf{K}_P = \omega_n^2\,\mathbf{I}_2,\ \ \mathbf{K}_D = 2\,\zeta\omega_n\,\mathbf{I}_2,\ \ \kappa_P = 3.5,\ \ \mathbf{Q} = \mathbf{I}_2,\ \ \delta = 0.04.$$

Controllers Providing Robust Performance

The schemes that are now considered are the IA-PB control and the r-α tracker. Interestingly, these two controllers can directly take into account and (theoretically) meet any user-defined tracking performance. The simulations with these two robust performance controllers have been implemented considering Scenarios 3 and 4 in Table 3.3. The respective results are shown in Figure 3.8 and Figure 3.9. As shown in Figure 3.8, the r-α tracker (orange) is conservative for the specified tracking performance, introducing high-gains and letting the tracking error become significantly smaller than required. As a consequence of the high gains used, when sampling effects are considered, the control commands show chattering, which is a typical phenomenon also found when using discontinuous controllers. On the other hand, even when considering the sampling effects, these simulations show a smooth behavior of the IA-PB scheme (blue) that meets the tracking performance still using smooth control commands. It is important to mention that the r-α tracking controller has been experimentally evaluated for a similar robot arm in [54]. In that work, such problematic chattering effects have not been

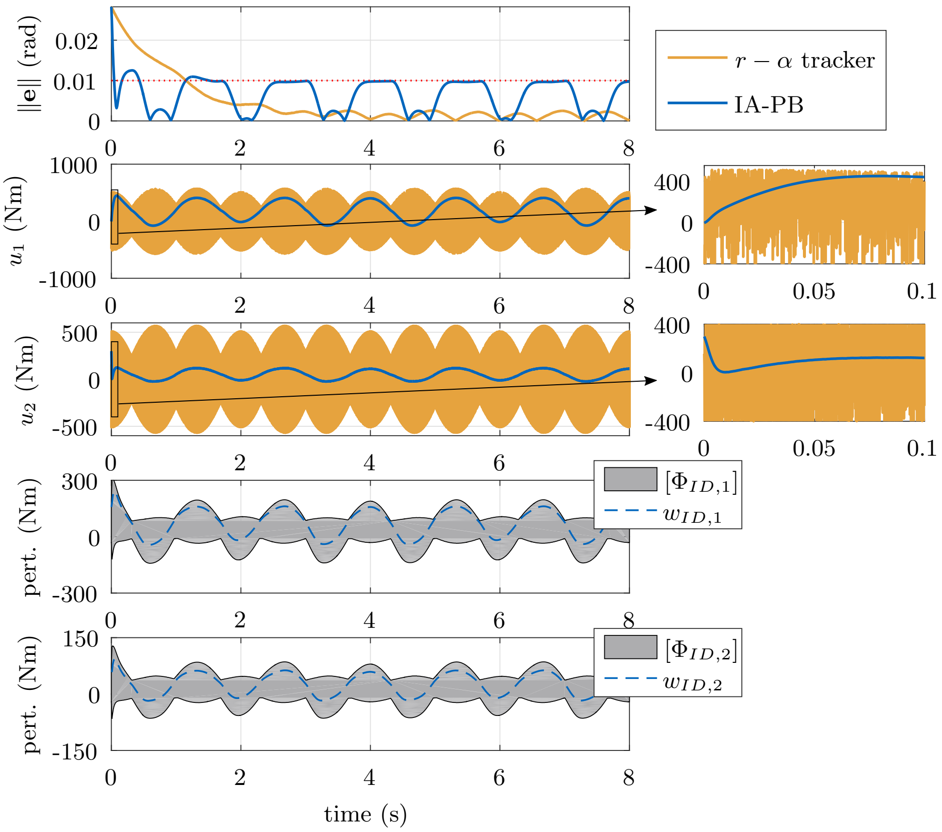


Figure 3.8: Tracking performance comparison between the interval-arithmetic passivity-based control (IA-PB) and the $r-\alpha$ tracker. Sampling effects (0.1ms sampling time), input disturbance, and no additive noise on the joint velocity measurements have been considered in this test (Scenario 3 in Table 3.3).

encountered, but the system considered in this thesis has more uncertainty and the resulting uncertainty bounds are thus larger. More importantly, gravity in [54] is also not considered since the robot arm in that work is constrained to move in the horizontal plane. The inclusion of gravity would significantly increase the conservativity of the r-α tracker.

The proposed IA-PB control law shows less conservatism, providing performances that are ultimately closer to the actual requirement. In particular, when switching from Scenario 3 (Figure 3.8) to 4 (Figure 3.9), the IA-PB controller shows remarkable insensitivity to the decrease of the sampling rate from $10kHz$ to $1kHz$, contrary to the r-α tracker. The latter instead fails to meet the required performance because of chattering. At the bottom of both Figures 3.8 and 3.9, the bounds of the perturbation vector $[\mathbf{\Phi}_{PB}]$, which have been computed online using interval arithmetic are shown to include the actual perturbation $\mathbf{w}_{PB}$. These bounds are

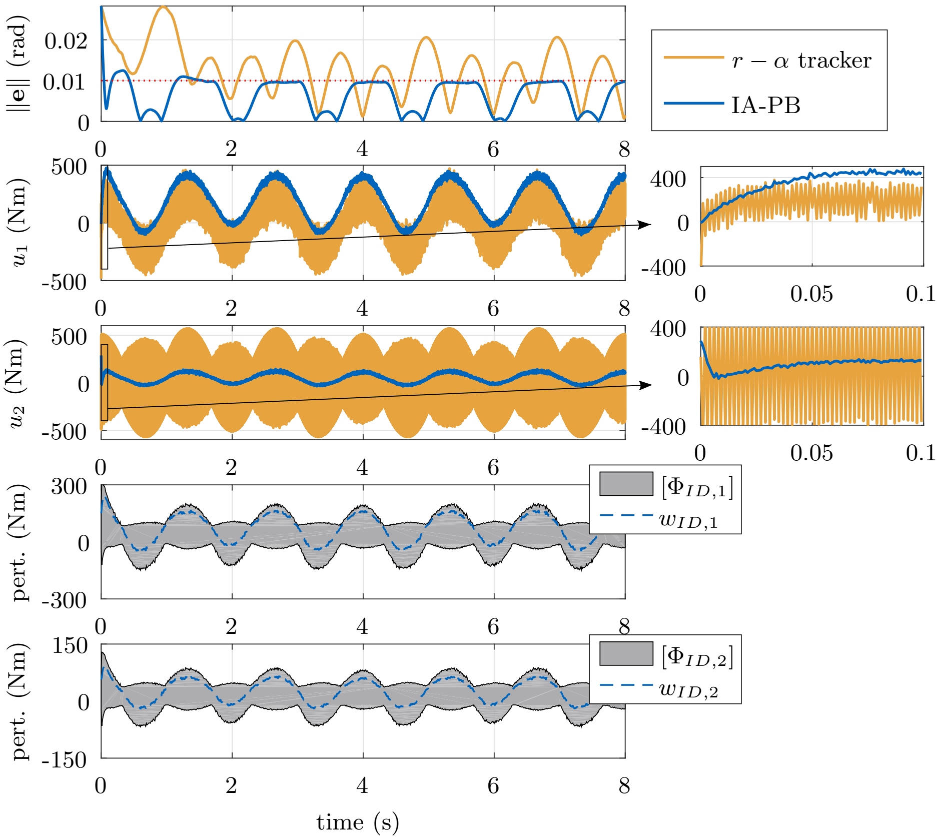


Figure 3.9: Tracking performance comparison between the interval-arithmetic passivity-based control (IA-PB) and the $r-\alpha$ tracker. Sampling effects (1ms sampling time), input disturbance, and additive noise on the joint velocity measurements have been considered in this test (Scenario 4 in Table 3.3).

significantly close to the actual perturbation values, implying that non-excessive conservativity is introduced. The tuning parameters of the IA-PB controller that have been employed are

$$\epsilon = 10^{-2}, \ \ \kappa_P = 2, \ \ \kappa_I = 2, \ \ \mathbf{K}_r = 30\,\mathbf{I}_2, \ \ \varphi_P = 100, \ \ \varphi_I = 100,$$

and those of the $r-\alpha$ tracker are

$$r = 10^{-2}, \ \ \alpha = 1, \ \ \Lambda = \mathbf{I}, \ \ \mathbf{Q} = 35\,\mathbf{I}, \ \ \delta_{zc} = 7 \cdot 10^{-5}.$$

In summary, these simulations show that the interval-arithmetic-based robust controllers solve practical difficulties of other existing methods for robust control. The price is usually paid in more computational complexity; this, however, is solved by using the algorithm proposed in [126], as discussed in Subsection 3.4.2.

3.6 Experimental Application

In addition to numerical simulations, the usefulness of the approach is demonstrated by real-world experiments in this section. The robot used for these experiments is Assembly III in Figure 2.11. For this evaluation, uncertainty in the modular parameters is considered as negligible for all modules, except for the end effector that can grasp different objects. The exact dynamical parameters of these objects are assumed to be unknown at the time of the controller's deployment. Only bounds for the parameters of the end effector are given, and they are supposed to bound the cases for the different grasp scenarios. For these tests, one payload of $0.5\,kg$ and one of $1\,kg$ are used. The nominal parameters for kinematics and dynamics are collected in Table 3.4, where the uncertainty bounds used for the end effector (the last link)

	Nominal parameters of links $i \in \{1, \ldots, 6\}$							Uncertainty	
	1	2	3	4	5	6		$\underline{\boldsymbol{\Delta}}$	$\overline{\boldsymbol{\Delta}}$
$\alpha_i\,(rad)$	$-\pi/2$	π	$-\pi/2$	$\pi/2$	$-\pi/2$	0			
$a_i\,(m)$	0	0.35	0	0	0	0			
$d_i\,(m)$	0.1013	0.1013	0.1013	0.3012	0	0.155			
$\theta_i - q_i\,(rad)$	0	$-\pi/2$	$-\pi/2$	0	0	0			
$m_i\,(kg)$	3.9	1.62	3.9	1	1.8	1.5	$m_6\,(kg)$	1.5	3
$c_{x,i}\,(m)$	0	-0.175	0	0	0	0	$c_{x,6}\,(m)$	0	0
$c_{y,i}\,(m)$	0.018	0	0.018	-0.129	0.012	0	$c_{y,6}\,(m)$	0	0
$c_{z,i}\,(m)$	0.019	-0.013	0.019	-0.048	0.012	-0.042	$c_{z,6}\,(m)$	-0.042	0.042
$I_{xx,i}$ $(10^{-3} kg\,m^2)$	15.3	16.8	15.3	6.6	5.4	1.9	$I_{xx,6}$ $(10^{-3} kg\,m^2)$	1.9	3.8
$I_{xy,i}$ $(10^{-3} kg\,m^2)$	0	0	0	0	0	0	$I_{xy,6}$ $(10^{-3} kg\,m^2)$	0	0
$I_{xz,i}$ $(10^{-3} kg\,m^2)$	0	0	0	0	0	0	$I_{xz,6}$ $(10^{-3} kg\,m^2)$	0	0
$I_{yy,i}$ $(10^{-3} kg\,m^2)$	11.6	24.4	11.6	1.6	4.1	1.8	$I_{yy,6}$ $(10^{-3} kg\,m^2)$	1.8	3.6
$I_{yz,i}$ $(10^{-3} kg\,m^2)$	-5.8	0	-5.8	1.8	-1.4	0	$I_{yz,6}$ $(10^{-3} kg\,m^2)$	0	0
$I_{zz,i}$ $(10^{-3} kg\,m^2)$	9.7	24.7	9.7	5.9	2.9	0.63	$I_{zz,6}$ $(10^{-3} kg\,m^2)$	0.63	1.26
$\beta_{v,i}\,(Nms)$	15.68	13.91	17.89	13.63	4.45	4.67			
$\beta_{c,i}\,(Nm)$	7.14	5.1	6.84	6.91	2.12	2.33			
$I_{m,i}$ $(10^{-5} kg\,m^2)$	5.23	5.23	5.23	5.23	1.7	1.7			
$\sigma_{r,i}$	160	160	160	160	100	100			
$\sigma_{m,i}\,(Nm/A)$	0.0429	0.0429	0.0429	0.0429	0.1	0.1			

Table 3.4: Nominal parameters of the robot and uncertainty considered for the last link.

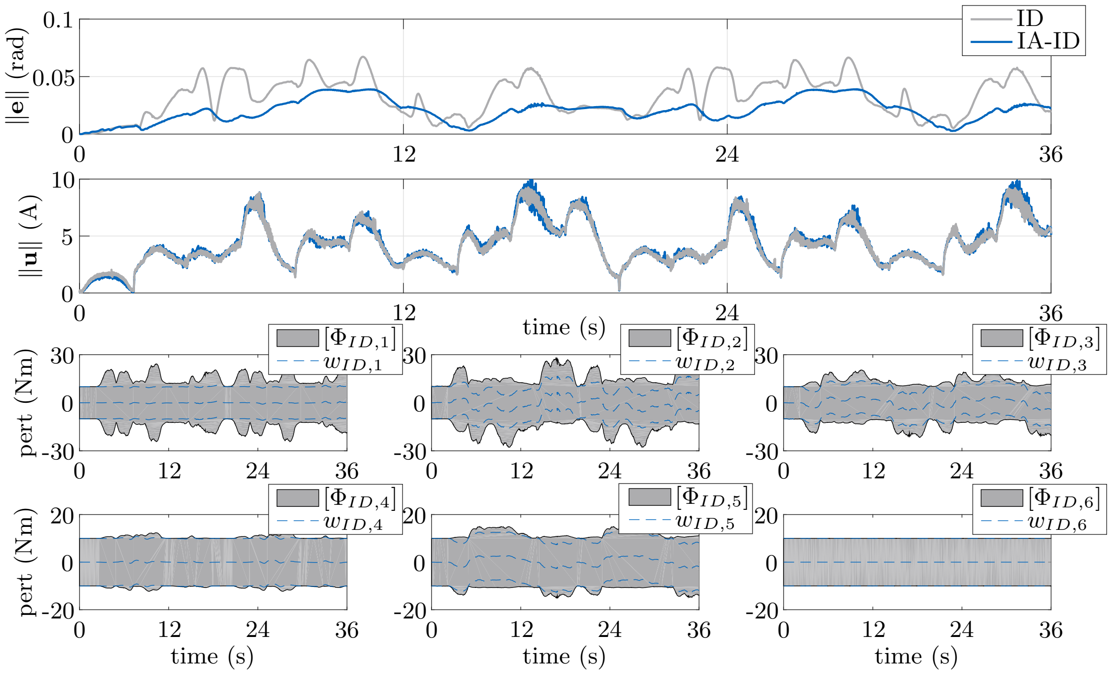


Figure 3.10: Tracking control test of the interval-arithmetic inverse-dynamics controller (IA-ID) and the nominal inverse-dynamics scheme (ID), with $0.5\,kg$ unmodeled payload. The bounds of the perturbation vector computed online using interval arithmetic are shown at the bottom.

are also collected. Considering that from Figure 2.13 in the previous chapter the model of the arm does not perfectly match (even without loads), an input disturbance is considered. This disturbance has been assumed as bounded in norm at $10\,Nm$, which has been estimated by observing the model mismatch from the results in Figure 2.13 for Assembly III.

3.6.1 Interval-Arithmetic Inverse-Dynamics Control

The first experiment presented is about the application of the interval-arithmetic inverse-dynamics control scheme, which can guarantee control with robust stability. The same trajectory used for tracking tests in the previous chapter and replicated once to double its duration is employed. The results of this experiment are collected in Figure 3.10 and Figure 3.11. Figure 3.10 shows the benefit of the inclusion of the interval-arithmetic-based robustifying term on the tracking error, with respect to the nominal inverse dynamics controller for the case of $0.5\,kg$ unmodeled payload. It should be noted that the amplification of the measurement noise that can be observed from the development over time of the norm of the current command vector is not significantly high, which implies that non-excessive conservativity is introduced.

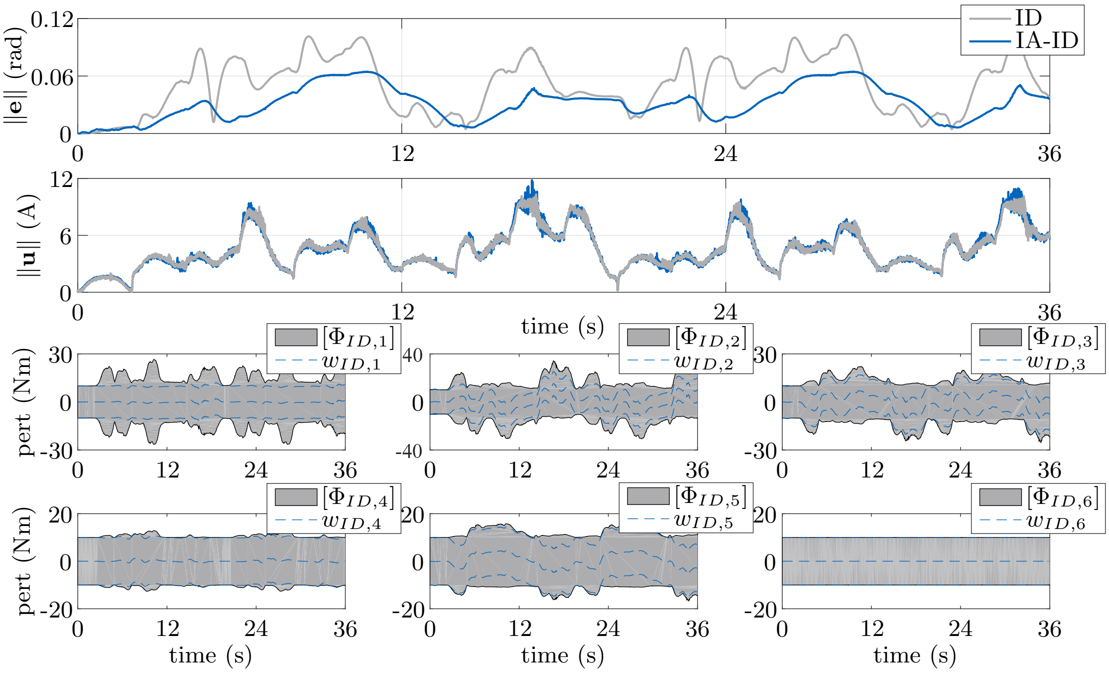


Figure 3.11: Tracking control test of the interval arithmetic inverse-dynamics controller (IA-ID) and the nominal inverse-dynamics scheme (ID), with 1 kg unmodeled payload. The bounds of the perturbation vector computed online using interval arithmetic are shown at the bottom.

This fact can also be observed more directly from the six plots at the bottom of Figure 3.10, where the bounds $[\Phi_{ID}]$ of the perturbation function $\mathbf{w}_{ID}$ that have been computed online are shown. The approach proposed in Subsection 3.4.2 for efficient computation of the interval-arithmetic-based robust controllers has been employed. For each of these plots three guesses of the real, unknown perturbation are also shown. These values have been obtained considering the known additional mass and the input disturbance fixed to zero, to the maximum, and to the minimum. The same analysis and conclusions can be made by observing Figure 3.11, which shows the results for a test with 1 kg payload. The gains that have been used in this experiment are the following:

$$\omega_n = 15, \;\; \zeta = 0.65, \;\; \mathbf{K}_P = \omega_n^2\,\mathbf{I}_2, \;\; \mathbf{K}_D = 2\,\zeta\omega_n\,\mathbf{I}_2, \;\; \kappa_P = 2, \;\; \mathbf{Q} = \mathbf{I}_2,$$

where κ_P has been obtained by means of manual tuning. Please note that tuning this parameter is very intuitive since its increase directly makes the tracking error decrease. On the other hand, the increase of this parameter increases the effect of the amplification of the measurement noise on the current commands. The tuning procedure aims at finding a trade-off between tracking precision and noise amplification. A practical way to automate this procedure can be to use a

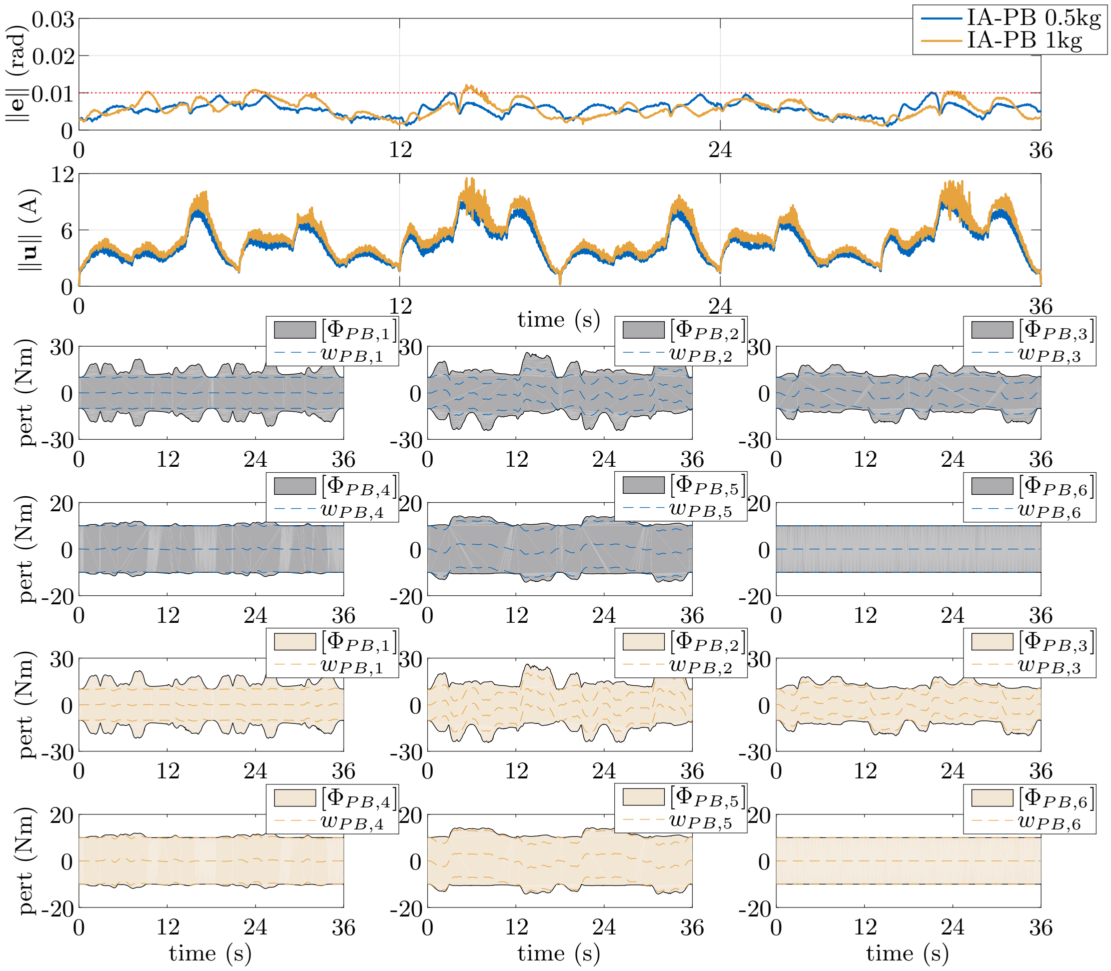


Figure 3.12: Tracking control with the IA-PB controller with 0.5 kg, 1 kg of unmodeled payload, and initial conditions IC_1. The bounds of the perturbation vector computed online using interval arithmetic are shown at the bottom for both cases of the used payloads: 0.5 kg (gray) and 0.5 kg (light-orange).

fast Fourier transform on the current command and to stop increasing the parameter κ_P when excessively high frequency components of these signals start to become non-negligible.

3.6.2 Interval-Arithmetic Passivity-Based Control

The second experiment aims to show the applicability of the IA-PB controller, which can guarantee control with robust stability and robust performance. The same experimental test bed, payloads, and desired trajectory used in the previous experiment are considered. In this

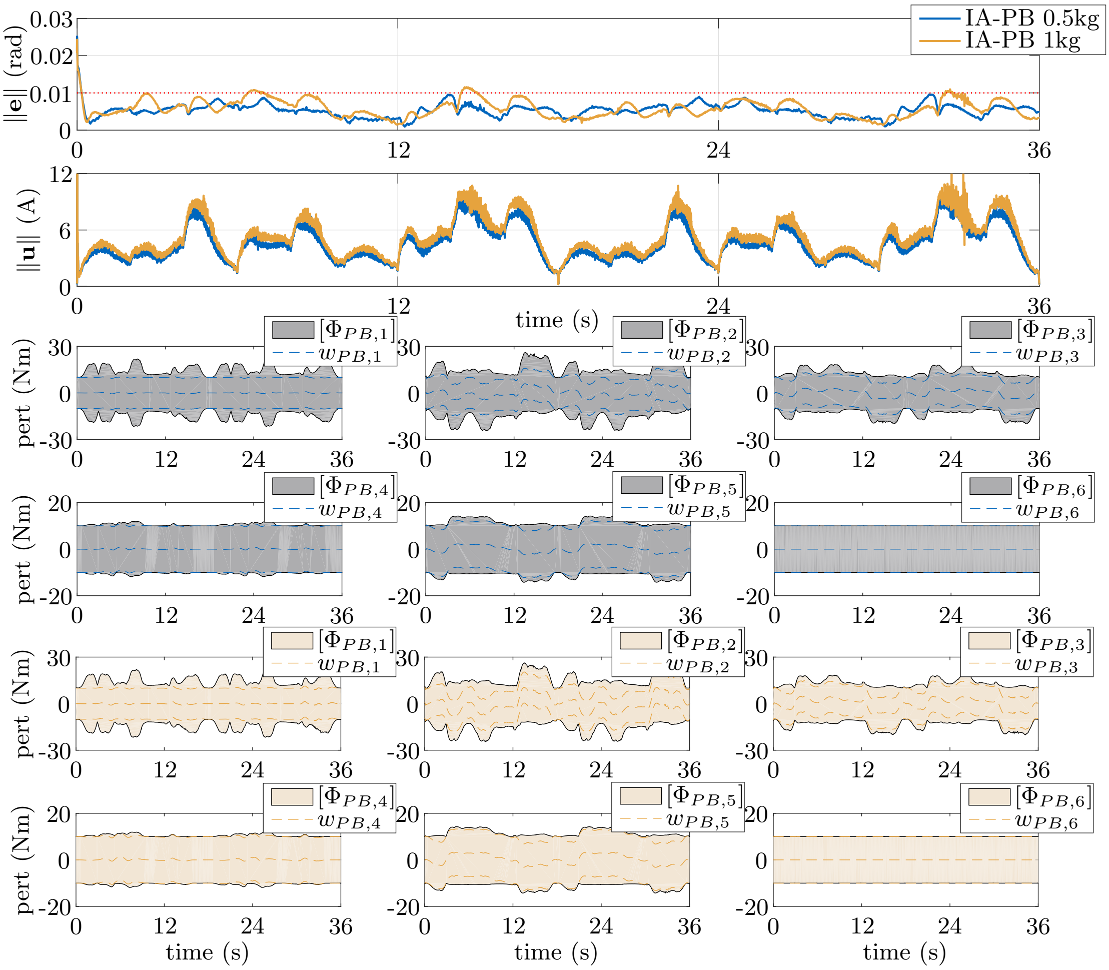


Figure 3.13: Tracking control with the IA-PB controller with $0.5\,kg$, $1\,kg$ of unmodeled payload, and initial conditions IC_2. The bounds of the perturbation vector computed online using interval arithmetic are shown at the bottom for both cases of the used payloads: $0.5\,kg$ (gray) and $0.5\,kg$ (light-orange).

experiment, the following initial conditions (IC_1 and IC_2) have been tested:

$$IC_1 := \begin{cases} \mathbf{q}(0) = (0 \;\; 0)^T \\ \dot{\mathbf{q}}(0) = (0 \;\; 0)^T, \end{cases} \qquad IC_2 := \begin{cases} \mathbf{q}(0) = 2\,\epsilon\,(1 \quad -1)^T \\ \dot{\mathbf{q}}(0) = (0 \;\; 0)^T. \end{cases}$$

The desired tracking performance is set to $\epsilon = 10^{-2}$.

Figure 3.12 and Figure 3.13 show the results for IC_1 and IC_2, respectively. For both cases of initial conditions and payloads, the controller shows its ability to approach the desired robust performance as expected. The IA-PB controller has been shown to perform significantly better with respect to the inverse-dynamics variant. It allows one to obtain better performance, given comparable amplification of the measurement noise affecting the current commands. The bounds of the perturbation vector computed online are shown at the bottom of Figures 3.12 and 3.13. The same approach has been followed for presenting the guesses of the real unknown perturbation, which is included in the bounds. The approach described in Subsection 3.4.2 has also been employed here in order to compute the control commands and no significant excess of conservativity is shown. It is important to mention that particular attention should be paid when deploying such a robust performance control scheme. In particular, this is due to the possible increase of the functions $\kappa(t)$ and $\varphi(t)$ to values that may excessively amplify the measurement noise. The impact of measurement noise has not been considered in the theoretical derivation of this control scheme, but it is important to consider for its practical implementation. In fact, as can be observed from Figure 3.12 and Figure 3.13, the integral action in $\kappa(t)$ and $\varphi(t)$ leads to an increase of the high-frequency components in the control commands (observable at about 32 seconds in the plot), which is due to the amplification of the measurement noise. This effect may become a problem in practice since it introduces the risk of chattering. Figure 3.14 shows the mentioned problem, in which a large error in the initial conditions is tested, setting them as ten times those of IC_2. In this figure, the performances of the IA-PB controller with naively implemented integral action (orange) and with deactivated integral action ($\kappa_I = \varphi_I = 0$, blue) are shown. As mentioned previously for the inverse-dynamics variant, a possible practical solution can be to include a fast-Fourier transform block for the control commands signals that stops the integration when excessively high-frequency components start becoming non-negligible. The integration should also be stopped before the error norm enters the desired bounds for the first time, which happens for initial conditions that let the tracking error be outside of the desired performance bounds at the start. Clearly, stopping the integral action implies that excessively high robust performance may not be achievable anymore, since the physical limits of the system would be reached. On the other hand, if the designer does not want to pay particular attention to the discussed integral action, the safest option is to deactivate it and settle for the provided robust stability. In this case, the tuning parameters κ_P, φ_P, and $\mathbf{K}_r$ can be selected intuitively, keeping in mind that their increase directly increases the tracking performance. Finally, it is worth reporting that the maximum total execution time required for computing the commands of the interval-arithmetic-based robust controllers is about $2 \cdot 10^{-5}\,s$ [126].

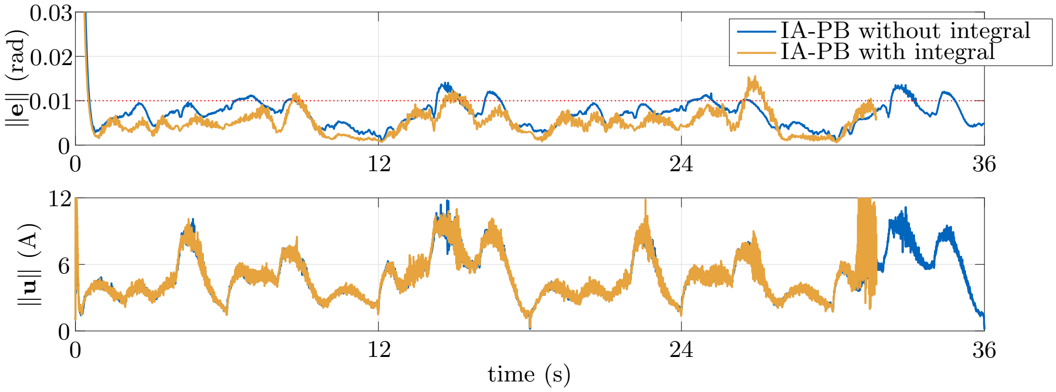


Figure 3.14: Tracking control test to show the possible phenomenon of chattering that may appear when no particular attention is paid to stopping the integral actions in $\kappa(t)$ and $\varphi(t)$. The test has been executed with a $1\,kg$ unmodeled payload and significantly wrong initial conditions (set at ten times the ones in IC_2). In this case, the test with the naively implemented integral action had to be stopped due to severe chattering.

3.7 Summary

In this chapter, control schemes that are robust to model uncertainties and external disturbances have been presented. After a literature review, a novel approach for robust control of robots was given. This method allows one to automatically deploy robust controllers which are continuous and do not need the estimation of state-dependent perturbation bounding functions that other methods require. This has been shown to be possible by exploiting interval arithmetic for automatic online computation of closed-loop perturbations. The over-approximative estimates obtained with interval arithmetic are directly used as a feedback for closed-loop control. This practice has been shown both theoretically and experimentally to provide control with ultimate robust stability and performance. This approach is especially useful for modular robot manipulators, which require quickly deployable controllers, as well as for classical rigid robots since the robust control implementation process is simplified.

The inclusion of interval arithmetic for automatically deploying a robust controller has been applied to the case in which the inverse-dynamics control scheme is used as a nominal controller and also for the case in which a passivity-based controller is used. The real world experiments have shown that the former can provide ultimate robust stability while the latter can provide ultimate robust performance. Simulation and real experiments additionally show that these schemes outperform other existing methods, as they are less conservative. The use of a recently proposed interval-arithmetic-based Newton-Euler algorithm for efficient numerical computation of the robust controllers ensures online computability.

Even though the proposed interval-arithmetic-based controllers are continuous, the noise and the limited sampling rate of real applications can introduce chattering effects for excessively high required performance. The issue may be solved by carefully defining the behavior of the tuning functions $\kappa(t)$ and $\varphi(t)$, which could be driven by the power spectrum of the control signal, for example. A proper behavior of these functions would ideally allow sacrificing performance when necessary to avoid chattering. This aspect surely represents an interesting direction to investigate for future research, together with the application of this approach to other nonlinear systems with uncertain dynamics.

Chapter 4

Control Design with Elastic-Joint Modules

This chapter focuses on the control design of flexible robot arms which enjoy an increasing popularity. In particular, the focus is on the case where the elasticity is introduced by the joints, e.g., by means of series elastic actuators. State-of-the-art approaches for addressing this control problem are first reviewed. Special attention is paid to their applicability and quick deployability for controlling modular reconfigurable robots. As in the previous chapter, inverse-dynamics (ID) and passivity-based (PB) tracking control are reviewed in more detail. Then, the author's recently proposed method [128, 129] is presented. This approach combines mutually exclusive benefits from the ID and PB control schemes: the efficient numerical computability of the former and the intrinsic robustness of the latter. Sections dedicated to a detailed robustness analysis, simulation, and experimental results complete this chapter. This chapter is largely based on the author's work in [128, 129].

4.1 Introduction and State of the Art

The intentional inclusion of elasticity in the structure of robots has gained large popularity in recent decades. This is often realized by using compliant components, e.g., elastic links, series elastic actuators, and variable stiffness actuators. With the use of these compliant structural elements, the aim is to obtain robots that are safe and resilient for tasks that may involve desired or undesired contact with the surrounding environment or humans [109]. Another collateral benefit provided by this modern practice is the possibility to exploit the intrinsically rich resulting dynamics, e.g., highly dynamic movements that were not possible with classical rigid structures [46].

Possible scenarios in which elasticity in the robotic structure is not negligible are elastic links, elastic joints, the mix of the two, and the mix of these cases with rigid components as well. For all these cases, specialized control algorithms should be deployed. Otherwise, when no particular attention is paid to the elasticity introduced by compliant components, the motion-control performance may be negatively affected, and undesired oscillatory behaviors or even instabilities may appear. Hereafter, the focus is narrowed to the case that the elasticity is introduced by elastic joints. The modular reconfigurable robotic scenario considered so far can thus be extended to the case of having joint modules equipped with series elastic actuators. As described in Section 4.2, this case can generate robot arms fully or partially composed of elastic joints. This second case can be approached as if the robot were fully composed of elastic joints as described in [31], provided that the missing motor-side dynamics of the rigid joints are virtually realized by the controller.

The control problem of elastic-joint robots has attracted numerous researchers since the '80s. Fundamental classical methods can be found e.g., in [39, 100], and more modern approaches in [33, 84]. A usual distinction that can be made is between the control schemes that provide regulation control of the joint positions to a constant desired value, and those that provide tracking of time-varying trajectories. Since the latter are usually complex control solutions, the former can be a very practical choice to keep the controller complexity low when high tracking precision is not required.

4.1.1 Regulation and Tracking

A simple regulator based on the use of a Proportional-Derivative (PD) and gravity compensation action has been proposed in [107]. This scheme uses feedback of motor-side position and velocity, and includes the gravity compensation term as a constant evaluated at the desired reference position. The author of [107] also provides a robustness analysis with respect to uncertain model parameters and evaluates the control performance using simulations. An enhancement of the previously mentioned PD regulator has been proposed in [38], where the authors introduce the use of an online-computed term for gravity compensation using a gravity-biased estimation of the motor-side position, still using only motor-side feedback variables. This results in more flexibility during the tuning phase and allows one to achieve better transient behavior. Exact cancellation of gravity is obtained by employing the scheme proposed in [35], which is more complex and requires readings of joint positions and velocities both from motor-side and link-side. A class of passivity-based regulators is defined in [80], in which the requirement of joint velocity measurements is eliminated. Regulators that do not rely on limited sensing capabilities but exploit the full state feedback, e.g., [2,3,83], can provide better damping of residual link-side oscillations.

The more complex control problem of the precise tracking of time-varying trajectories has also received significant attention from the research community. Without making any simplifying assumptions on the elastic-joint robot model, dealing with tracking control becomes very complex. For such a scenario one can resort to the approach proposed in [70], where the authors introduce a method that uses a dynamic feedback approach to obtain input-output decoupling and linearization. However, many existing tracking controllers rely on the popular simplifying "Spong's assumptions", introduced in [105]. In that work it is first assumed that the contributions to the kinetic energy of each rotor are negligible, except those due to its own rotation only. The second assumption made is about the symmetric distribution of the mass of the rotor/gears about its axis of rotation. These assumptions are very reasonable in practice, especially for the large class of actuators with high gear ratios.

With Spong's assumptions, the tracking control problem simplifies significantly. They lead to a reduced model that enables the realization of static feedback linearizing controllers (inverse-dynamics) as well as the development of controllers based on the singular perturbation approach [103]. The latter schemes rely on the time-scale separation between the motor-side (fast) and link-side (slow) dynamics, for sufficiently weak joint elasticity. An experimental comparison of the feedback linearization and the singular perturbation technique for controlling a single link elastic joint arm can be found in [101]. From this study it emerges that the approach based on singular perturbation is effective for weakly elastic joints. Instead, one should resort to the more complex inverse-dynamics scheme to directly cancel modeled nonlinearities for highly flexible joints.

An effective, practical solution to the tracking control problem is to use a simple PD feedback control law controlling motor-side variables and a feedforward action as proposed in [32]. This scheme ensures local stability only, but has the benefit of using only motor-side feedback variables. This approach can also be enhanced by including torque readings as in [3]. Please note that joint position and velocity readings on the motor side are typically less noisy with respect to the readings on the link side, given the same encoder. This is due to the presence of the suppression of noise by the gear ratio. A scheme that provides semiglobal tracking is presented in [69], where the control law has the merit of not requiring feedback of the link-side jerks, which is instead common for global trackers.

A description of pioneering global tracking controllers for elastic joint robots can be found in [19], where schemes based on decoupling, backstepping, and passivity are compared. In that work, an effective static passivity-based tracking controller is also presented. Among more recent global trackers are the schemes in [7, 8, 68], where the authors consider lossless systems and interestingly achieve uniform global stability without requiring measurements of joint velocities. Additionally, recent passivity-based tracking controllers that provide global stability and damping assignment with nonlinear elastic-joints can be found in [58, 59].

4.1.2 Efficient Implementability and Robustness

Among the above mentioned global tracking schemes, the inverse-dynamics approach is particularly interesting thanks to specialized recursive algorithms that have been recently proposed for its efficient numerical computation in [20, 21, 77]. However, inverse-dynamics controllers rely on the perfect cancellation of nonlinear coupling terms of the system dynamics and may be sensitive to even small model uncertainties. Besides similar arguments by other authors, e.g., in [19, 105], this issue can be observed from the simulation results in [128]. Instead, it is widely acknowledged that approaches which avoid complete feedback linearization, such as passivity-based schemes, can be more robust [18, 34, 81]. On the other hand, most passivity-based global trackers require high-order derivatives of the model terms which prevents their efficient numerical computability, as it is available instead for the inverse-dynamics schemes.

An approach that allows one to combine the typical enhanced robustness of passivity-based controllers with the efficient numerical computability of inverse-dynamics control schemes using modern recursive algorithms has been introduced in the author's work in [127]. The proposed scheme is a combined Inverse-Dynamics/Passivity-Based (ID/PB) controller, based on the partial cancellation of modeled system dynamics and the exploitation of passivity-related properties of the robot model to ensure stability. This approach has been shown to be effective in simulation and real experiments. Further, an extension of this method has been developed which exploits a feedforward inverse-dynamics action. A detailed robustness analysis and additional experimental results are included in the author's work [129].

4.2 Formulation of the Control Problem

The control problem can be formulated by considering a robot manipulator composed of N links serially connected through elastic joints. By adopting the popular "Spong's assumptions" [99], the following model can be used for describing the system dynamics:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{K}(\mathbf{q} - \boldsymbol{\theta}) = \boldsymbol{\tau}_{ext}, \tag{4.1a}$$

$$\mathbf{J}\ddot{\boldsymbol{\theta}} + \mathbf{f}(\dot{\boldsymbol{\theta}}) + \underbrace{\mathbf{K}(\boldsymbol{\theta} - \mathbf{q})}_{:=\boldsymbol{\tau}_e} = \mathbf{u}. \tag{4.1b}$$

In the relations above, (4.1a) models the link-side dynamics and (4.1b) models the motor-side dynamics. Further, $\mathbf{q} \in \mathbb{R}^N$ is the vector of link-side joint position variables, $\boldsymbol{\theta} \in \mathbb{R}^N$ is the vector of motor-side joint position variables, $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{N\times N}$ is the inertia matrix of the rigid links assembly, $\mathbf{J} \in \mathbb{R}^{N\times N}$ is the constant diagonal matrix of the rotor inertia moments through the square of the respective gear ratios, $\mathbf{K} \in \mathbb{R}^{N\times N}$ is the diagonal joint stiffness matrix, $\mathbf{u} \in \mathbb{R}^N$ is the vector of control input forces/torques, $\mathbf{f}(\dot{\boldsymbol{\theta}}) \in \mathbb{R}^N$ is the vector of the motor-side friction terms, $\boldsymbol{\tau}_e \in \mathbb{R}^N$ is the elastic torque vector, and $\boldsymbol{\tau}_{ext} \in \mathbb{R}^N$ is a vector of external forces applied to the manipulator mapped onto the joint space. The centrifugal, Coriolis, gravitational, and link side friction contributions are collected in the following term:

$$\mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{D}\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}), \tag{4.2}$$

where $\mathbf{g}(\mathbf{q}) \in \mathbb{R}^N$ is the vector of the gravity terms, $\mathbf{D} \in \mathbb{R}^{N\times N}$ is the matrix of the link-side viscous damping coefficients, and $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\,\dot{\mathbf{q}} \in \mathbb{R}^N$ is the vector of Coriolis and centrifugal terms, with $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^{N\times N}$ being a matrix such that the property in (2.2) holds. Please note that the model in (4.1) is not able to properly describe the dynamics of an assembled arm with mixed rigid/elastic joints. However, this model does not limit generality since any mixed rigid/elastic joint robot can be modeled by (4.1) provided that missing motor-side dynamics are virtually realized (see e.g., [31]).

The control problems considered in this chapter are different if the aim is simple regulation or if it is trajectory tracking. Regulators should be able to stabilize the robot at a specific constant link-side desired pose. Trackers instead should be able to provide global convergence to zero of link-side joint-space error ($\mathbf{e}(t) \to \mathbf{0}$ for $t \to \infty$) for desired trajectories that are sufficiently smooth (in this chapter at least four times differentiable). However, in the event that uncertainty in the assembled robot description is non-negligible, a more practical stability target is considered. In this case, the design aims at providing ultimate boundedness of the error trajectories:

$$\|\mathbf{e}(t)\| < \epsilon, \ \ \forall\, t \geq t_1,$$

for a finite positive ϵ and a finite time t_1.

With reference to the model in (4.1), no external forces are assumed in the following (i.e., $\boldsymbol{\tau}_{ext} = \mathbf{0}$), except in Section 4.6, where robustness with respect to external disturbances and model mismatches is considered.

4.3 Regulation

When simplicity of the control-loop is desired and tracking performance is not of primary importance, simple regulators are deployed. One of the simplest approaches is to use the following control law for the system in (4.1):

$$\mathbf{u} = \mathbf{K}_P(\boldsymbol{\theta}_d - \boldsymbol{\theta}) - \mathbf{K}_D\dot{\boldsymbol{\theta}} + \mathbf{g}(\mathbf{q}_d), \tag{4.3}$$

where $\mathbf{K}_P$ and $\mathbf{K}_D$ are symmetric positive-definite matrices of proper dimensions and

$$\boldsymbol{\theta}_d = \mathbf{q}_d + \mathbf{K}^{-1}\mathbf{g}(\mathbf{q}_d).$$

This scheme has been introduced in [107], where authors show that global asymptotic stability of the equilibrium state $\mathbf{q} = \mathbf{q}_d$, $\boldsymbol{\theta} = \boldsymbol{\theta}_d$ and $\dot{\mathbf{q}} = \dot{\boldsymbol{\theta}} = \mathbf{0}$ is provided if the following condition is satisfied:

$$\lambda_{\min}\left(\widehat{\mathbf{K}}\right) > \left\|\frac{\partial \mathbf{g}(\mathbf{q})}{\partial \mathbf{q}}\right\| \quad \forall \mathbf{q}, \quad \widehat{\mathbf{K}} = \begin{pmatrix} \mathbf{K} & -\mathbf{K} \\ -\mathbf{K} & \mathbf{K} + \mathbf{K}_P \end{pmatrix}. \tag{4.4}$$

As proposed in [38], an improvement of this scheme providing better transient behavior can be implemented with the following control law:

$$\mathbf{u} = \mathbf{K}_P(\boldsymbol{\theta}_d - \boldsymbol{\theta}) - \mathbf{K}_D\dot{\boldsymbol{\theta}} + \mathbf{g}(\tilde{\boldsymbol{\theta}}), \tag{4.5}$$

where

$$\tilde{\boldsymbol{\theta}} = \boldsymbol{\theta} + \mathbf{K}^{-1}\mathbf{g}(\mathbf{q}_d).$$

Differently from (4.3), the gravity compensation term in (4.5) is computed using the measured motor-side position in $\tilde{\boldsymbol{\theta}}$. Under the condition in (4.4), this controller also guarantees global asymptotic stability of the equilibrium state $\mathbf{q} = \mathbf{q}_d$, $\boldsymbol{\theta} = \boldsymbol{\theta}_d$ and $\dot{\mathbf{q}} = \dot{\boldsymbol{\theta}} = \mathbf{0}$, as shown in [38].

These regulators are interesting due to their implementation simplicity. In fact, they can be simply deployed once the gravity term of the dynamical model and the joint stiffness coefficients are available. The suitability of these laws to the case of a modular reconfigurable arm with elastic joints is immediately apparent when considering the results presented in Chapter 2. In fact, with that framework the gravity compensation term can be automatically obtained for arbitrary assemblies, based on module data (see Chapter 2). In this case, the module data for joint modules should be augmented to include the values of the joint stiffness. For a quick automatic deployment of the control law, without the use of symbolic toolboxes for computing $\frac{\partial \mathbf{g}(\mathbf{q})}{\partial \mathbf{q}}$ analytically, manual tuning is required in practice to increase $\mathbf{K}_P$ until a satisfactory closed-loop behavior is obtained. It is worth mentioning that the more advanced regulator in [35] realizes exact cancellation of gravity and the tuning is simpler since there is no need to fulfill the condition in (4.4). However, the significant increase in complexity, which becomes similar to the case of trajectory tracking controllers, makes it rather unattractive for regulation purposes with respect to the simple schemes described above.

4.4 Tracking

In this section, effective methods that can be employed for solving the trajectory-tracking control problem of elastic joint robots are described: feedforward/feedback, inverse-dynamics, passivity-based, and combined inverse-dynamics/passivity-based trackers. The first three schemes have been respectively proposed in [19,32,105]. Their presentation is a precursor to the author's most recent results [127,129]. While the first scheme provides local stability, the others guarantee it globally. The applicability of these schemes for controlling modular robot manipulators will be described in the next section, where the efficient implementability of the controllers is discussed.

Hereafter, the mathematical description is kept succinct by omitting the time dependence of time-varying variables and the variable dependencies of the model terms. For example, the matrix $\mathbf{M}\big(\mathbf{q}(t)\big)$ is simply denoted by $\mathbf{M}$, its first time derivative $\dot{\mathbf{M}}\big(\mathbf{q}(t), \dot{\mathbf{q}}(t)\big)$ by $\dot{\mathbf{M}}$, and its second time derivative $\ddot{\mathbf{M}}\big(\mathbf{q}(t), \dot{\mathbf{q}}(t), \ddot{\mathbf{q}}(t)\big)$ by $\ddot{\mathbf{M}}$. The subscript d is used for the model terms that are evaluated along the desired trajectory. For example, we denote the model term $\mathbf{M}(\mathbf{q}_d(t))$ simply by $\mathbf{M}_d$, $\dot{\mathbf{M}}(\mathbf{q}_d(t), \dot{\mathbf{q}}_d(t))$ by $\dot{\mathbf{M}}_d$, and $\ddot{\mathbf{M}}(\mathbf{q}_d(t), \dot{\mathbf{q}}_d(t), \ddot{\mathbf{q}}_d(t))$ by $\ddot{\mathbf{M}}_d$. Further, the derivatives $\frac{d^n}{dt^n}\mathbf{x}(t)$ with $n \in \{3, 4\}$ are denoted by $\mathbf{x}^{[n]}$. The same notation is employed for the other model terms.

4.4.1 Feedforward/Feedback Tracking Control

A very practical tracking controller can be implemented by using a PD control action on motor-side feedback variables, enhanced by a feedforward term to approximately cancel high-order nonlinear couplings as proposed in [32]. For the system in (4.1), this control law is implemented as follows:

$$\mathbf{u} = \mathbf{u}_d + \mathbf{K}_P\big(\boldsymbol{\theta}_d - \boldsymbol{\theta}\big) + \mathbf{K}_D\left(\dot{\boldsymbol{\theta}}_d - \dot{\boldsymbol{\theta}}\right). \tag{4.6}$$

In the equation above, $\mathbf{u}_d$ is the feedforward term that can be obtained by evaluating (4.1b) along the desired trajectory, which yields

$$\mathbf{u}_d = \mathbf{J}\ddot{\boldsymbol{\theta}}_d + \mathbf{f}(\dot{\boldsymbol{\theta}}_d) + \mathbf{M}_d\ddot{\mathbf{q}}_d + \mathbf{h}_d. \tag{4.7}$$

In both (4.6) and (4.7) the presence of the motor-side desired trajectories can be noticed. Since the user usually defines the desired trajectory on the link-side variables, motor-side quantities must be properly computed. The required relations can be obtained by rearranging (4.1a), differentiating it once and twice with respect to time, and by evaluating the resulting relations along the desired link-side trajectory $\mathbf{q}_d$, $\dot{\mathbf{q}}_d$, $\ddot{\mathbf{q}}_d$, $\mathbf{q}_d^{[3]}$, and $\mathbf{q}_d^{[4]}$. With this approach, the terms $\boldsymbol{\theta}_d$, $\dot{\boldsymbol{\theta}}_d$, and $\ddot{\boldsymbol{\theta}}_d$, can be computed respectively as

$$\boldsymbol{\theta}_d = \mathbf{q}_d + \mathbf{K}^{-1}\big(\mathbf{M}_d\ddot{\mathbf{q}}_d + \mathbf{h}_d\big), \tag{4.8}$$

$$\dot{\boldsymbol{\theta}}_d = \dot{\mathbf{q}}_d + \mathbf{K}^{-1}\big(\mathbf{M}_d\mathbf{q}_d^{[3]} + \dot{\mathbf{M}}_d\ddot{\mathbf{q}}_d + \dot{\mathbf{h}}_d\big), \tag{4.9}$$

$$\ddot{\boldsymbol{\theta}}_d = \ddot{\mathbf{q}}_d + \mathbf{K}^{-1}\big(\mathbf{M}_d\mathbf{q}_d^{[4]} + 2\,\dot{\mathbf{M}}_d\mathbf{q}_d^{[3]} + \ddot{\mathbf{M}}_d\ddot{\mathbf{q}}_d + \ddot{\mathbf{h}}_d\big). \tag{4.10}$$

4.4.2 Inverse-Dynamics Tracking Control

Similar to the inverse-dynamics control of rigid robots, the version for elastic-joint arms also aims at canceling nonlinear and coupling terms through static feedback. This control scheme was first introduced in [99] and can be implemented using the following control command:

$$\mathbf{u}_{ID} = \mathbf{J}\mathbf{K}^{-1}\left[\mathbf{M}\,\mathbf{y} + 2\,\dot{\mathbf{M}}\,\mathbf{q}^{[3]} + \ddot{\mathbf{M}}\,\ddot{\mathbf{q}} + \ddot{\mathbf{h}}\right] + [\mathbf{M} + \mathbf{J}]\,\ddot{\mathbf{q}} + \mathbf{h} + \mathbf{f}, \tag{4.11}$$

with

$$\ddot{\mathbf{h}} = (\mathbf{C} + \mathbf{D})\,\mathbf{q}^{[3]} + \ddot{\mathbf{C}}\dot{\mathbf{q}} + 2\,\dot{\mathbf{C}}\ddot{\mathbf{q}} + \ddot{\mathbf{g}}, \tag{4.12}$$

and where $\mathbf{y} \in \mathbb{R}^N$ is an auxiliary control input from which the system can be seen as linear and decoupled. In fact, by rewriting (4.1) as

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{h} = \mathbf{u} - \mathbf{f} - \mathbf{J}\ddot{\boldsymbol{\theta}}, \tag{4.13}$$

and by using (4.11) with

$$\ddot{\boldsymbol{\theta}} = \mathbf{K}^{-1}\left[\mathbf{M}\,\mathbf{q}^{[4]} + 2\,\dot{\mathbf{M}}\,\mathbf{q}^{[3]} + \ddot{\mathbf{M}}\,\ddot{\mathbf{q}} + \ddot{\mathbf{h}}\right] + \ddot{\mathbf{q}}, \tag{4.14}$$

a linear decoupled system is obtained:

$$\mathbf{q}^{[4]} = \mathbf{y}.$$

The inverse-dynamics control law can now be completed by assigning asymptotically stable dynamics of the trajectory tracking error through $\mathbf{y}$ such as

$$\mathbf{y} = \mathbf{q}_d^{[4]} + \mathbf{K}_{ID3}\mathbf{e}^{[3]} + \mathbf{K}_{ID2}\ddot{\mathbf{e}} + \mathbf{K}_{ID1}\,\dot{\mathbf{e}} + \mathbf{K}_{ID0}\,\mathbf{e}, \tag{4.15}$$

where $\mathbf{K}_{IDi}$ with $i \in \{0,1,2,3\}$ are typically diagonal gain matrices with $K_{IDi_{jj}}$ being the jth element of the diagonal, such that

$$id_j(s) := s^4 + K_{ID3_{jj}}\,s^3 + K_{ID2_{jj}}\,s^2 + K_{ID1_{jj}}\,s + K_{ID0_{jj}}, \quad \forall j \in \{1,\dots,N\}, \tag{4.16}$$

are Hurwitz polynomials. From this, exponential stability of the error dynamics follows directly.

Please note that even though the dependence of model terms from the joint variables and their derivatives has been omitted for brevity, (4.11) requires feedback of link-side joint accelerations and jerks. These quantities can be obtained in principle with existing sensing devices such as precise encoders, tachometers, and/or torque sensors. For example, assuming availability of an encoder for the link side and another for the motor side, $\mathbf{q}$ and $\boldsymbol{\theta}$ are directly measured, while $\dot{\mathbf{q}}$ and $\dot{\boldsymbol{\theta}}$ can be obtained by their respective numerical differentiation[1]. Then, the joint accelerations and jerks can be computed using (4.1a) and its first derivative as

$$\ddot{\mathbf{q}} = \mathbf{M}^{-1}\big(\boldsymbol{\tau}_e - \mathbf{h}\big), \tag{4.17}$$

$$\mathbf{q}^{[3]} = \mathbf{M}^{-1}\left(\dot{\boldsymbol{\tau}}_e - \left(\dot{\mathbf{M}}\,\ddot{\mathbf{q}} + \dot{\mathbf{h}}\right)\right), \tag{4.18}$$

where $\boldsymbol{\tau}_e = \mathbf{K}\,(\boldsymbol{\theta} - \mathbf{q})$ and consequently $\dot{\boldsymbol{\tau}}_e = \mathbf{K}\,(\dot{\boldsymbol{\theta}} - \dot{\mathbf{q}})$.

4.4.3 Passivity-Based Tracking Control

A static controller that does not aim at obtaining a linear decoupled system through perfect cancellation of modeled nonlinearities and provides global tracking has been proposed in [19].

[1] The numerical differentiation can be seen as a cheaper, practical solution to the use of tachometers which is very effective in practice, provided that modern precise encoders are employed.

This scheme relies on the passivity-based control approach and can be formulated using the following terms:

$$\overline{\mathbf{M}} = \begin{pmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{J} \end{pmatrix}, \quad \overline{\mathbf{C}} = \begin{pmatrix} \mathbf{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}, \quad \overline{\mathbf{K}} = \begin{pmatrix} \mathbf{K} & -\mathbf{K} \\ -\mathbf{K} & \mathbf{K} \end{pmatrix}, \tag{4.19}$$

$$\overline{\mathbf{u}} = \begin{pmatrix} \mathbf{0} \\ \mathbf{u} \end{pmatrix}, \quad \overline{\mathbf{f}} = \begin{pmatrix} \mathbf{f}_q \\ \mathbf{f}_\theta \end{pmatrix}, \quad \overline{\mathbf{g}} = \begin{pmatrix} \mathbf{g} \\ \mathbf{0} \end{pmatrix}, \tag{4.20}$$

where $\mathbf{f}_q = \mathbf{D}\dot{\mathbf{q}}$ is the link-side friction term (assumed to be only viscous), and $\mathbf{f}_\theta$ is the motor-side friction term which can have both static and viscous components. In addition, the following vectors are introduced:

$$\mathbf{x} = \begin{pmatrix} \mathbf{q} \\ \boldsymbol{\theta} \end{pmatrix}, \quad \mathbf{x}_d = \begin{pmatrix} \mathbf{q}_d \\ \boldsymbol{\theta}_d \end{pmatrix}, \quad \overline{\mathbf{e}} = \mathbf{x}_d - \mathbf{x} = \begin{pmatrix} \mathbf{e}_q \\ \mathbf{e}_\theta \end{pmatrix}, \quad \mathbf{K}_V = \begin{pmatrix} \mathbf{K}_{V,q} & \mathbf{0} \\ \mathbf{0} & \mathbf{K}_{V,\theta} \end{pmatrix}, \tag{4.21}$$

$$\overline{\mathbf{r}} = \dot{\overline{\mathbf{e}}} + \mathbf{K}_V\,\overline{\mathbf{e}} = \begin{pmatrix} \mathbf{r}_q \\ \mathbf{r}_\theta \end{pmatrix}, \quad \dot{\mathbf{x}}_a = \begin{pmatrix} \dot{\mathbf{q}}_a \\ \dot{\boldsymbol{\theta}}_a \end{pmatrix} = \dot{\mathbf{x}}_d + \mathbf{K}_V\,\overline{\mathbf{e}}. \tag{4.22}$$

With the extended vectors and matrices of (4.19)-(4.22), the model in (4.1) can be rewritten as

$$\overline{\mathbf{M}}\,\dot{\overline{\mathbf{r}}} + \overline{\mathbf{C}}\,\overline{\mathbf{r}} + \overline{\boldsymbol{\Lambda}}\,\overline{\mathbf{r}} + \overline{\mathbf{K}}\,\overline{\mathbf{e}} = \overline{\boldsymbol{\psi}}, \tag{4.23}$$

where $\overline{\boldsymbol{\Lambda}} = \begin{pmatrix} \boldsymbol{\Lambda}_q & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Lambda}_\theta \end{pmatrix}$ is a positive definite matrix of proper dimensions and

$$\overline{\boldsymbol{\psi}} = \overline{\mathbf{u}} - \left(\overline{\mathbf{M}}\ddot{\mathbf{x}}_a + \overline{\mathbf{C}}\dot{\mathbf{x}}_a + \overline{\mathbf{K}}\,\mathbf{x}_d + \overline{\mathbf{f}} + \overline{\mathbf{g}}\right) + \overline{\boldsymbol{\Lambda}}\,\overline{\mathbf{r}}.$$

Now, it can be shown by algebraic manipulation that the perturbation $\overline{\boldsymbol{\psi}}$ is set to zero by employing the following static controller

$$\mathbf{u}_{PB} = \mathbf{J}\left(\ddot{\boldsymbol{\theta}}_d + \mathbf{K}_{V,\theta}\left(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{\theta}}_d\right)\right) - \mathbf{K}(\mathbf{q}_d - \boldsymbol{\theta}_d) - \boldsymbol{\Lambda}_\theta\,\mathbf{r}_\theta + \mathbf{f}_\theta, \tag{4.24}$$

with

$$\boldsymbol{\theta}_d = \mathbf{q}_d + \mathbf{K}^{-1}\left(\mathbf{M}\,\ddot{\mathbf{q}}_a + \mathbf{C}\,\dot{\mathbf{q}}_a + \mathbf{D}\dot{\mathbf{q}} + \mathbf{g} - \boldsymbol{\Lambda}_q\,\mathbf{r}_q\right), \tag{4.25}$$

$$\dot{\boldsymbol{\theta}}_d = \dot{\mathbf{q}}_d + \mathbf{K}^{-1}\left(\mathbf{M}\,\mathbf{q}_a^{[3]} + \dot{\mathbf{M}}\,\ddot{\mathbf{q}}_a + \dot{\mathbf{C}}\,\dot{\mathbf{q}}_a + \mathbf{C}\,\ddot{\mathbf{q}}_a + \mathbf{D}\ddot{\mathbf{q}} + \dot{\mathbf{g}} - \boldsymbol{\Lambda}_q\,\dot{\mathbf{r}}_q\right), \tag{4.26}$$

$$\ddot{\boldsymbol{\theta}}_d = \ddot{\mathbf{q}}_d + \mathbf{K}^{-1}\left(\mathbf{M}\,\mathbf{q}_a^{[4]} + 2\,\dot{\mathbf{M}}\,\mathbf{q}_a^{[3]} + \ddot{\mathbf{M}}\,\ddot{\mathbf{q}}_a + \mathbf{C}\,\mathbf{q}_a^{[3]} + \ddot{\mathbf{C}}\,\dot{\mathbf{q}}_a + 2\,\dot{\mathbf{C}}\,\ddot{\mathbf{q}}_a + \mathbf{D}\mathbf{q}^{[3]} + \ddot{\mathbf{g}} - \boldsymbol{\Lambda}_q\,\ddot{\mathbf{r}}_q\right). \tag{4.27}$$

With the above controller, since $\overline{\boldsymbol{\psi}} = \mathbf{0}$ in (4.23) and given that the matrices $\mathbf{K}_{V,q}$, $\mathbf{K}_{V,\theta}$ are chosen to be equal, diagonal, and positive definite, global convergence to zero of the tracking error directly follows as shown in [19], since (4.23) defines an output-strictly-passive mapping (see e.g., [82]) $\overline{\boldsymbol{\psi}} \rightarrow \overline{\mathbf{r}}$ with the storage function

$$V = \frac{1}{2}\overline{\mathbf{r}}^T\overline{\mathbf{M}}\overline{\mathbf{r}} + \frac{1}{2}\overline{\mathbf{e}}^T\overline{\mathbf{K}}\overline{\mathbf{e}}. \tag{4.28}$$

It is important to point out that to compute (4.24) with (4.25), (4.26), and (4.27), high-order derivatives of the model terms arise as for the inverse-dynamics scheme. However, an important difference with respect to the inverse-dynamics scheme is that in this case, the first and second derivatives of the matrices **M** and **C** that fulfill property (2.2) are also required. This significantly complicates the implementation as clarified in section 4.5. Additionally, the controller requires feedback of the link-side accelerations and jerks, which can be obtained with the same approach used for the inverse-dynamics scheme.

4.4.4 Combined Inverse-Dynamics/Passivity-Based Tracking Control

The novel idea of combining the benefits of the inverse-dynamics and the passivity-based control methodologies has been recently proposed by the author in [128]. The aim is to obtain robustness with respect to model mismatches typical of passivity-based controllers, but still maintain efficient numerical computability available today for the inverse-dynamics schemes. The approach that can be adopted for efficient computability is described in detail in Section 4.5, while a detailed robustness analysis of this novel control approach is detailed in Section 4.6. The nominal formulation of the combined inverse-dynamics/passivity-based (ID/PB) tracking controller (i.e., assuming perfect knowledge of the robot model) follows.

The basic idea is to mimic the cancellation of model nonlinearities in a similar way as the inverse-dynamics scheme does, with the difference that not all of them are canceled. In fact, the nonlinear coupling terms that allow the controller to be completed with the passivity-based methodology are left and exploited so that passivity-based control principles can be used to ensure stability and performance. The controller can be formulated by following similar lines of the description of the pure inverse-dynamics version. By using (4.14) with

$$\ddot{\mathbf{h}} = (\mathbf{C} + \mathbf{D})\,\mathbf{q}^{[3]} + \ddot{\mathbf{C}}\dot{\mathbf{q}} + 2\,\dot{\mathbf{C}}\ddot{\mathbf{q}} + \ddot{\mathbf{g}}, \tag{4.29}$$

in (4.13), the following relation is obtained:

$$\mathbf{J}\mathbf{K}^{-1}\big(\mathbf{M}\,\mathbf{q}^{[4]} + 2\,\dot{\mathbf{M}}\,\mathbf{q}^{[3]} + \ddot{\mathbf{M}}\,\ddot{\mathbf{q}} + \ddot{\mathbf{h}}\big) + \big(\mathbf{M} + \mathbf{J}\big)\ddot{\mathbf{q}} + \mathbf{h} + \mathbf{f} = \mathbf{u}. \tag{4.30}$$

The derivation of the combined ID/PB controller can now be seen as a modification of (4.11). In fact, including a new auxiliary vector $\boldsymbol{\nu}$, expanding $\ddot{\mathbf{h}}$ in (4.11) with (4.29), and removing the terms $\mathbf{M}\,\mathbf{y}$ and $\mathbf{C}\mathbf{q}^{[3]}$, yields

$$\mathbf{u}_{IDPB} = \mathbf{J}\mathbf{K}^{-1}\big(2\,\dot{\mathbf{M}}\,\mathbf{q}^{[3]} + \ddot{\mathbf{M}}\,\ddot{\mathbf{q}} + \mathbf{D}\,\mathbf{q}^{[3]} + \ddot{\mathbf{C}}\dot{\mathbf{q}} + 2\,\dot{\mathbf{C}}\ddot{\mathbf{q}} + \ddot{\mathbf{g}}\big) + \big(\mathbf{M} + \mathbf{J}\big)\ddot{\mathbf{q}} + \mathbf{h} + \mathbf{f} + \boldsymbol{\nu}. \quad (4.31)$$

Now, by applying (4.31) to (4.30), the complete cancellation of the overall system dynamics is avoided, and the following relation is obtained:

$$\mathbf{J}\mathbf{K}^{-1}\big(\mathbf{M}\,\mathbf{q}^{[4]} + \mathbf{C}\,\mathbf{q}^{[3]}\big) = \boldsymbol{\nu}. \quad (4.32)$$

From the position of (4.32), the combined ID/PB control scheme can be completed by using

$$\boldsymbol{\nu} = \mathbf{J}\mathbf{K}^{-1}\big(\mathbf{M}\,\mathbf{q}_a^{[4]} + \mathbf{C}\,\mathbf{q}_a^{[3]} + \boldsymbol{\Lambda}\,\mathbf{r}\big), \quad (4.33)$$

where $\boldsymbol{\Lambda}$ is any positive definite matrix of proper dimensions and

$$\begin{aligned} \mathbf{r} &= \mathbf{e}^{[3]} + \mathbf{K}_{HY3}\ddot{\mathbf{e}} + \mathbf{K}_{HY2}\dot{\mathbf{e}} + \mathbf{K}_{HY1}\,\mathbf{e}, \quad (4.34)\\ \mathbf{q}_a^{[3]} &= \mathbf{q}_d^{[3]} + \mathbf{K}_{HY3}\ddot{\mathbf{e}} + \mathbf{K}_{HY2}\dot{\mathbf{e}} + \mathbf{K}_{HY1}\,\mathbf{e},\\ \mathbf{q}_a^{[4]} &= \mathbf{q}_d^{[4]} + \mathbf{K}_{HY3}\mathbf{e}^{[3]} + \mathbf{K}_{HY2}\ddot{\mathbf{e}} + \mathbf{K}_{HY1}\,\dot{\mathbf{e}}, \end{aligned}$$

with $\mathbf{K}_{HY3}$, $\mathbf{K}_{HY2}$, $\mathbf{K}_{HY1}$ being properly chosen gain matrices. By inserting the control law (4.31) with (4.33) into the system (4.30), the following closed-loop relation is obtained:

$$\mathbf{M}\,\dot{\mathbf{r}} + \mathbf{C}\mathbf{r} + \boldsymbol{\Lambda}\,\mathbf{r} = \mathbf{0}. \quad (4.35)$$

Since no perturbation is assumed so far, the right-hand side of (4.35) is zero. This implies that $\mathbf{r}$ will globally and asymptotically converge to zero as theoretically shown in the literature on passivity-based control (see e.g., [82]). The tracking error converges to zero for a proper selection of the diagonal gain matrices $\mathbf{K}_{HY3}$, $\mathbf{K}_{HY2}$, and $\mathbf{K}_{HY1}$. In particular, this is ensured provided that the gains are selected so that the following are Hurwitz polynomials:

$$hy_j(s) := s^3 + K_{HY3_{jj}}\,s^2 + K_{HY2_{jj}}\,s + K_{HY1_{jj}}, \quad \forall j \in \{1, \ldots N\},$$

where $\mathbf{K}_{HYi_{jj}}$ with $i \in \{1,2,3\}$ denote the jth elements of the diagonal of these matrices. This requirement can simply be understood by applying the Laplace transform to each coordinate of (4.34). In fact, by doing so using the Laplace variable s, the transfer function from $r_j(s)$ to $e_j(s)$ can be observed in the following

$$e_j(s) = \frac{1}{s^3 + K_{HY3_{jj}}\,s^2 + K_{HY2_{jj}}\,s + K_{HY1_{jj}}}\,r_j(s), \quad (4.36)$$

which is stable for the gains selected as mentioned above. In addition, for computing (4.31), the feedback of acceleration and jerk vectors is necessary and can be obtained as previously described for the pure inverse-dynamics controller.

Combined Feedforward-ID/PB Control

Even if the model-based approach for obtaining the joint accelerations and jerks is theoretically viable, it can provide poor estimates in practice. In particular, this may be the case when the knowledge of the model is limited, or when noisy and biased torque readings are available. A more robust approach to the estimation of these quantities can be done by using an observer. For example, a practical approach was implemented in [128] where the model knowledge is assumed to be accurate, but the torque readings noisy. When the model knowledge is also poor and systematic errors are observed for the acceleration estimates, one can resort to using a simple kinematic Kalman filter as in [13]. However, the designer should not forget that filtering operations applied online introduce an estimation error/delay that endangers stability and deteriorates the global tracking properties.

The combined ID/PB controller is based on the partial cancellation of nonlinear coupling terms through feedback, and these terms depend on the estimates of accelerations and jerks. Errors in their estimation produce a perturbation which affects the closed-loop and the control performance directly. A practical improvement in such cases has been introduced in the author's work [129], in which a feedforward action for the inverse-dynamics part of the combined scheme is used. This provides the following combined feedforward-ID/PB scheme:

$$\mathbf{u}_{ffwdIDPB} = \mathbf{u}_{d,ID}(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d, \mathbf{q}_d^{[3]}) + \mathbf{f}(\dot{\boldsymbol{\theta}}) + \mathbf{J}\mathbf{K}^{-1}\big(\mathbf{M}(\mathbf{q})\,\mathbf{q}_a^{[4]} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\,\mathbf{q}_a^{[3]} + \boldsymbol{\Lambda}\,\mathbf{r}\big), \quad (4.37)$$

where

$$\begin{aligned}\mathbf{u}_{d,ID}(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d, \mathbf{q}_d^{[3]}) = \mathbf{J}\mathbf{K}^{-1}\Big(&2\,\dot{\mathbf{M}}_d\,\mathbf{q}_d^{[3]} + \ddot{\mathbf{M}}_d\,\ddot{\mathbf{q}}_d + \mathbf{D}\,\mathbf{q}_d^{[3]} \\ &+ \ddot{\mathbf{C}}_d\dot{\mathbf{q}}_d + 2\,\dot{\mathbf{C}}_d\ddot{\mathbf{q}}_d + \ddot{\mathbf{g}}_d\Big) + \big(\mathbf{M}_d + \mathbf{J}\big)\ddot{\mathbf{q}}_d + \mathbf{h}_d.\end{aligned}$$

Even though global tracking properties are lost, this control scheme has been shown to be very effective in practice. The effectiveness of this variant of the combined ID/PB controller has been verified through experiments, which are presented in Section 4.8.

4.5 Efficient Computation of the Control Commands

An important aspect to consider for modern robot manipulators that have many degrees of freedom (typically 6 or more) is the computational complexity required for implementing and evaluating the above presented controllers. This is particularly critical for modular robot manipulators requiring automatic deployment of the control after their assembly. Possible computational threats are immediately apparent when considering that the presented global tracking controllers require not only the standard link-side model terms, but also their first and second derivatives.

Starting from the fundamental work in [71] and the more recent results in [17], the authors in [20, 21] present a novel algorithm for efficient computation of inverse-dynamics tracking controllers for elastic-joint robots. This algorithm is denoted by the Elastic Joint Newton-Euler Algorithm (EJNEA) and takes as input the vectors of joint positions, velocities, accelerations, jerks, and snaps, to provide the following relation:

$$\mathrm{EJNEA}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}},\mathbf{q}^{[3]},\mathbf{q}^{[4]}) = \mathbf{J}\mathbf{K}^{-1}\left(\mathbf{M}\,\mathbf{q}^{[4]} + 2\,\dot{\mathbf{M}}\,\mathbf{q}^{[3]} + \ddot{\mathbf{M}}\,\ddot{\mathbf{q}} + \ddot{\mathbf{h}}\right) + [\mathbf{M}+\mathbf{J}]\,\ddot{\mathbf{q}} + \mathbf{h} + \mathbf{f}.$$

From this algorithm it is trivial to extract reduced versions (NEA′ and NEA″) which provide

$$\begin{aligned}\mathrm{NEA}'(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}},\mathbf{q}^{[3]}) &= \mathbf{M}\mathbf{q}^{[3]} + \dot{\mathbf{M}}\ddot{\mathbf{q}} + \dot{\mathbf{h}},\\ \mathrm{NEA}''(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}},\mathbf{q}^{[3]},\mathbf{q}^{[4]}) &= \mathbf{M}\,\mathbf{q}^{[4]} + 2\,\dot{\mathbf{M}}\,\mathbf{q}^{[3]} + \ddot{\mathbf{M}}\,\ddot{\mathbf{q}} + \ddot{\mathbf{h}}.\end{aligned}$$

With these three algorithms and the standard recursive Newton-Euler one (for *NEA*, see also Appendix A.1) at hand, the feedforward/feedback controller presented in Subsection 4.4.1 and the inverse-dynamics scheme described in subsection 4.4.2 can be efficiently computed. In particular, the feedforward/feedback tracking controller can be simply implemented using

$$\mathbf{u}_{ffwd/fbk} = \mathbf{u}_d + \mathbf{K}_P\left(\boldsymbol{\theta}_d - \boldsymbol{\theta}\right) + \mathbf{K}_D\left(\dot{\boldsymbol{\theta}}_d - \dot{\boldsymbol{\theta}}\right),$$

where

$$\mathbf{u}_d = \mathbf{J}\ddot{\boldsymbol{\theta}}_d + \mathbf{f}(\dot{\boldsymbol{\theta}}_d) + \mathrm{NEA}(\mathbf{q}_d,\dot{\mathbf{q}}_d,\ddot{\mathbf{q}}_d),$$

and

$$\begin{aligned}\boldsymbol{\theta}_d &= \mathbf{q}_d + \mathbf{K}^{-1}\mathrm{NEA}(\mathbf{q}_d,\dot{\mathbf{q}}_d,\ddot{\mathbf{q}}_d),\\ \dot{\boldsymbol{\theta}}_d &= \dot{\mathbf{q}}_d + \mathbf{K}^{-1}\mathrm{NEA}'(\mathbf{q}_d,\dot{\mathbf{q}}_d,\ddot{\mathbf{q}}_d,\mathbf{q}_d^{[3]}),\\ \ddot{\boldsymbol{\theta}}_d &= \ddot{\mathbf{q}}_d + \mathbf{K}^{-1}\mathrm{NEA}''(\mathbf{q}_d,\dot{\mathbf{q}}_d,\ddot{\mathbf{q}}_d,\mathbf{q}_d^{[3]},\mathbf{q}_d^{[4]}).\end{aligned}$$

Concerning the inverse-dynamics controller, its efficient implementation is straightforward when the EJNEA is available. In fact, with this algorithm at hand, the inverse-dynamics tracking controller can be numerically computed online with

$$\mathbf{u}_{ID} = \text{EJNEA}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbf{q}^{[3]}, \mathbf{y}),$$

where $\mathbf{y}$ is computed as in (4.15). Please note that $\ddot{\mathbf{q}}$ and $\mathbf{q}^{[3]}$ can be also computed efficiently from (4.38) and (4.39) as follows:

$$\ddot{\mathbf{q}} = \mathbf{M}^{-1}\big(\boldsymbol{\tau}_e - \text{NEA}(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{0})\big), \tag{4.38}$$

$$\mathbf{q}^{[3]} = \mathbf{M}^{-1}\left(\dot{\boldsymbol{\tau}}_e - \text{NEA}'(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbf{0})\right). \tag{4.39}$$

The computation of the mass matrix can be done by extracting each column singularly, or by using a specialized algorithm as described in further detail in [20, Section IV].

Contrary to many other passivity-based controllers, the combined ID/PB scheme presented in Subsection 4.4.4 can be efficiently implemented in a similar way as inverse-dynamics control schemes can. This can be done by directly using the EJNEA and the modified recursive Newton-Euler algorithm with neglected gravity (for NEA_0^*, see Appendix A.1) as follows:

$$\mathbf{u}_{IDPB} = \text{EJNEA}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbf{q}^{[3]}, \mathbf{0}) + \mathbf{JK}^{-1}\big(\text{NEA}_0^*(\mathbf{q}, \dot{\mathbf{q}}, \underbrace{\mathbf{q}_a^{[3]} - \mathbf{q}^{[3]}}_{=\mathbf{r}}, \mathbf{q}_a^{[4]}) + \boldsymbol{\Lambda}\mathbf{r}\big) + \mathbf{f}(\dot{\boldsymbol{\theta}}). \tag{4.40}$$

Consequently, the combined feedforward-ID/PB variant is simply computed as

$$\mathbf{u}_{ffwdIDPB} = \text{EJNEA}(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d, \mathbf{q}_d^{[3]}, \mathbf{0}) + \mathbf{JK}^{-1}\big(\text{NEA}_0^*(\mathbf{q}, \dot{\mathbf{q}}, \underbrace{\mathbf{q}_a^{[3]} - \mathbf{q}^{[3]}}_{=\mathbf{r}}, \mathbf{q}_a^{[4]}) + \boldsymbol{\Lambda}\mathbf{r}\big) + \mathbf{f}(\dot{\boldsymbol{\theta}}). \tag{4.41}$$

Once the accelerations and jerks are efficiently computed, the algorithms mentioned in this section can be used online and have linear computational complexity. The combined inverse-dynamics/passivity-based control laws clearly inherit these features in light of (4.40) and (4.41). On the other hand, many other globally stable passivity-based trackers (e.g., [19, 58]) cannot be implemented in the same computationally efficient way. This limitation is due to the required matrices $\mathbf{M}$ and $\mathbf{C}$ and their first and second time derivatives. Instead, with the efficiently implementable schemes discussed above, such contributions are already included in the numerical computations of the EJNEA.

All the algorithms mentioned in this section require as input the kinematic description and the dynamical parameters (e.g., **DH** and **DynPar** with reference to Chapter 2). Their presence as arguments of the algorithms has been omitted in this subsection for brevity. Of course, by extending the set of module data necessary for an elastic joint module, the automatic deployment of the controller can also be realized for the more general case of elastic joints. The additional parameter required for each elastic-joint module is simply the joint stiffness.

4.6 Robustness of the Combined Inverse-Dynamics/Passivity-Based Control

In this section, the assumption of perfect knowledge of the dynamical model in (4.1) is relaxed. Moreover, it is considered that the external torque vector $\boldsymbol{\tau}_{ext}$ can be different from zero. Since model uncertainties are considered, the model data available to the designer are only nominal. Hereafter, the nominal model terms are denoted by a subscript 0, while the perturbation from the respective real values are denoted by a subscript δ. For example, for any real unknown model term or feedback vector $\boldsymbol{\Xi}$, the following relations hold:

$$\boldsymbol{\Xi} = \boldsymbol{\Xi}_0 + \boldsymbol{\Xi}_\delta, \text{ and consequently } \boldsymbol{\Xi}_\delta = \boldsymbol{\Xi} - \boldsymbol{\Xi}_0. \tag{4.42}$$

In the following analysis, the dynamical parameters of the link-side dynamics are considered as uncertain while the ones of the motor-side, such as $\mathbf{J}$ and $\mathbf{K}$, are considered to be known with high precision. This can be easily justified since these parameters can be precisely estimated when the actuation unit is calibrated, before it equips the robot.

Considering the model in (4.1) and following the same procedure described in Subsection 4.4.4 to obtain (4.30) for the more general case in which $\boldsymbol{\tau}_{ext} \neq 0$, the following can be obtained:

$$\mathbf{J}\mathbf{K}^{-1}\Big(\mathbf{M}\,\mathbf{q}^{[4]} + \mathbf{C}\,\mathbf{q}^{[3]} + \boldsymbol{\eta}\Big) + \boldsymbol{\beta} = \mathbf{u} + \mathbf{J}\mathbf{K}^{-1}\mathbf{d}, \tag{4.43}$$

where

$$\boldsymbol{\eta} = 2\,\dot{\mathbf{M}}\,\mathbf{q}^{[3]} + \ddot{\mathbf{M}}\,\ddot{\mathbf{q}} + \mathbf{D}\,\mathbf{q}^{[3]} + \ddot{\mathbf{C}}\dot{\mathbf{q}} + 2\,\dot{\mathbf{C}}\ddot{\mathbf{q}} + \ddot{\mathbf{g}},$$
$$\boldsymbol{\beta} = \big(\mathbf{M} + \mathbf{J}\big)\ddot{\mathbf{q}} + \mathbf{h} + \mathbf{f}, \quad \text{and} \quad \mathbf{d} = \mathbf{K}\mathbf{J}^{-1}\boldsymbol{\tau}_{ext} + \ddot{\boldsymbol{\tau}}_{ext}.$$

In the above equation, the external torque vector that is part of $\mathbf{d}$ is considered to be sufficiently smooth and bounded so that $\|\mathbf{d}\| < \kappa_{\mathbf{d}}$, for some positive constant $\kappa_{\mathbf{d}}$.

Given that only nominal model information is available, the combined ID/PB control law for the system in (4.43) can be rewritten as follows:

$$\mathbf{u}_{IDPB_0} = \mathbf{J}\mathbf{K}^{-1}\boldsymbol{\eta}_0 + \boldsymbol{\beta}_0 + \mathbf{J}\mathbf{K}^{-1}\big(\mathbf{M}_0\,\mathbf{q}_{a0}^{[4]} + \mathbf{C}_0\,\mathbf{q}_{a0}^{[3]} + \boldsymbol{\Lambda}\,\mathbf{r}_0\big), \tag{4.44}$$

where

$$\boldsymbol{\eta}_0 = 2\,\dot{\mathbf{M}}_0\,\mathbf{q}_0^{[3]} + \ddot{\mathbf{M}}_0\,\ddot{\mathbf{q}}_0 + \mathbf{D}_0\,\mathbf{q}_0^{[3]} + \ddot{\mathbf{C}}_0\dot{\mathbf{q}} + 2\,\dot{\mathbf{C}}_0\ddot{\mathbf{q}}_0 + \ddot{\mathbf{g}}_0,$$
$$\boldsymbol{\beta}_0 = \big(\mathbf{M}_0 + \mathbf{J}\big)\ddot{\mathbf{q}}_0 + \mathbf{h}_0 + \mathbf{f}_0.$$

It is important to notice that in (4.44) the vectors containing the accelerations and jerks are also denoted as nominal. This is important because their model-based (or observer-based) estimation will be certainly imperfect for the uncertainty scenario considered. This mismatch is an additional source of perturbation that affects the closed-loop performance. The nominal feedback vectors available can be written as

$$\mathbf{r}_0 = \big(\mathbf{q}_d^{[3]} - \mathbf{q}_0^{[3]}\big) + \mathbf{K}_{HY3}\big(\ddot{\mathbf{q}}_d - \ddot{\mathbf{q}}_0\big) + \mathbf{K}_{HY2}\dot{\mathbf{e}} + \mathbf{K}_{HY1}\,\mathbf{e},$$
$$\mathbf{q}_{a0}^{[3]} = \mathbf{q}_d^{[3]} + \mathbf{K}_{HY3}\big(\ddot{\mathbf{q}}_d - \ddot{\mathbf{q}}_0\big) + \mathbf{K}_{HY2}\dot{\mathbf{e}} + \mathbf{K}_{HY1}\,\mathbf{e},$$
$$\mathbf{q}_{a0}^{[4]} = \mathbf{q}_d^{[4]} + \mathbf{K}_{HY3}\big(\mathbf{q}_d^{[3]} - \mathbf{q}_0^{[3]}\big) + \mathbf{K}_{HY2}\big(\ddot{\mathbf{q}}_d - \ddot{\mathbf{q}}_0\big) + \mathbf{K}_{HY1}\,\dot{\mathbf{e}}.$$

It is worth mentioning that no prior work considers this crucial aspect for analyzing robustness of controllers with high-order derivatives, to the best knowledge of the author.

By using the control law in (4.44) for the system in (4.43), and after performing algebraic manipulations that involve the use of the rule in (4.42), the following closed-loop relation can be obtained:

$$\mathbf{M}\,\dot{\mathbf{r}} + \mathbf{C}\,\mathbf{r} + \boldsymbol{\Lambda}\,\mathbf{r} = \boldsymbol{\psi}, \tag{4.45}$$

where

$$\boldsymbol{\psi} = \boldsymbol{\Lambda}\,\mathbf{r}_\delta + \mathbf{M}\mathbf{q}_{a\delta}^{[4]} + \mathbf{C}\,\mathbf{q}_{a\delta}^{[3]} + \mathbf{M}_\delta\mathbf{q}_{a0}^{[4]} + \mathbf{C}_\delta\,\mathbf{q}_{a0}^{[3]} + \boldsymbol{\eta}_\delta + \mathbf{K}\mathbf{J}^{-1}\boldsymbol{\beta}_\delta - \mathbf{d}. \tag{4.46}$$

In the above equation, $\boldsymbol{\psi}$ is a perturbation vector that affects the closed-loop. It is important to notice that (4.45) defines an output-strictly-passive operator $\boldsymbol{\psi} \rightarrow \mathbf{r}$ with storage function $V = \frac{1}{2}\mathbf{r}^T\mathbf{M}\mathbf{r}$, provided that $\boldsymbol{\Lambda}$ is positive definite.

Now, in light of the composition of $\boldsymbol{\psi}$ expressed in (4.46), additional assumptions are introduced subsequently.

Assumption 1. *There exist positive constants κ_{M_δ}, κ_{C_δ}, and $\kappa_{\eta\beta_\delta}$ that bound the deviation between the nominal and the real model terms as follows:*

$$\|\mathbf{M}_\delta\| < \kappa_{M_\delta}, \quad \|\mathbf{C}_\delta\| < \kappa_{C_\delta}, \quad \|\boldsymbol{\eta}_\delta + \mathbf{K}\mathbf{J}^{-1}\boldsymbol{\beta}_\delta\| < \kappa_{\eta\beta_\delta}. \tag{4.47}$$

This assumption implies that the mismatch between the real model terms involved and the nominal ones is not expected to be unbounded in practice.

Assumption 2. *There exist positive constants $\kappa_{a\delta}$ and $\kappa_{j\delta}$ that bound the deviation between the real and the estimated accelerations and jerks as follows:*

$$\|\ddot{\mathbf{q}}_\delta\| < \kappa_{a\delta}, \quad \|\mathbf{q}_\delta^{[3]}\| < \kappa_{j\delta}. \tag{4.48}$$

This assumption is reasonable in practice once the estimator of the accelerations and jerks is properly calibrated and when considering that real actuators have speed and saturation limits. Please note that the boundedness of the deviation in the estimation of the accelerations and jerks should always be evaluated before one can use them for feedback control. This assumption implies that

$$\|\mathbf{q}_{a\delta}^{[4]}\| < \kappa_{as\delta}, \quad \|\mathbf{q}_{a\delta}^{[3]}\| < \kappa_{aj\delta}, \tag{4.49}$$

for some positive constants $\kappa_{as\delta}$ and $\kappa_{aj\delta}$, which can be shown by considering that the following relations hold

$$\mathbf{q}_{a\delta}^{[4]} \overset{(4.42)}{=} -\mathbf{K}_{HY3}\mathbf{q}_\delta^{[3]} - \mathbf{K}_{HY2}\ddot{\mathbf{q}}_\delta, \;\; \mathbf{q}_{a\delta}^{[3]} \overset{(4.42)}{=} -\mathbf{K}_{HY3}\ddot{\mathbf{q}}_\delta.$$

Assumption 3. *There exist positive constants κ_M and κ_C so that*

$$\|\mathbf{M}\| \leq \kappa_M, \quad \|\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\| \leq \kappa_C \|\dot{\mathbf{q}}\|. \tag{4.50}$$

This assumption is usual in control of robot manipulators; the interested reader can refer to fundamental books (such as [18]) for further detail.

With these assumptions it is now possible to show that any user defined tracking precision can be met in principle, provided that the gains ($\boldsymbol{\Lambda}$, $\mathbf{K}_{HY3}$, $\mathbf{K}_{HY2}$, $\mathbf{K}_{HY1}$) are properly selected. This can be shown using the storage function

$$V = \frac{1}{2}\mathbf{r}^T\mathbf{M}\mathbf{r},$$

whose derivative is

$$\dot{V} = \mathbf{r}^T\big(- \boldsymbol{\Lambda}\,\mathbf{r} + \boldsymbol{\psi}\big) \leq -\lambda_{\min}(\boldsymbol{\Lambda})\,\|\mathbf{r}\|^2 + \|\mathbf{r}\|\|\boldsymbol{\psi}\|. \tag{4.51}$$

The assumptions made so far allow one to define the following perturbation bounding function

$$\rho = \kappa + \kappa_\nu \|\dot{\mathbf{q}}\| + \kappa_{M_\delta}\|\mathbf{q}_{a0}^{[4]}\| + \kappa_{C_\delta}\|\mathbf{q}_{a0}^{[3]}\| + \varphi_p, \tag{4.52}$$

where

$$\kappa = \kappa_M \kappa_{as\delta} + \kappa_{\eta\beta_\delta} + \kappa_d, \quad \kappa_\nu = \kappa_C \kappa_{aj}, \quad \varphi_p > 0.$$

By considering the assumptions, (4.46), (4.52), and the selection of $\boldsymbol{\Lambda} = \rho\mathbf{I}$, with $\mathbf{I}$ being the identity matrix of proper dimensions, it can be shown that the following inequality holds:

$$(1 + \|\mathbf{r}_\delta\|)\rho > \|\boldsymbol{\psi}\|.$$

Now, the inequality in (4.51) can be rewritten as

$$\dot{V} < \rho\|\mathbf{r}\|(-\|\mathbf{r}\| + \|\mathbf{r}_\delta\| + 1).$$

The above inequality implies that $\dot{V} < 0$ for

$$\|\mathbf{r}\| \geq \kappa_{j\delta} + \|\mathbf{K}_{HY3}\|\kappa_{a\delta} + 1 \geq \|\mathbf{r}_\delta\| + 1,$$

since

$$\mathbf{r}_\delta \overset{(4.42)}{=} -\mathbf{q}_\delta^{[3]} - \mathbf{K}_{HY3}\ddot{\mathbf{q}}_\delta. \tag{4.53}$$

From this result, uniform ultimate boundedness of the trajectory $\mathbf{r}$ follows. In particular, since the inertia matrix is quadratically bounded by

$$\lambda_m\|\mathbf{x}\|^2 \leq \mathbf{x}^T\mathbf{M}\mathbf{x} \leq \lambda_M\|\mathbf{x}\|^2,$$

where $\lambda_m > 0$ and $\lambda_M < \infty$ are the respective minimum and maximum eigenvalues of the inertia matrix ($\forall\,\mathbf{q}$), the trajectory $\mathbf{r}$ is ultimately bounded by

$$\|\mathbf{r}\| \leq (\|\mathbf{r}_\delta\| + 1)\sqrt{\frac{\lambda_M}{\lambda_m}}, \overset{(4.53)}{\leq} (\kappa_{j\delta} + \|\mathbf{K}_{HY3}\|\kappa_{a\delta} + 1)\sqrt{\frac{\lambda_M}{\lambda_m}}. \tag{4.54}$$

This follows from the same arguments presented for the robust PB controller in Subsection 3.4.1.

The relation in (4.54) shows that the bounds of $\mathbf{r}$ do not necessarily shrink by increasing the gains. In particular, the increase of $\mathbf{K}_{HY3}$ is counterproductive and enlarges these bounds. Although there is no known means to shrink the bounds of $\mathbf{r}$, the bounds on the trajectory tracking error $\mathbf{e}$ can still be made arbitrarily small. This can be seen by considering (4.36). In fact, considering this transfer function as the result of a series of first order passive low pass filters for each coordinate j as

$$G_j(s) = \frac{e_j(s)}{r_j(s)} = \frac{1}{(s + K_G)^3} = \frac{1}{s^3 + 3K_G\,s^2 + 3K_G^2\,s + K_G^3},$$

with a positive K_G, results in the gains $\mathbf{K}_{HY3} = 3K_G\,\mathbf{I}$, $\mathbf{K}_{HY2} = 3K_G^2\,\mathbf{I}$, and $\mathbf{K}_{HY1} = K_G^3\,\mathbf{I}$. With these gains and $\|\mathbf{r}\|$ ultimately bounded by (4.54), it follows that

$$\begin{aligned}\|e_j\| &\leq \frac{(\kappa_{j\delta} + \|\mathbf{K}_{HY3}\|\kappa_{a\delta} + 1)}{K_G^3}\sqrt{\frac{\lambda_M}{\lambda_m}} \\ &\leq \frac{(\kappa_{j\delta} + 3K_G\kappa_{a\delta} + 1)}{K_G^3}\sqrt{\frac{\lambda_M}{\lambda_m}} \\ &\leq \left(\frac{(\kappa_{j\delta} + 1)}{K_G^3} + \frac{3\kappa_{a\delta}}{K_G^2}\right)\sqrt{\frac{\lambda_M}{\lambda_m}},\end{aligned}$$

for the jth coordinate. This relation shows that the bounds of the tracking error can be made arbitrarily small by selecting a large enough tuning parameter K_G.

It should be noted that other choices of the gains can be made (e.g., $\mathbf{K}_{HY2} = (3K_G^2 + \delta_2)\,\mathbf{I}$ and $\mathbf{K}_{HY1} = (K_G^3 + \delta_1)\,\mathbf{I}$) for some positive δ_1 and δ_2 that still maintain the transfer function (4.36) stable and sufficiently input-output suppressing (i.e., with sufficiently low H_∞ norm). Please note that although the meaning of all bounds considered has been detailed, obtaining the exact function (4.52) is not simple in practice. One possible shortcut is to find a large enough *constant* ρ from tuning iterations that provides satisfactory tracking, along with a large enough K_G.

4.7 Performance Evaluation using Simulations

This section presents simulation results of the comparison of global tracking controllers that can be automatically deployed, thanks to their efficient numerical computability. The comparison is between the combined ID/PB scheme and the classical inverse-dynamics controller. The tracking performance and the sensitivity to small mismatches of the system dynamics are considered. After describing the simulation test bed, a practical solution for estimating joint accelerations and jerks using an observer is described. Then, simulation results that also show the influence of the observer are discussed.

4.7.1 Simulation Test Bed

Two simple planar robots with two and six revolute joints (2R and 6R, respectively) composed of cylindrical links and working in the vertical plane are considered. The link-side assembly of the 2R robot is equivalent to the one in Figure 3.4, while the assembly of the 6R robot is a straightforward extension of it with four more links.

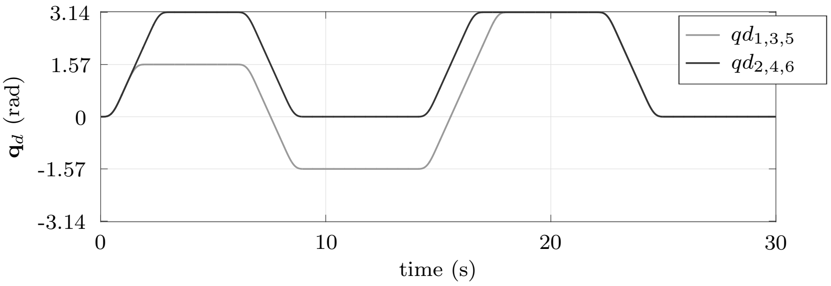


Figure 4.1: Required joint-space trajectory using sin^2 profiles.

The nominal parameters of the ith link used are the length $l_i = 0.5\,m$, the radius of the enclosing cylinder $r_i = 0.025\,m$, and its mass $m_i = 2.5\,kg$. The parameters considered for the motor-side dynamics are the rotor inertia $J_i = 0.5\,kgm^2$ and the elastic-joint stiffness coefficient $K_i = 4650\,Nm/rad$. In addition, the sampling time of $1\,ms$ is introduced using zero-order hold. Sensitivity to uncertain model parameters is assessed by simulating small mismatches of the actual mass (and inertia) of the last link of the considered arms.

The trajectory to track is defined in joint space, using a sequence of smooth sin^2 profiles as in [111, Section 6.3]; the maximum acceleration is set to $3\,rad/s^2$ and the maximum velocity is set to $1.5\,rad/s$. The resulting trajectory used for these simulations is shown in Figure 4.1.

4.7.2 Estimation of Joint Accelerations and Jerks

In practice, the use of an observer to obtain better estimates with respect to those obtained by means of (4.38) and (4.39) may be necessary. This is typically necessary when the torque readings are very noisy or when the model mismatch is significant. Better estimates can be obtained following the idea of the kinematic Kalman filter proposed in [13].

When considering precise model knowledge but noisy torque readings, the estimation of accelerations and jerks with (4.38) and (4.39) may be affected by an excessive amount of noise. With the state vector $\mathbf{x}(t) = [q \;\; \dot{q} \;\; \ddot{q} \;\; q^{[3]}]^T$, the following model can be used for designing the filter:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\,\mathbf{x}(t) + \mathbf{\Gamma}\,w(t), \quad \mathbf{y}(t) = \mathbf{C}\,\mathbf{x}(t) + \mathbf{e}(t),$$

where $w(t)$ is an unbiased white process noise with covariance c, $\mathbf{y}(t)$ the vector of the available measurements, and $\mathbf{e}(t)$ is the vector of unbiased white measurement noise having covariance matrix $\mathbf{R}$. Now, by assuming availability of measurements of joint positions, velocities (using tachometers or numerical differentiation of position measurements), and noisy joint accel-

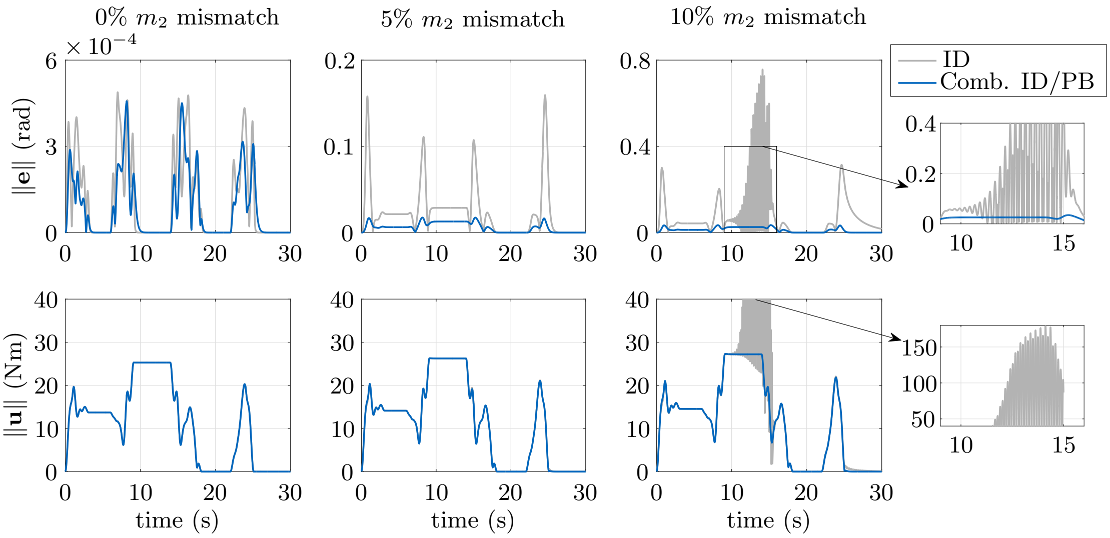


Figure 4.2: Simulation comparison of tracking performance and torque commands between ID control and combined ID/PB control for a 2R planar arm. The performances of the classical ID control (ID) are compared with those of the combined ID/PB control (Comb. ID/PB). The simulations have been performed considering perfect model information (first column of plots), an additive 5% mismatch for the second link mass (second column of plots), and finally an additive 10% mismatch for the second link mass (third column of plots).

erations from (4.38), the following system matrices can be considered:

$$\mathbf{A} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad \mathbf{\Gamma} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}, \quad \mathbf{C} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

Please note that one can simply resort to the use of position and velocity measurements for the filter, if significant systematic errors in the estimate of the accelerations using (4.38) is present. Such a version of the filter is simply adapted by removing the third row of $\mathbf{C}$. Using similar arguments as in [12], $w(t)$ can be seen as a surrogate of $q^{[4]}$ and its covariance c as a filter-tuning parameter since the above mentioned stochastic model can only approximate the actual robot motion. This alternative version does not require model knowledge and can be directly implemented in each elastic-joint module.

4.7.3 Simulation Results

The first simulation results presented involve the 2R robot for which model mismatch has been introduced. The mismatches considered are 5% and 10% of unexpected additional mass for

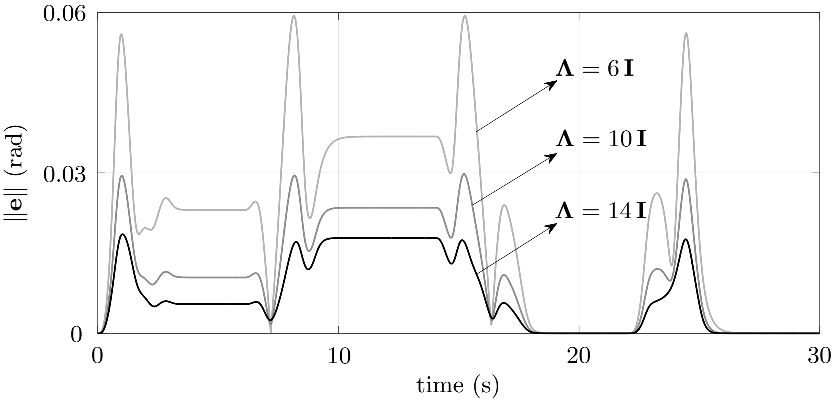


Figure 4.3: Effect of $\mathbf{\Lambda}$ on the tracking performance of the combined ID/PB controller for a 2R planar arm considering an additive 10% mismatch for the second link mass and keeping the other gain matrices fixed.

the second link. The results of these simulations are collected in Figure 4.2. This figure shows the tracking performance of the classical inverse-dynamics scheme and the combined inverse-dynamics/passivity-based control scheme for three cases: perfect model information and the two scenarios including model mismatch. By observing the development in time of the tracking error for the increasing model mismatch (second and third columns of plots), the combined ID/PB control scheme has remarkable insensitivity with respect to the model mismatch compared to the classical inverse-dynamics controller. As it is noticeable from the first column of plots in Figure 4.2, the gains for the two controllers under test have been selected in a fair way, by tuning them to provide equivalent performance when perfect model information is assumed. Please note that the sampling effects considered in these simulations prevent the tracking error from approaching the numerical precision of the solver as one would otherwise expect for the case of perfect model knowledge. For these simulations, the joint accelerations and jerks have been estimated using (4.38) and (4.39). The consequences from using an observer for their estimation are presented next. The gains that have been used for ID control and for combined ID/PB control are respectively: $\mathbf{K}_{ID3} = 150\mathbf{I}$, $\mathbf{K}_{ID2} = 7 \cdot 10^3\mathbf{I}$, $\mathbf{K}_{ID1} = 140 \cdot 10^3\mathbf{I}$, $\mathbf{K}_{ID0} = 1 \cdot 10^6\mathbf{I}$, and $\mathbf{K}_{HY3} = 150\mathbf{I}$, $\mathbf{K}_{HY2} = 6,5 \cdot 10^3\mathbf{I}$, $\mathbf{K}_{HY1} = 40 \cdot 10^3\mathbf{I}$, $\Lambda = 9\mathbf{I}$, where $\mathbf{I}$ is the 2×2 identity matrix.

An additional benefit of the combined ID/PB controller is its easy tunability using the gain matrix $\mathbf{\Lambda}$. This gain matrix has an intuitive influence on the tracking error since it allows one to directly inject damping in (4.45). This results in the direct reduction of the tracking error when increasing entries of $\mathbf{\Lambda}$, without having to consider the roots of the polynomials (4.4.4) for a required increase of tracking performance. The effect of this intuitive tuning knob is presented in Figure 4.3, which shows the development in time of the tracking error for three cases of $\mathbf{\Lambda}$. Of course, the matrices $\mathbf{K}_{HY1}$, $\mathbf{K}_{HY2}$ and $\mathbf{K}_{HY3}$ have not been modified for these simulations, and the same values of the previously mentioned simulations have been used.

The effect of the use of an observer for estimating joint accelerations and jerks has also been evaluated using simulations. For this purpose, position/velocity measurement noise has been introduced such that the following covariance matrix is used: $\mathbf{R} = diag\{10^{-9}, 10^{-7}, 20\}$. The results of all tracking tests have been collected in Figure 4.4. For this evaluation, both the 2R (first column of plots) and the 6R (second column of plots) planar robots have been considered. For all cases, a 10% mismatch of the mass of the last link has been introduced. The three rows of plots correspond to the case of no observer and no noise present, noise and observer (with $c = 20$), and noise and observer (with $c = 16$), respectively. By observing the results in these rows, it clearly appears that the introduction of the observer significantly robustifies the ID controller and leads to comparable performance for the 2R case between the classical ID and the combined ID/PB control. For the 6R case, higher error peaks can be noticed for the classical ID controller. The second and third rows of plots show the effect of the reduction of the filter parameter c. Its reduction increases the noise suppression at the expense of more delayed estimates. The simulation results show that the ID controller is more sensitive to the delayed estimation from the filter with lower values of c, with the increase of the complexity of the robot considered. Therefore, the combined ID/PB scheme is still preferred. The gains used for these simulations are the same as the ones used in the previously mentioned simulations with $\Lambda = 9\mathbf{I}$.

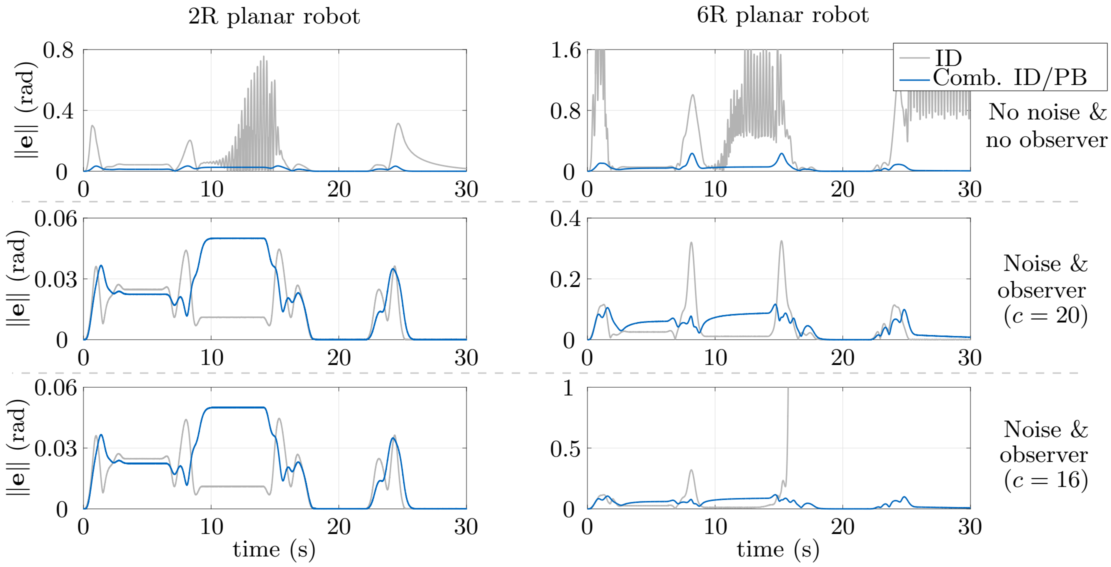


Figure 4.4: Simulation comparison showing the effect of the observer for a 2R planar robot (left column of plots) and for a 6R planar robot (right column of plots). The performance of the classical ID control (ID) are compared with those of the combined ID/PB control (Comb. ID/PB). Each plot represents a simulation with 10% added mass mismatch at the last link of each arm. The first row of plots collects results for a scenario without noise and without observer in use. The second row considers measurement noise and the use of the observer with $c = 20$ and the last row considers measurement noise and the use of the observer with $c = 16$.

4.8 Experimental Application

In addition to the numerical results presented so far, real-world experiments have also been performed to verify the effectiveness of the combined ID/PB control schemes and their efficient numerical computability. In this section, the robot test beds used for the experiments and the achieved tracking control performance are presented.

Two different robots have been employed. One is a simple reconfigurable platform which is composed of link modules and two elastic-joint modules. This platform is especially suitable for demonstrating the efficient automatic implementation of global trackers. The second platform used is a fixed-configuration seven degrees-of-freedom robotic arm with four elastic joints. This robots is used to demonstrate the combined ID/PB control variants on a complex robot arm. The control design of this arm needs special considerations since it is composed of mixed rigid/elastic joints. Even though the second test bed is based on a fixed-configuration arm, the results presented are still useful to understand how the controller would behave for a reconfigurable robot. This is possible since the approach described in Chapter 2 can be used

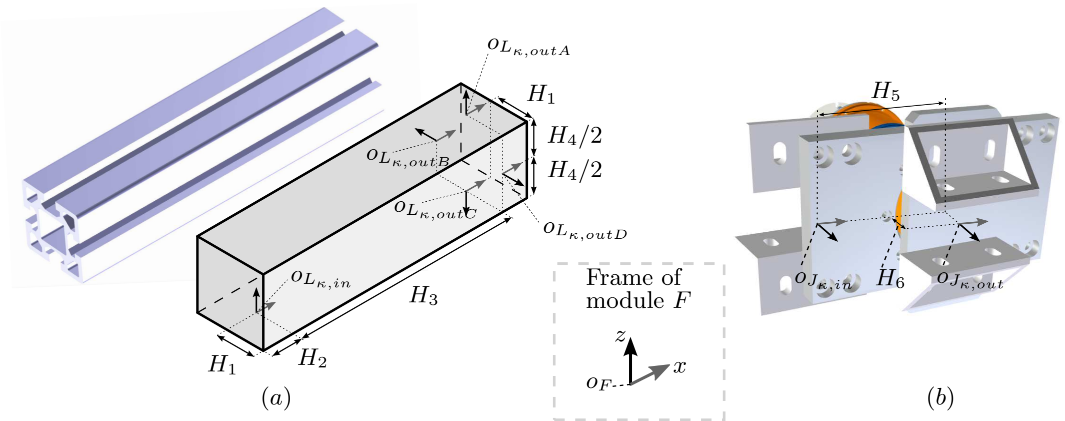


Figure 4.5: Definition of the input and output frames for the available link-modules (a) and the elastic-joint modules (b). The legend in the bottom shows the frame notation used for compactness, where the y axis is omitted and assumed to complete the right-handed coordinate system.

for automatically obtaining the link-side models of arbitrary assemblies.

4.8.1 2R Reconfigurable Elastic-Joint Robot

This robot test bed is composed of aluminum profiles as link-modules and elastic-joint-modules illustrated in Figure 4.5. The available measurements provided by each motor driver board are the motor-side joint position, the deflection of the elastic element, and the rate of change of these quantities. Therefore, knowing the elastic-joint stiffness, the elastic torque $\boldsymbol{\tau}_e = \mathbf{K}(\boldsymbol{\theta} - \mathbf{q})$ and its rate of change $\dot{\boldsymbol{\tau}}_e = \mathbf{K}(\dot{\boldsymbol{\theta}} - \dot{\mathbf{q}})$ can be directly obtained. The control laws under test have been implemented using Simulink Real Time 2015b and an embedded target PC (PC/104) equipped with 2 GB RAM and an Intel Core 2 Duo CPU running at 1.86 GHz. The sampling time is 1 ms.

Derivation of the Module Data

The first step for characterizing the modules is to set the input and the output coordinate frames. While for the available joint modules only one pair of input-output connectors is considered, multiple output connectors ($OutA$, $OutB$, $OutC$, $OutD$) can be defined for these link modules (see Figure 4.5a). This enlarges the set of possible resulting assemblies of the robot arm. The input-output frames, as shown in Figure 4.5, are assigned such that the output frame of one module matches the input frame of the subsequent one (as in Chapter 2), when a connection is realized. Once these frames are assigned, the set of kinematic and dynamic parameters of the module data for each pair of input-output frames can be obtained applying

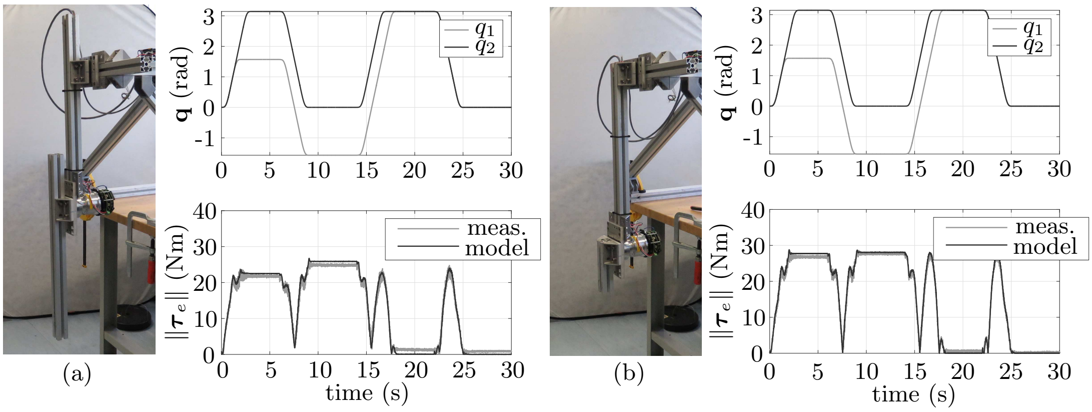
	a^l	α^l	p^l	n^l	$\delta^{l,in}$	$\delta^{l,out}$
in-outA	H_3	0	$-H_4$	0	0	0
in-outB	H_3	$-\pi/2$	$-H_4/2$	H_1	0	0
in-outC	H_3	$-\pi$	0	0	0	0
in-outD	H_3	$\pi/2$	$-H_4/2$	H_1	0	0

Table 4.1: Kinematic module data of link-module L_κ, with reference to Figure 4.5a.

Proximal Part					Distal Part					Joint	
a^{pl}	α^{pl}	p^{pl}	n^{pl}	δ^{pl}	a^{dl}	α^{dl}	p^{dl}	n^{dl}	δ^{dl}	δ^J	Joint type
H_5	0	$-H_6/2$	0	0	0	0	$-H_6/2$	0	0	0	Revolute

Table 4.2: Kinematic module data of joint-module J_κ, with reference to Figure 4.5b.

the notation described in detail in Chapter 2. The generic kinematic parameters derived for this selection of input-output frames and this module geometry are collected in Table 4.1 and Table 4.2. In addition to the kinematic parameters, the dynamic ones (mass, inertia tensor, and coordinates of center of mass) for link modules and the proximal part of joint modules are expressed with respect to the input frame, while the parameters for the distal part of the joint module are expressed with respect to the output frame. We obtain these kinematic and dynamic parameters using CAD data. The motor-side friction parameters of the joint modules as well as the rotor inertia have been estimated by a simple identification procedure equivalent to the one applied in Subsection 2.6.2. In all the experiments with this test bed the motor-side friction terms have been compensated by feedforward control. The elastic-joint stiffness is about $4650\,Nm/rad$, which has been estimated by locking the rotor and by applying different known loads at the link-side to measure the corresponding joint deflection.



(a)

(b)

Figure 4.6: Evaluation of the automatically generated model matching with the measured torque readings for two different assemblies (a) and (b).

Validation of the Models

For each configuration of the robot, the model-based controller is automatically generated. The quality of the automatically obtained robot models can be evaluated by measuring the elastic joint torques $\boldsymbol{\tau}_e$ while the robot performs movements and by comparing them with the torques predicted with the automatically generated model. Figure 4.6a and Figure 4.6b show the match between the model and the actual torque measurements, where two different robot assemblies are considered. From these figures, a significantly good match is shown during the motion.

Evaluation of the Control Performance

Using the robot test bed of Figure 4.6a, both the classical ID control and the combined ID/PB control have been tested. The results of these experiments are shown in Figure 4.7. This figure contains the results of tracking control for two different cases. The first case assumes the robot dynamics to be close to the available ones from the nominal model. The second case involves a significant model mismatch that has been introduced by adding $2\,kg$ of unmodeled payload (i.e., assumed as unknown for the control). The tracking performance for the former scenario can be observed in the first column of plots of Figure 4.7, while the latter case is shown in the second column of plots of the same figure. For a fair comparison, the tuning of the two controllers has been performed so that they have comparable performance (tracking error and amplification of the measurement noise) for the case of no payload. The mentioned fairness can be seen in the first column of plots in Figure 4.7 where a smooth tracking behavior for both

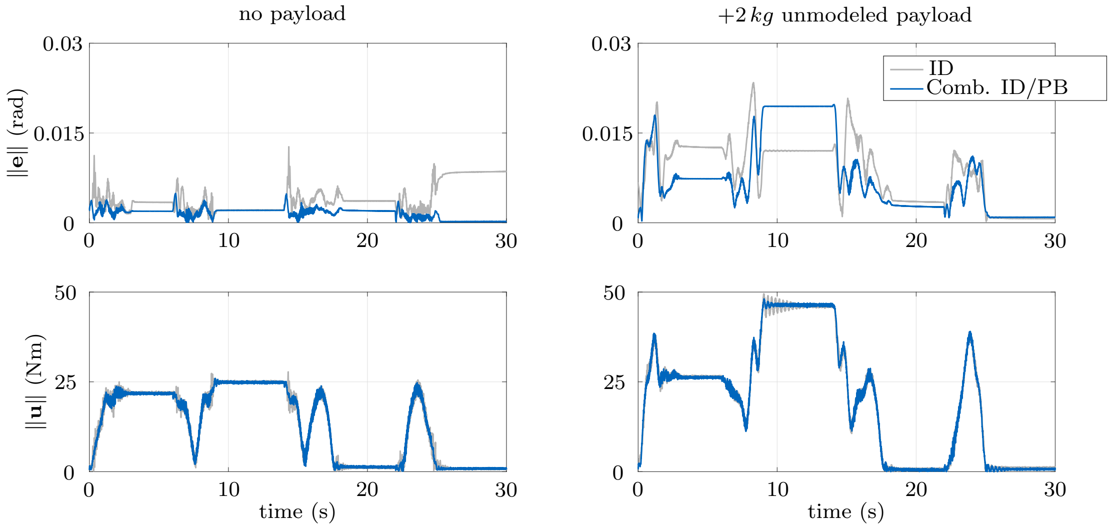


Figure 4.7: Trajectory tracking performance of ID control and combined ID/PB control with the robot of Figure 4.6a (with and without unmodeled payload).

	ID control	Comb. ID/PB control
Maximum (10^{-4}s)	1.445	1.449
Mean (10^{-5}s)	5.32	5.39
Standard deviation (10^{-5}s)	2.61	2.63

Table 4.3: Comparison of the total execution time of the controller outputs used in the experiments with the 2R reconfigurable elastic-joint robot.

controllers is shown, as well as by observing the development of the torque commands over time. For these tests the same trajectory has been used as for the previously mentioned simulations. It is interesting to notice that Figure 4.7 also shows acceptable performance for the loaded robot, for both controllers. This is consistent with the simulation results discussed previously, thanks to the robustifying effect introduced by the observer for the 2R robot. The use of the observer has been necessary for performing experiments with this robot for dealing with noisy torque readings. While these experiments show real-word applicability of the tested controllers, it is important to mention that the global tracking properties of the considered schemes are not necessarily guaranteed when the observer is used. In general, even though the observer can be a practical solution for estimating the accelerations and jerks, it is preferable not to use it from a theoretical point view, if torque readings and the model available are sufficiently precise. Finally, in these experiments no significant computational effort has been required for either tested controllers, as shown in Table 4.3.

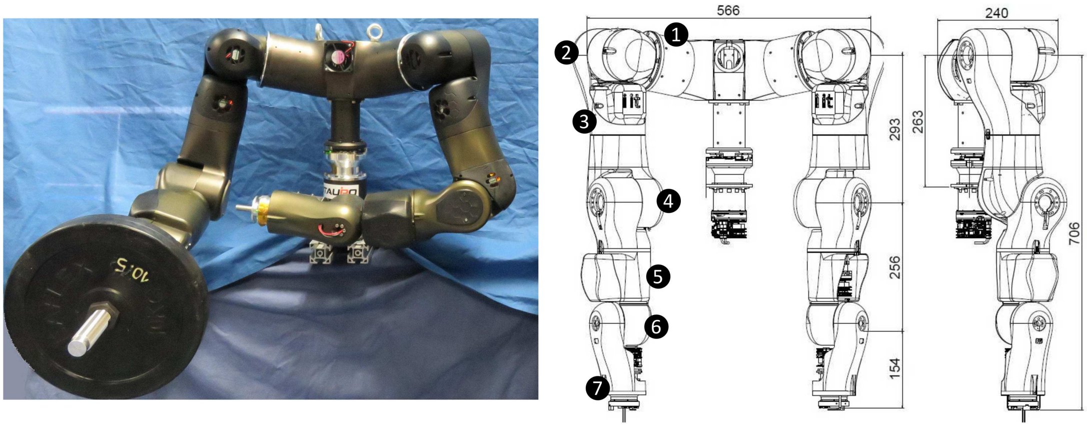


Figure 4.8: Picture (*left*) and dimensions in mm (*right*) of the bi-manual robotic platform [9] serving for the experimental evaluation of the proposed controllers.

4.8.2 7R Mixed Rigid/Elastic Joint Robot

Next, the proposed approach is demonstrated by a 7 degrees of freedom arm, which is the right arm of the humanoid robot torso shown in Figure 4.8. Technical details of the actuation units employed are collected in Table 4.4. While the first four joints of the kinematic structure have a stiffness of about 6000 Nm/rad, the last three are much more rigid and have a stiffness of 21000 Nm/rad, which is in the order of the harmonic drive flexibility. Because of this significant difference, the last three joints are considered to be rigid. The arm can be seen as a mixed rigid/elastic joint manipulator, implying that the dynamics cannot be directly modeled by (4.1). However, the implementation of the controllers described in this chapter is still possible, provided that the elastic joints dynamics are virtually created in the controller itself as described in [31]. This has been realized by including dynamic feedback terms that emulate the presence of the virtual motor-side dynamics in the controller, so that the whole arm can virtually be seen as fully composed of elastic joints. The controllers for these experiments have been implemented using Simulink Real Time 2015b and an Intel NUC with 3 GB DDR-3 RAM and a Core i5-3427U processor (3M cache) running at 1.8 GHz.

For all the experiments described in this subsection, the desired trajectories have been computed using 7th order polynomials for point to point motions in joint space. The details of these trajectories can be found in Appendix B.

For the above mentioned test bed, the tracking performance of the combined ID/PB control and its variant with feedforward ID action are presented next. The tests performed involved fast

type	gear ratio	joint index	velocity max. [rad/s]	peak torque [Nm]	sensor stiffness [Nm/rad]
Medium A	160	1, 2	5.7	147	6 000
Medium B	160	3, 4	8.2	147	6 000
Small A	100	5, 6	11.6	55	21 000
Small B	100	7	20.3	27	21 000

Table 4.4: Actuator specifications for right arm. Joint indices refer to Figure 4.8.

trajectory tracking and unknown payloads of significant magnitude. In addition, a test where the gains are reduced is also presented, to evaluate the behavior of the feedforward-ID/PB controller with respect to a simple PD-based scheme, showing the benefits of using a model.

Validation of the Model

The link-side model of this robot has been derived using CAD data, while the motor-side parameters (with the exception of the friction parameters) have been obtained from the motor data sheets. A simple identification procedure has been used for estimating the motor-side friction coefficients and is described next. The following simple friction model is used:

$$f_i(\theta_i) = \beta_{v,i}\dot{\theta} + \beta_{c,i}\text{sign}(\dot{\theta}),$$

for the ith joint axis, where $\beta_{v,i}$ and $\beta_{c,i}$ are the viscous and static friction coefficients, respectively. The friction coefficients have been obtained by performing the following regression for each ith axis with an elastic joint and using k samples:

$$\begin{pmatrix}\beta_{v,i} & \beta_{c,i}\end{pmatrix}^T = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{b},$$

where

$$\mathbf{\Phi} = \begin{pmatrix} \dot{\theta}_i(1) & \text{sign}(\dot{\theta}_i(1)) \\ \vdots & \vdots \\ \dot{\theta}_i(k) & \text{sign}(\dot{\theta}(k)) \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} u_i(1) - \tau_{e,i}(1) - J_i\ddot{\theta}_i(1) \\ \vdots \\ u_i(k) - \tau_{e,i}(k) - J_i\ddot{\theta}_i(k) \end{pmatrix}.$$

The acceleration data required for the regression have been obtained using zero-phase digital filtering [79] on the double numerical differentiation of the joint position data. For estimating the friction coefficients of the axes with rigid joints, the same approach is used; however, the model of the rigid-link assembly using a recursive Newton-Euler algorithm is evaluated in place

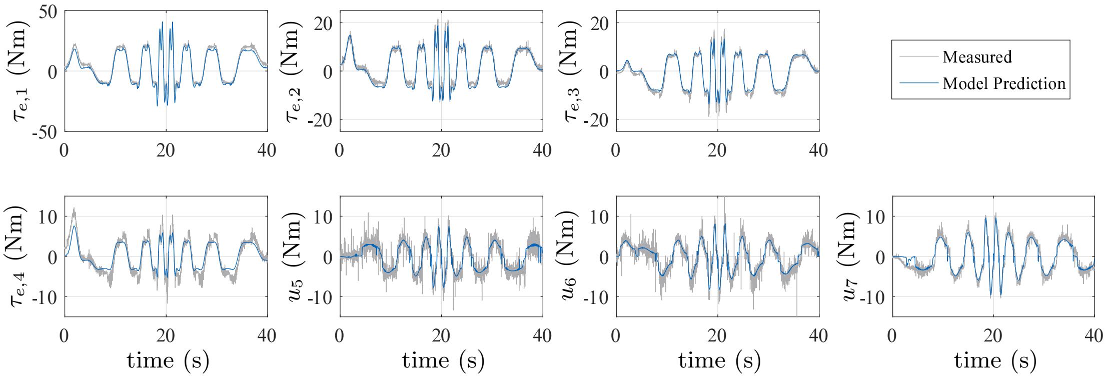


Figure 4.9: Evaluation of model accuracy for a closed-loop trajectory tracking control test. The model prediction is computed offline by using the recursive Newton-Euler algorithm for which we use zero-phase digital filtering for estimating the acceleration.

of the measured τ_e in this case. In all the experiments, the motor-side friction terms have been compensated by feedforward action.

The quality of the available model can be observed in Figure 4.9, which shows the match of the model prediction derived using CAD data, motor data-sheets, and identified friction coefficients with respect to the real torque measurements for the axes equipped with elastic joints. Further, the applied actuation torques (from current signals) in comparison with the model prediction is shown for the axes with rigid joints. This figure shows a satisfactory match. Better results are expected if dedicated identification procedures are performed (see e.g., [6, 15, 108]). However, this is not desirable for a modular reconfigurable scenario since it would be time consuming and limit swift reconfigurability. One of the additional purposes of these experiments is to show applicability of the combined ID/PB control schemes for cases in which the model is simply derived from CAD data. In fact, this would be the case if the robot to control were an assembly of a reconfigurable arm running the framework described in Chapter 2.

Practical Estimation of Joint Accelerations and Jerks

The finite precision of the available model and the noisy torque readings make the estimates of joint accelerations and jerks using (4.38) and (4.39) not sufficiently accurate. For this implementation, a simple kinematic Kalman filter of the type described in Subsection 4.7.2 has been used, for the case in which $\mathbf{C} = (\mathbf{I}_{2\times2}, \mathbf{0}_{2\times2})$. Additional filtering by a first-order low-pass filter with cutoff frequency at $5\,rad/s$ has been included for an additional practical suppression of undesired spectral components. The results of a test for showing the estimation quality of

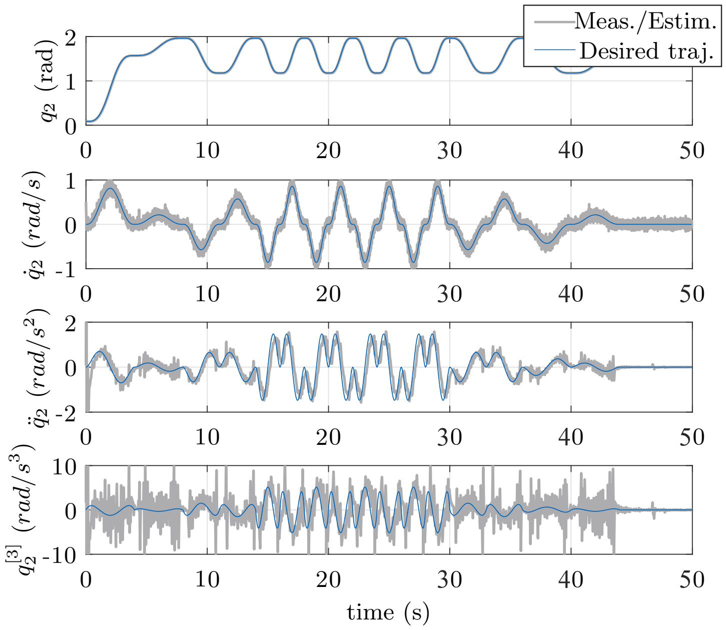


Figure 4.10: Quality of the estimation of acceleration and jerk for the second axis of the robot during a trajectory tracking control test.

the acceleration and jerk for an exemplary axis is presented in Figure 4.10. These results are relative to the second axis only since the ones of the other axes do not differ significantly.

Evaluation of the Control Performance

The first results presented aim at verifying the tracking performance of the combined ID/PB control and its variant with feedforward inverse dynamics, when fast trajectories are required. Further, the two controllers are compared with a simpler, model-free PD controller which results from removing all link-side model terms as well as the accelerations and jerks feedback from the combined ID/PB law:

$$\mathbf{u}_{PD} = -\mathbf{J}\mathbf{K}^{-1}\boldsymbol{\Lambda}\left(\mathbf{K}_{HY2}\dot{\mathbf{e}} + \mathbf{K}_{HY1}\,\mathbf{e}\right) + \mathbf{f}(\dot{\boldsymbol{\theta}}).$$

The results of this first series of experiments are presented in Figure 4.11. From this figure the performance of the controllers can be observed by considering the development over time of the tracking error norm. The zoomed part of the tracking error plot in Figure 4.11 highlights the behavior of the controllers when the trajectory becomes fast. In this case, the improved performance of the combined ID/PB schemes can be observed. In particular, the controller that provides better performance is the combined feedforward-ID/PB. This is due to the filtering

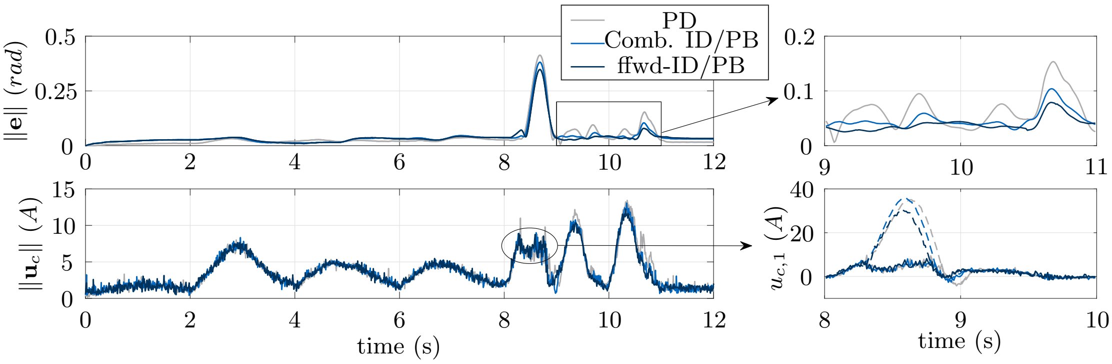


Figure 4.11: Experiments of tracking control with fast desired trajectory. The dashed lines in the bottom right plot represent the current commands required from the controllers for the first joint ($u_{c,1}$), where a saturation is experienced. The trajectory used for this test is in the appendix.

procedure used for obtaining the accelerations and jerks which introduces an unavoidable phase lag that can also be seen from Figure 4.10. Please note that this experiment has been executed with a trajectory that has led the robot to reach the physical actuation limits. This can also be observed in the bottom right of Figure 4.11, where the desired actuation current and its saturated execution for the first axis of the robot are shown. The gains used in this experiment are

$$\begin{aligned}
\mathbf{K}_{HY3} &= 10^3\text{diag}([60, 120, 120, 120, 2000, 2000, 2000]),\\
\mathbf{K}_{HY2} &= 10^3\text{diag}([2, 3.5, 3.5, 3.5, 50, 50, 50]),\\
\mathbf{K}_{HY1} &= \text{diag}([50, 50, 50, 50, 375, 375, 375]),\\
\mathbf{\Lambda} &= \text{diag}([70, 40, 40, 40, 20, 20, 20]). \qquad (4.55)
\end{aligned}$$

The performance of the combined ID/PB controllers have been additionally tested for different unmodeled payloads. The results of these tests are shown in Figure 4.12, where the different payloads used can be noticed. By observing the trajectory tracking error norms in Figure 4.12, smooth evolutions over time are shown from both controllers, without any sign of instability and oscillations. The performance of the combined ID/PB controller and that of its feedforward-ID/PB variant are not easily distinguishable due to the relatively slow speed of the desired trajectory. This is expected since for sufficiently slow trajectories the delay introduced by the acceleration and jerk filtering becomes negligible. The maximum speed of this trajectory has been limited to comply with the physical limits of the robot for reliably carrying a 4kg payload. The gains have not been modified from the previously described experiments.

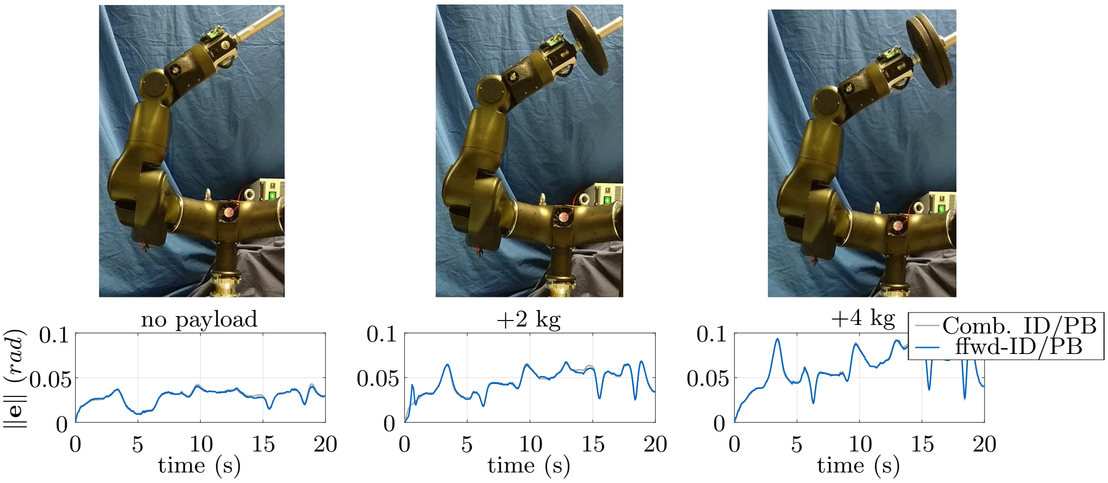


Figure 4.12: Test results for trajectory tracking with different payloads. The trajectory used for this test is in the appendix.

An additional experiment has been executed to evaluate the effect of the gain reduction on the tracking performance. The aim is to show the benefit of the feedforward-ID/PB controller with respect to its reduced PD version when manually reducing the gains. The reduction of the gains may be desired for achieving movements with low impedance, but this would not be practical if the tracking performance were significantly corrupted. For this test, the desired trajectory has been selected such that the robot can perform a fast sawing motion. The oscillatory behavior of the error can observed in Figure 4.13, where a significant increase in amplitude for the case of the PD control is noticed. In this test, all entries of $\mathbf{\Lambda}$ have been decreased to 50% and 25% of their original values in (4.55) (orange and red areas in Figure 4.13, respectively). The peak-to-peak value of the oscillations raises from about 0.128 *rad* to 0.205 *rad* when the model-free PD scheme is used. In contrast, the peak-to-peak error of the feedforward-ID/PB control is not significantly affected and raises only from 0.046 *rad* to 0.065 *rad*. The reduced amplitude oscillations allow more precise movements and lead to safer conditions for letting e.g., a human establish a physical interaction with the arm. Finally, it is important to report that the maximum total execution time required from the combined ID/PB controllers on the available target machine was about 0.14 *ms*, which allowed real-time computations.

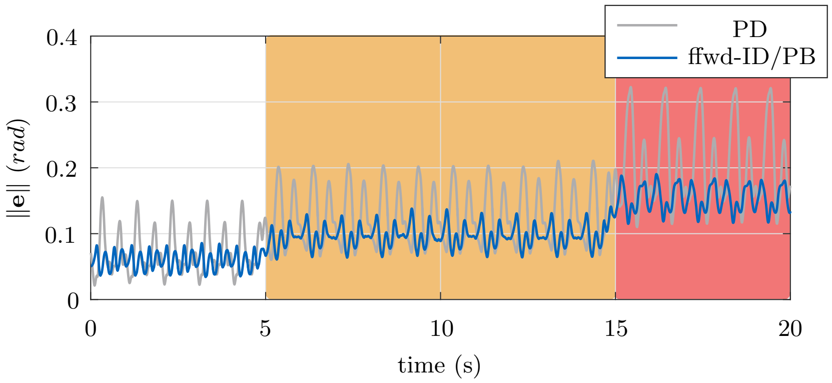


Figure 4.13: Tracking test for a sawing motion with online reduction of gains. Starting from the gains of the first experiment, the tuning matrix $\boldsymbol{\Lambda}$ is reduced from (4.55) during motion of 25% (orange area) and of 50% (red area). The trajectory used for this test is in the appendix.

4.9 Summary

In this chapter, existing and novel controllers for elastic-joint robots have been presented, with a special focus on modular reconfigurable arms. For this reason, the efficient numerical implementability of the schemes has been considered to be a necessary condition, to allow the automatic controller design in conjunction with the framework of Chapter 2.

Simple regulators have been recalled because they represent an attractive solution due to their implementation simplicity, when high precision tracking is not required. Instead, when considering precise trajectory tracking control, available methods that can be efficiently implemented are the feedforward/feedback control approach, the classical inverse-dynamics, and the recently proposed combined ID/PB schemes. While the first provides local stability, the others can guarantee global tracking. Among the control approaches that provide tracking with global stability, the combined ID/PB approach has been shown to provide remarkable robustness to model mismatches with respect to the classical inverse-dynamics scheme in simulations. Thanks to the fact that both have the same efficient numerical computability, the combined ID/PB control approach is the preferable choice. The low total execution time experienced in the experiments with a complex robot arm with 7 degrees-of-freedom has additionally shown the online applicability of these model-based schemes with current technology.

Even though global tracking controllers should be generally preferred, the locally stable trackers presented in this chapter are surely attractive, since they are very effective when accurate acceleration and jerk feedback is not available. However, worse performances in terms of the damping of residual link-side oscillations are expected for the feedforward/feedback approach that uses feedback of motor-side variables only.

From simulations and experiments, the importance of the accurate estimation of accelerations and jerks clearly emerged. Practical solutions have been adopted in this thesis with good results. However, the development of robust observers that can guarantee bounded estimation errors when considering the actuator limitations is certainly an important improvement that can be considered for future research.

Chapter 5

Conclusion and Future Directions

Conclusion

New methods for solving fundamental problems in the control of modular robot manipulators have been presented. Contrary to most previously published approaches that foster the application of decentralized control techniques, this thesis shows that the applicability of centralized model-based techniques is possible. In particular, it was shown both theoretically and with experiments that this is possible in a systematic way, even when modules are significantly different from one another (heterogeneous), when there are external disturbances, when modules have significant uncertainty in their parameters, and when non-negligible elasticity in the joints is present.

The core control framework, which provides a systematic way for automatic centralized controller design, has also been shown to allow the use of model-based control schemes for modular robots which can be frequently and arbitrarily re-assembled. The framework introduced here has the benefit of being complete and systematic. Its completeness is achieved because it includes kinematics (accounting also for special cases which have not been completely handled by previous approaches), dynamics, and control aspects. Its systematicity is provided by the fact that it can be implemented as a step-by-step procedure for different modular robot setups with heterogeneous modules. The experimental results have shown applicability of the framework with different modular setups and commercially available hardware, thus showing readiness of this approach for applications outside research as well. The combination of this framework with algorithms for quickly finding the task-optimal assembly such as [132] [53], and for programming by demonstration such as [119], enables the realization of a novel framework for flexible manufacturing as shown in [130].

With the aim of guaranteeing robustness to the automatically generated control schemes with the above mentioned framework, a novel approach for robust control has been introduced. The benefit of this new robust control approach exceeds the domain of modular reconfigurable robots and also has implications for the robust control of classical rigid robots. In fact, the approach presented allows the automatic deployment of a controller with guaranteed robust performance, without the need for empirical and time-consuming estimations of bounds of non-linear model terms (which are also state dependent and complicate the problem in practice significantly). Such bounds can instead be computed automatically using interval arithmetic, starting from known intervals in which the real dynamical parameters of the arms can lie. Interestingly, with the aim of reducing the computational complexity of this scheme, a new interval-arithmetic-based Newton-Euler algorithm has been introduced. Thanks to this algorithm, the computational complexity of the overall robust control scheme has been made linear in the number of joints. Performance of this approach has been shown to overcome those of existing reference methods in terms of reduced conservativity and deployment simplicity. Real world applicability and effectiveness has been verified with experiments. The interval-arithmetic-based Newton-Euler algorithm has also been shown to be useful as the base for robust scaling of trajectories and for sensor-less collision detection strategies. The first results in this direction have been collected in [133].

The recent trend of the intentional inclusion of elasticity in the joints has significantly increased the complexity of the motion control problem of modern robot arms. A large number of results have been presented in the literature for such control problems so far. In this thesis, the most suitable existing approaches that can be used for modular arms in conjunction with the framework of Chapter 2 have been described. Of particular importance for a control scheme to be considered for automatic controller generation is the possibility for its efficient numerical implementation. In contrast, symbolic manipulation requirements would impose significant computational resources and dedicated software packages, typically not available for a central control unit of a robot. While efficient numerical computability of control schemes has been possible for regulation control purposes as for the rigid joint ones, the efficient numerical computability of global tracking control schemes has been enabled only recently. Latest algorithms for this purpose indeed enable the efficient numerical implementation of inverse-dynamics controllers for elastic-joint robots as the standard N-E algorithm did for rigid robots in former times. Particularly relevant for global tracking control is a new control scheme that has been presented in this thesis, which provides enhanced robustness with respect to the classical inverse-dynamics scheme. This new controller merges the efficient computability of the inverse-dynamics schemes with the enhanced robustness from passivity-based control. The robustness of the resulting combined inverse-dynamics/passivity-based controller has been analyzed in detail. Remarkably, this analysis has shown that a suitable selection of the gains can theoretically make the tracking error to become arbitrarily small, even though model uncertainties and input disturbances are considered. The application of this novel scheme to a reconfigurable robot test bed and to a 7 degrees-of-freedom arm has verified its effectiveness and real-world applicability.

Future Directions

Logical continuations of this work may involve the application of the framework proposed for automatic centralized controller design to deploy automatically advanced control schemes such as impedance, admittance, and hybrid force-position controllers. In addition, the problem of automatic deployment of computationally efficient inverse kinematics schemes that beneficially exploit possible redundancies and globally consider avoidance of obstacles, self-collisions, and joint limits is as challenging as important for future research. Clearly, for this purpose the automatically generated models could be used for solving optimal control problems after assembly. The interval arithmetic robust control approach presented in this thesis would benefit from the development of strategies for properly stopping the integral actions that may cause excessive amplification of measurement noise. It would additionally be interesting to consider this novel control approach for the robust control of other systems having highly nonlinear, coupled, and uncertain dynamics. The basic idea is that the use of interval arithmetic for computing worst case perturbations online may allow the automatic deployment of formally robust controllers for other engineering fields as well. Enhancement of the combined inverse-dynamics/passivity-based controller proposed could be introduced by considering the case of nonlinear springs and the use of interval arithmetic to simplify tuning when dynamics are uncertain. In addition, the simulations, experiments, and the theoretical robustness analysis highlighted the importance of having robust observers for the estimation of joint accelerations and jerks given position, velocity readings, and an uncertain robot model. Having a method that can formally guarantee small bounds for the estimation errors of accelerations and jerks would simplify tuning efforts significantly.

Appendix A

Efficient Recursive Algorithms

A.1 Standard and Modified Recursive Newton-Euler Algorithms

The Newton-Euler method for modeling the dynamics of robots is based on the balance of the fundamental equations of motion of each body that composes the structure. A computationally efficient (recursive) variant of the Newton-Euler method was first proposed in [71] with a complexity that grows linearly in the number of the joints $\big(\mathcal{O}(N)\big)$. The standard recursive Newton-Euler algorithm (NEA) studies the manipulator link by link and is composed of two recursions, one for kinematics (forward recursion) and one for the balance of forces and torques (backward recursion). An implementation of the standard algorithm is detailed next in Algorithm 2, which has been adapted from [96, Section 7]. This algorithm receives as input the vectors of joint positions $\mathbf{q}$, velocities $\dot{\mathbf{q}}$, accelerations $\ddot{\mathbf{q}}$, the gravity vector $\mathbf{g}$, the DH table $\mathbf{DH}$, and the dynamical parameters of the arm $\mathbf{DynPar}$. It provides as output the vector of forces/torques $\mathbf{u}$, which results in

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{c}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{f}(\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) = \mathbf{u} = NEA_g\big(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}},\mathbf{DH},\mathbf{DynPar}\big).$$

In Algorithm 2, with reference to (2.15), (2.16), and Figure 2.6a, $\mathbf{f}_i^i$ and $\mathbf{n}_i^i$ are action-reaction force and torque vectors exerted on the ith link, $\boldsymbol{\omega}_i^i$ is its angular velocity, $\mathbf{a}_i^i$ is the linear acceleration of the link-fixed frame, $\mathbf{a}_{c,i}^i$ is the linear acceleration of its center of mass, $\mathbf{r}_{D_{i-1},D_i}^i$ is the vector between two consecutive frames, m_i is the mass, and $\mathbf{I}_i^i$ is the inertia tensor. Further, $f_i(\dot{q}_i)$, $\sigma_{r,i}$, and $I_{m,i}$ are the friction model, the gear ratio, and the rotor inertia, respectively. The superscript of vectors denotes in which frame they are defined. The influence of gravity is included in the base frame's acceleration during initialization of the forward recursion (line 3). The presence of gravity can be neglected by setting $\mathbf{g} = 0$ in the algorithm; such a case is denoted as NEA_0 in this thesis.

Enhanced versions of this algorithm have been presented: for example, in [94] the authors consider inertia effects of the rotors and in [34] the authors provide a modified version of the algorithm for fault detection and passivity-based control. In particular, the algorithm proposed in [34] allows the computation of the matrix $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ that satisfies the property in (2.2). As originally proposed, the approach is suitable for robots with revolute joints. A small extension to handle prismatic joints in addition has been introduced in the author's work [126]. This algorithm is presented next in Algorithm 3. Its structure is equivalent to the standard algorithm except for the introduction of auxiliary angular velocity vectors ($\dot{\mathbf{q}}_a$ and $\boldsymbol{\omega}_a$), which are used to split the quadratic dependence of the term $\mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}$ into $\mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_a) = \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}_a$.

The models generated by the proposed extension for prismatic joints allow the property in (2.2) to be fulfilled; this verification was perfomed by adopting a similar testing approach that authors in [34] use for revolute joints. The fully general symbolic models were tested for all combinations of revolute and prismatic joints (up to 4 degrees-of-freedom for the high computational burden) using the *Symbolic Math Toolbox*® in MATLAB. With this approach, all tests have been successful. The modified recursive N-E algorithm (NEA^*) described in Algorithm 3 provides

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}_a + \mathbf{f}(\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) = \mathbf{u} = NEA_g^*\big(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_a, \ddot{\mathbf{q}}, \mathbf{DH}, \mathbf{DynPar}\big).$$

Algorithm 2 Standard recursive N-E algorithm: $NEA_g(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbf{DH}, \mathbf{DynPar})$.

```
Input: $\mathbf{q}$, $\dot{\mathbf{q}}$, $\ddot{\mathbf{q}}$, $\mathbf{DynPar}$, $\mathbf{DH}$, and the gravity vector $\mathbf{g}$
Output: $\mathbf{u}$
1: Initialize number of links N (from $\mathbf{DH}$ )
2: Initialize $\boldsymbol{\omega}_0^0$, $\dot{\boldsymbol{\omega}}_0^0$, and $\mathbf{a}_0^0$ to zero and $\mathbf{z}_0 \leftarrow [0, 0, 1]^T$
3: Include effect of gravity $\mathbf{a}_0^0 \leftarrow \mathbf{a}_0^0 - \mathbf{g}$
4: for $i = 1$ to $N$ do
5:     $\mathbf{R}_i^{i-1} \leftarrow$ compute (2.3) using $\mathbf{DH}$ and $q_i$
6:     $\mathbf{R}_{i-1}^i \leftarrow transpose\left(\mathbf{R}_i^{i-1}\right)$
7:     $\mathbf{z}_i \leftarrow \mathbf{R}_{i-1}^i \mathbf{z}_0$
8:     $\mathbf{r}_{D_{i-1},D_i}^i \leftarrow$ compute (2.26) using $\mathbf{DH}$ and $q_i$
9: end for
```

```
for $i = 1$ to $N$ do                                              ▷ Start of forward recursion
    if $i^{th}$ joint is revolute then
        $\boldsymbol{\omega}_i^i \ \leftarrow \mathbf{R}_{i-1}^i \left(\boldsymbol{\omega}_{i-1}^{i-1} + \dot{q}_i \mathbf{z}_0\right)$
        $\dot{\boldsymbol{\omega}}_i^i \ \leftarrow \mathbf{R}_{i-1}^i \left(\dot{\boldsymbol{\omega}}_{i-1}^{i-1} + \ddot{q}_i \mathbf{z}_0 + \boldsymbol{\omega}_{i-1}^{i-1} \times \dot{q}_i \mathbf{z}_0\right)$
     $\mathbf{a}_i^i \leftarrow \mathbf{R}_{i-1}^i \mathbf{a}_{i-1}^{i-1} + \dot{\boldsymbol{\omega}}_i^i \times \mathbf{r}_{D_{i-1},D_i}^i + \boldsymbol{\omega}_i^i \times \left(\boldsymbol{\omega}_i^i \times \mathbf{r}_{D_{i-1},D_i}^i\right)$
    else $i^{th}$ joint is prismatic
        $\boldsymbol{\omega}_i^i \ \leftarrow \mathbf{R}_{i-1}^i \boldsymbol{\omega}_{i-1}^{i-1}$
        $\dot{\boldsymbol{\omega}}_i^i \ \leftarrow \mathbf{R}_{i-1}^i \dot{\boldsymbol{\omega}}_{i-1}^{i-1}$
     $\mathbf{a}_i^i \leftarrow \mathbf{R}_{i-1}^i \mathbf{a}_{i-1}^{i-1} + \dot{\boldsymbol{\omega}}_i^i \times \mathbf{r}_{D_{i-1},D_i}^i + \boldsymbol{\omega}_i^i \times \left(\boldsymbol{\omega}_i^i \times \mathbf{r}_{D_{i-1},D_i}^i\right) + 2\boldsymbol{\omega}_i^i \times \dot{q}_i \mathbf{z}_i + \ddot{q}_i \mathbf{z}_i$
    end if
    $\mathbf{a}_{c,i}^i \leftarrow \mathbf{a}_i^i + \left(\dot{\boldsymbol{\omega}}_i^i \times \mathbf{r}_{D_i,C_i}^i\right) + \left(\boldsymbol{\omega}_i^i \times \left(\boldsymbol{\omega}_i^i \times \mathbf{r}_{D_i,C_i}^i\right)\right)$
end for
Initialize $\mathbf{f}_{N+1}^{N+1}$, $\mathbf{n}_{N+1}^{N+1}$, and $\mathbf{R}_{N+1}^N$
for $i = N$ to $1$ do                                              ▷ Start of backward recursion
    $\mathbf{F}_i^i \leftarrow m_i \mathbf{a}_{c,i}^i$
    $\mathbf{f}_i^i \ \leftarrow \left(\mathbf{R}_{i+1}^i \mathbf{f}_{i+1}^{i+1}\right) + \mathbf{F}_i^i$
    $\mathbf{n}_i^i \leftarrow \left(\mathbf{R}_{i+1}^i \mathbf{n}_{i+1}^{i+1}\right) + \left(\mathbf{r}_{D_i,C_i}^i \times \mathbf{F}_i^i\right) + \left(\mathbf{r}_{D_{i-1},D_i}^i \times \mathbf{f}_i^i\right) + \left(\mathbf{I}_i^i \dot{\boldsymbol{\omega}}_i^i\right) + \left(\boldsymbol{\omega}_i^i \times \left(\mathbf{I}_i^i \boldsymbol{\omega}_i^i\right)\right)$
    if $i^{th}$ joint is revolute then
        $u_i \leftarrow \mathit{transpose}\left(\mathbf{n}_i^i\right) \mathbf{z}_i + f_i(\dot{q}_i) + \sigma_{r,i}^2 I_{m,i} \ddot{q}_i$
    else $i^{th}$ joint is prismatic
        $u_i \leftarrow \mathit{transpose}\left(\mathbf{f}_i^i\right) \mathbf{z}_i + f_i(\dot{q}_i) + \sigma_{r,i}^2 I_{m,i} \ddot{q}_i$
    end if
end for
```

Algorithm 3 Modified recursive N-E algorithm: $NEA_g^*(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_a, \ddot{\mathbf{q}}, \mathbf{DH}, \mathbf{DynPar})$.

```
Input: $\mathbf{q}$, $\dot{\mathbf{q}}$, $\dot{\mathbf{q}}_a$, $\ddot{\mathbf{q}}$, $\mathbf{DynPar}$, $\mathbf{DH}$, and the gravity vector $\mathbf{g}$
Output: $\mathbf{u}$
Initialize number of links N (from DH )
Initialize $\boldsymbol{\omega}_{a,0}^0$, $\boldsymbol{\omega}_0^0$, $\dot{\boldsymbol{\omega}}_0^0$, and $\mathbf{a}_0^0$ to zero and $\mathbf{z}_0 \ \leftarrow [0, 0, 1]^T$
Include effect of gravity $\mathbf{a}_0^0 \leftarrow \mathbf{a}_0^0 - \mathbf{g}$
for $i = 1$ to $N$ do
    $\mathbf{R}_i^{i-1} \ \leftarrow$ compute (2.3) using DH and $q_i$
    $\mathbf{R}_{i-1}^i \ \leftarrow \mathit{transpose}\left(\mathbf{R}_i^{i-1}\right)$
    $\mathbf{z}_i \qquad \leftarrow \mathbf{R}_{i-1}^i \mathbf{z}_0$
    $\mathbf{r}_{D_{i-1},D_i}^i \leftarrow$ compute (2.26) using DH and $q_i$
end for
```

```
for $i = 1$ to $N$ do                                                  ▷ Start of forward recursion
    if $i^{th}$ joint is revolute then
        $\boldsymbol{\omega}_i^i \leftarrow \mathbf{R}_{i-1}^i \left(\boldsymbol{\omega}_{i-1}^{i-1} + \dot{q}_i \mathbf{z}_0\right)$
        $\boldsymbol{\omega}_{a,i}^i \leftarrow \mathbf{R}_{i-1}^i \left(\boldsymbol{\omega}_{a,i-1}^{i-1} + \dot{q}_{a,i}\, \mathbf{z}_0\right)$
        $\dot{\boldsymbol{\omega}}_i^i \leftarrow \mathbf{R}_{i-1}^i \left(\dot{\boldsymbol{\omega}}_{i-1}^{i-1} + \ddot{q}_i \mathbf{z}_0 + \boldsymbol{\omega}_{a,i-1}^{i-1} \times \dot{q}_i \mathbf{z}_0\right)$
        $\mathbf{a}_i^i \leftarrow \mathbf{R}_{i-1}^i \mathbf{a}_{i-1}^{i-1} + \dot{\boldsymbol{\omega}}_i^i \times \mathbf{p}_{i-1,i}^i + \boldsymbol{\omega}_i^i \times \left(\boldsymbol{\omega}_{a,i}^i \times \mathbf{r}_{D_{i-1},D_i}^i\right)$
    else $i^{th}$ joint is prismatic
        $\boldsymbol{\omega}_i^i \leftarrow \mathbf{R}_{i-1}^i \boldsymbol{\omega}_{i-1}^{i-1}$
        $\boldsymbol{\omega}_{a,i}^i \leftarrow \mathbf{R}_{i-1}^i \boldsymbol{\omega}_{a,i-1}^{i-1}$
        $\dot{\boldsymbol{\omega}}_i^i \leftarrow \mathbf{R}_{i-1}^i \dot{\boldsymbol{\omega}}_{i-1}^{i-1}$
        $\mathbf{a}_i^i \leftarrow \mathbf{R}_{i-1}^i \mathbf{a}_{i-1}^{i-1} + \dot{\boldsymbol{\omega}}_i^i \times \mathbf{r}_{D_{i-1},D_i}^i + \boldsymbol{\omega}_i^i \times \left(\boldsymbol{\omega}_{a,i}^i \times \mathbf{r}_{D_{i-1},D_i}^i\right)$
            $+\boldsymbol{\omega}_{a,i}^i \times \dot{q}_i \mathbf{z}_i + \boldsymbol{\omega}_i^i \times \dot{q}_{a,i} \mathbf{z}_i + \ddot{q}_i \mathbf{z}_i$
    end if
    $\mathbf{a}_{c,i}^i \leftarrow \mathbf{a}_i^i + \left(\dot{\boldsymbol{\omega}}_i^i \times \mathbf{r}_{D_i,C_i}^i\right) + \left(\boldsymbol{\omega}_i^i \times \left(\boldsymbol{\omega}_{a,i}^i \times \mathbf{r}_{D_i,C_i}^i\right)\right)$
end for
Initialize $\mathbf{f}_{N+1}^{N+1}$, $\mathbf{n}_{N+1}^{N+1}$, and $\mathbf{R}_{N+1}^{N}$
for $i = N$ to $1$ do                                                  ▷ Start of backward recursion
    $\mathbf{F}_i^i \leftarrow m_i \mathbf{a}_{c,i}^i$
    $\mathbf{f}_i^i \leftarrow \left(\mathbf{R}_{i+1}^i \mathbf{f}_{i+1}^{i+1}\right) + \mathbf{F}_i^i$
    $\mathbf{n}_i^i \leftarrow \left(\mathbf{R}_{i+1}^i \mathbf{n}_{i+1}^{i+1}\right) + \left(\mathbf{r}_{D_i,C_i}^i \times \mathbf{F}_i^i\right) + \left(\mathbf{r}_{D_{i-1},D_i}^i \times \mathbf{f}_i^i\right) + \left(\mathbf{I}_i^i \dot{\boldsymbol{\omega}}_i^i\right) + \left(\boldsymbol{\omega}_{a,i}^i \times \left(\mathbf{I}_i^i \boldsymbol{\omega}_i^i\right)\right)$
    if $i^{th}$ joint is revolute then
        $u_i \leftarrow \mathit{transpose}\left(\mathbf{n}_i^i\right) \mathbf{z}_i + f_i(\dot{q}_i) + \sigma_{r,i}^2 I_{m,i} \ddot{q}_i$
    else $i^{th}$ joint is prismatic
        $u_i \leftarrow \mathit{transpose}\left(\mathbf{f}_i^i\right) \mathbf{z}_i + f_i(\dot{q}_i) + \sigma_{r,i}^2 I_{m,i} \ddot{q}_i$
    end if
end for
```

A.2 Interval-Arithmetic-Based Recursive Newton-Euler Algorithm

In the author's work [126], the idea of enhancing a recursive N-E algorithm with interval arithmetic computations has been proposed. This approach allows one to obtain guaranteed over-approximative sets of joint torques/forces arising from uncertain dynamic parameters of the robot links in a computationally efficient way. Considering the preliminaries of Section 3.3, the enhanced algorithm is presented next in Algorithm 4. This algorithm shares the same algorithmic complexity and structure of the standard and modified recursive N-E algorithms presented in Appendix A.1, being composed of two recursions. While the first recursion (starting at line 10) computes the kinematic relations between subsequent links from the basis to the end effector, the second recursion (starting at line 25) runs backward from the end effector to the basis to compute the balance of forces and torques of the Newton-Euler equations of dynamics. Over-approximative estimations of the torques/forces intervals from uncertain dynamic parameters are obtained by direct inclusion of set based operations for those that involve uncertain dynamical parameters. This is included in lines 22-32 of Algorithm 4[1].

Algorithm 4 Interval-arithmetic-based N-E algorithm: $IANEA^{*}_{g}(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_a, \ddot{\mathbf{q}}, \mathbf{DH}, [\boldsymbol{\Delta}])$.

```
Input: $\mathbf{q}$, $\dot{\mathbf{q}}$, $\dot{\mathbf{q}}_a$, $\ddot{\mathbf{q}}$, $[\boldsymbol{\Delta}]$, $\mathbf{DH}$, and the gravity vector $\mathbf{g}$
Output: $[\mathbf{u}]$
 1: Initialize number of links N (from DH )
 2: Initialize $\boldsymbol{\omega}^0_{a,0}$, $\boldsymbol{\omega}^0_0$, $\dot{\boldsymbol{\omega}}^0_0$, and $\mathbf{a}^0_0$ to zero and $\mathbf{z}_0 \leftarrow [0,0,1]^T$
 3: Include effect of gravity $\mathbf{a}^0_0 \leftarrow \mathbf{a}^0_0 - \mathbf{g}$
 4: for $i = 1$ to $N$ do
 5:     $\mathbf{R}^{i-1}_i \leftarrow$ compute (2.3) using DH and $q_i$
 6:     $\mathbf{R}^i_{i-1} \leftarrow transpose\left(\mathbf{R}^{i-1}_i\right)$
 7:     $\mathbf{z}_i \leftarrow \mathbf{R}^i_{i-1}\mathbf{z}_0$
 8:     $\mathbf{r}^i_{D_{i-1},D_i} \leftarrow$ compute (2.26) using DHtab and $q_i$
 9: end for
10: for $i = 1$ to $N$ do                                   ▷ Start of forward recursion
11:     if $i^{th}$ joint is revolute then
12:         $\boldsymbol{\omega}^i_i \leftarrow \mathbf{R}^i_{i-1}\left(\boldsymbol{\omega}^{i-1}_{i-1} + \dot{q}_i\mathbf{z}_0\right)$
13:         $\boldsymbol{\omega}^i_{a,i} \leftarrow \mathbf{R}^i_{i-1}\left(\boldsymbol{\omega}^{i-1}_{a,i-1} + \dot{q}_{a,i}\,\mathbf{z}_0\right)$
14:         $\dot{\boldsymbol{\omega}}^i_i \leftarrow \mathbf{R}^i_{i-1}\left(\dot{\boldsymbol{\omega}}^{i-1}_{i-1} + \ddot{q}_i\mathbf{z}_0 + \boldsymbol{\omega}^{i-1}_{a,i-1} \times \dot{q}_i\mathbf{z}_0\right)$
15:     $\mathbf{a}^i_i \leftarrow \mathbf{R}^i_{i-1}\mathbf{a}^{i-1}_{i-1} + \dot{\boldsymbol{\omega}}^i_i \times \mathbf{r}^i_{D_{i-1},D_i} + \boldsymbol{\omega}^i_i \times \left(\boldsymbol{\omega}^i_{a,i} \times \mathbf{r}^i_{D_{i-1},D_i}\right)$
16:     else $i^{th}$ joint is prismatic
17:         $\boldsymbol{\omega}^i_i \leftarrow \mathbf{R}^i_{i-1}\boldsymbol{\omega}^{i-1}_{i-1}$
18:         $\boldsymbol{\omega}^i_{a,i} \leftarrow \mathbf{R}^i_{i-1}\boldsymbol{\omega}^{i-1}_{a,i-1}$
19:         $\dot{\boldsymbol{\omega}}^i_i \leftarrow \mathbf{R}^i_{i-1}\dot{\boldsymbol{\omega}}^{i-1}_{i-1}$
20:     $\mathbf{a}^i_i \leftarrow \mathbf{R}^i_{i-1}\mathbf{a}^{i-1}_{i-1} + \dot{\boldsymbol{\omega}}^i_i \times \mathbf{r}^i_{D_{i-1},D_i} + \boldsymbol{\omega}^i_i \times \left(\boldsymbol{\omega}^i_{a,i} \times \mathbf{r}^i_{D_{i-1},D_i}\right)$
            $+\boldsymbol{\omega}^i_{a,i} \times \dot{q}_i\mathbf{z}_i + \boldsymbol{\omega}^i_i \times \dot{q}_{a,i}\mathbf{z}_i + \ddot{q}_i\mathbf{z}_i$
21:     end if
22:     $[\mathbf{a}^i_{c,i}] \leftarrow \mathbf{a}^i_i \oplus \left(\dot{\boldsymbol{\omega}}^i_i \otimes [\mathbf{r}^i_{D_i,C_i}]\right) \oplus \left(\boldsymbol{\omega}^i_i \otimes \left(\boldsymbol{\omega}^i_{a,i} \otimes [\mathbf{r}^i_{D_i,C_i}]\right)\right)$
23: end for
```

[1] Please note that $\boldsymbol{\Delta}$ was used instead of **DynPar** for maintaining consistency with the description in Chapter 3.

```
24: Initialize $[\mathbf{f}_{N+1}^{N+1}]$, $[\mathbf{n}_{N+1}^{N+1}]$, and $\mathbf{R}_{N+1}^{N}$
25: for $i = N$ to 1 do                                        ▷ Start of backward recursion
26:     $[\mathbf{F}_i^i] \leftarrow [m_i] \odot [\mathbf{a}_{c,i}^i]$
27:     $[\mathbf{f}_i^i] \leftarrow \left(\mathbf{R}_{i+1}^i \odot [\mathbf{f}_{i+1}^{i+1}]\right) \oplus [\mathbf{F}_i^i]$
28:     $[\mathbf{n}_i^i] \leftarrow \left(\mathbf{R}_{i+1}^i \odot [\mathbf{n}_{i+1}^{i+1}]\right) \oplus \left([\mathbf{r}_{D_i,C_i}^i] \otimes [\mathbf{F}_i^i]\right)$
        $\oplus \left(\mathbf{p}_{i-1,i}^i \otimes [\mathbf{f}_i^i]\right) \oplus \left([\mathbf{I}_i^i] \odot \dot{\boldsymbol{\omega}}_i^i\right) \oplus \left(\boldsymbol{\omega}_{a,i}^i \otimes \left([\mathbf{I}_i^i] \odot \boldsymbol{\omega}_i^i\right)\right)$
29:     if $i^{th}$ joint is revolute then
30:         $[u_i] \leftarrow transpose\left([\mathbf{n}_i^i]\right) \odot \mathbf{z}_i$
31:     else $i^{th}$ joint is prismatic
32:         $[u_i] \leftarrow transpose\left([\mathbf{f}_i^i]\right) \odot \mathbf{z}_i$
33:     end if
34: end for
```

A.3 Computation of the Geometric Jacobian and its Derivative

The classical algorithm for computing the geometric Jacobian composes the matrix column by column as follows

$$\mathbf{J} = \begin{pmatrix} \mathbf{J}_{P_1} & \dots & \mathbf{J}_{P_N} \\ \mathbf{J}_{\omega_1} & \dots & \mathbf{J}_{\omega_N} \end{pmatrix},$$

where $\mathbf{J}_{P_i}, \mathbf{J}_{O_i} \in \mathbb{R}^3$, for $i \in \{1, \dots, N\}$, and exploits the following relation [96]:

$$\begin{pmatrix} \mathbf{J}_{P_i} \\ \mathbf{J}_{\omega_i} \end{pmatrix} = \begin{cases} \begin{pmatrix} \mathbf{z}_{i-1} \times (\mathbf{p}_N - \mathbf{p}_{i-1}) \\ \mathbf{z}_{i-1} \end{pmatrix}, & \text{(revolute joint)}, \\ \begin{pmatrix} \mathbf{z}_{i-1} \\ \mathbf{0} \end{pmatrix}, & \text{(prismatic joint)}. \end{cases} \tag{A.1}$$

In the above relations, $\mathbf{z}_{i-1}$ is the third column of the matrix $\mathbf{R}_{i-1}^0$, $\mathbf{p}_N$ is the position vector of the end effector frame from the base frame, and $\mathbf{p}_{i-1}$ is the position vector of the frame $i-1$ from the base frame. These vectors are all functions of the joint positions; the dependence has been omitted for the sake of brevity. They are computed by subsequently multiplying homogeneous transformation matrices relating subsequent link-fixed frames. This can be found in lines 4-6 of Algorithm 5.

The derivative of the geometric Jacobian over time can be simply computed by applying the time derivative of (A.1) as follows:

$$\dot{\mathbf{J}} = \begin{pmatrix} \dot{\mathbf{J}}_{P_i} \\ \dot{\mathbf{J}}_{\omega_i} \end{pmatrix} = \begin{cases} \begin{pmatrix} (\boldsymbol{\omega}_{i-1} \times \mathbf{z}_{i-1}) \times (\mathbf{p}_N - \mathbf{p}_{i-1}) + \mathbf{z}_{i-1} \times (\dot{\mathbf{p}}_N - \dot{\mathbf{p}}_{i-1}) \\ \boldsymbol{\omega}_{i-1} \times \mathbf{z}_{i-1} \end{pmatrix}, & \text{(revolute joint),} \\ \begin{pmatrix} \boldsymbol{\omega}_{i-1} \times \mathbf{z}_{i-1} \\ \mathbf{0} \end{pmatrix}, & \text{(prismatic joint).} \end{cases}$$

To compute the above, the following relations are used:

$$\boldsymbol{\omega}_i = \begin{cases} \boldsymbol{\omega}_{i-1} + \dot{q}_i \, \mathbf{z}_{i-1}, & \text{(revolute joint),} \\ \boldsymbol{\omega}_{i-1}, & \text{(prismatic joint),} \end{cases}$$

$$\dot{\mathbf{p}}_i = \begin{cases} \boldsymbol{\omega}_i \times (\mathbf{p}_i - \mathbf{p}_{i-1}) + \dot{\mathbf{p}}_{i-1}, & \text{(revolute joint),} \\ \boldsymbol{\omega}_i \times (\mathbf{p}_i - \mathbf{p}_{i-1}) + \dot{\mathbf{p}}_{i-1} + \dot{q}_i \mathbf{z}_{i-1}, & \text{(prismatic joint).} \end{cases}$$

Algorithm 5 Algorithm for efficiently computing the geometric Jacobian and its derivative: $dJacobian(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{DH})$.

```
Input: $\mathbf{q}$, $\dot{\mathbf{q}}$, $\mathbf{DH}$
Output: $\dot{\mathbf{J}}(\mathbf{q}, \dot{\mathbf{q}})$, $\mathbf{J}(\mathbf{q})$
Initialize number of links N (from $\mathbf{DH}$ )
Initialize $\mathbf{p}_0$ to zero, $\mathbf{T}_0 \leftarrow \mathbf{I}_{4\times4}$, and $\mathbf{z}_0 \leftarrow [0, 0, 1]^T$
for $i = 1$ to $N$ do
    $\mathbf{T}_i \leftarrow \mathbf{T}_{i-1}\mathbf{A}_i^{i-1}$ from (2.3) using $\mathbf{DH}$ and $q_i$
    $\mathbf{p}_i \leftarrow \mathbf{T}_i(1:3, 4)$
    $\mathbf{z}_i \leftarrow \mathbf{T}_i(1:3, 1:3)\,\mathbf{z}_0$
end for
for $i = 1$ to $N$ do
    if $i$th joint is revolute then
        $\mathbf{J}_{p,i} \leftarrow \mathbf{z}_{i-1} \times (\mathbf{p}_N - \mathbf{p}_{i-1})$
        $\mathbf{J}_{\omega,i} \leftarrow \mathbf{z}_{i-1}$
        $\mathbf{J}(:, i) \leftarrow [\mathbf{J}_{p,i}^T, \mathbf{J}_{\omega,i}^T]^T$
        $\boldsymbol{\omega}_i \leftarrow \boldsymbol{\omega}_{i-1} + \dot{q}_i\mathbf{z}_{i-1}$
        $\dot{\mathbf{p}}_i \leftarrow \boldsymbol{\omega}_i \times (\mathbf{p}_i - \mathbf{p}_{i-1}) + \dot{\mathbf{p}}_{i-1}$
    else $i$th joint is prismatic
        $\mathbf{J}_{p,i} \leftarrow \mathbf{z}_{i-1}$
        $\mathbf{J}_{\omega,i} \leftarrow \mathbf{0}$
        $\mathbf{J}(:, i) \leftarrow [\mathbf{J}_{p,i}^T, \mathbf{J}_{\omega,i}^T]^T$
        $\boldsymbol{\omega}_i \leftarrow \boldsymbol{\omega}_{i-1}$
        $\dot{\mathbf{p}}_i \leftarrow \boldsymbol{\omega}_i \times (\mathbf{p}_i - \mathbf{p}_{i-1}) + \dot{\mathbf{p}}_{i-1} + \dot{q}_i\mathbf{z}_{i-1}$
    end if
end for
```

```
for $i = 1$ to $N$ do
    if $i$th joint is revolute then
        $\dot{\mathbf{J}}_{p,i} \leftarrow (\boldsymbol{\omega}_{i-1} \times \mathbf{z}_{i-1}) \times (\mathbf{p}_N - \mathbf{p}_{i-1}) + \mathbf{z}_{i-1} \times (\dot{\mathbf{p}}_N - \dot{\mathbf{p}}_{i-1})$
        $\dot{\mathbf{J}}_{\omega,i} \leftarrow \boldsymbol{\omega}_{i-1} \times \mathbf{z}_{i-1}$
        $\dot{\mathbf{J}}(:,i) \leftarrow [\dot{\mathbf{J}}^T_{p,i}, \dot{\mathbf{J}}^T_{\omega,i}]^T$
    else $i$th joint is prismatic
        $\dot{\mathbf{J}}_{p,i} \leftarrow \boldsymbol{\omega}_{i-1} \times \mathbf{z}_{i-1}$
        $\dot{\mathbf{J}}_{\omega,i} \leftarrow \mathbf{0}$
        $\dot{\mathbf{J}}(:,i) \leftarrow [\dot{\mathbf{J}}^T_{p,i}, \dot{\mathbf{J}}^T_{\omega,i}]^T$
    end if
end for
```

Appendix B

Test Trajectories

The experiments presented in Subsections 2.6.3 and 2.6.4 have been executed with point-to-point motions in joint space using 5th order polynomials with null initial/final velocity and acceleration. The specification of these trajectories is collected in Table B.1.

Time	q_1	q_2	q_3	q_4	q_5	q_6
(s)	(rad)	(rad)	(rad)	(rad)	(rad)	(rad)
0	0	0	0	0	0	0
2	-	$\pi/8$	$\pi/4$	$-\pi/8$	$\pi/8$	$-\pi/4$
3	$-\pi/2$	-	-	-	-	-
4	-	$\pi/4$	$\pi/8$	$\pi/8$	$-\pi/4$	$-\pi/8$
6	$\pi/8$	$-\pi/8$	0	$-\pi/4$	$-\pi/2$	0
8	-	$\pi/4$	$\pi/8$	$\pi/8$	$-\pi/4$	$\pi/4$
9	$-\pi/4$	-	-	-	-	-
10	-	0	$-\pi/8$	$\pi/6$	0	$-\pi/8$
12	0	$\pi/10$	0	$\pi/4$	$\pi/8$	$-\pi/6$
14	-	$-pi/3$	$pi/8$	0	$pi/4$	0
15	$\pi/2$	-	-	-	-	-
16	-	$-\pi/8$	$\pi/4$	$-\pi/4$	0	$\pi/6$
18	0	0	0	0	0	0

Table B.1: Specifications of the point-to-point motions for the experiments related to Figures 2.13, 2.14, and 2.15.

The reference trajectories used for the experiments in Subsection 4.8.2 are a result of point-to-point motions in joint space using 7th order polynomials with null initial/final velocity, acceleration, and jerk. The points are specified in Tables B.2, B.3, and B.4 for the experiments related to Figures 4.11, 4.12, and 4.13 respectively.

Time	q_1	q_2	q_3	q_4	q_5	q_6	q_7
(s)	(rad)	(rad)	(rad)	(rad)	(rad)	(rad)	(rad)
0	0	0.1	0	0	0	0	$-\pi/8$
2	0	$\pi/8$	0	0	0	0	$-\pi/8$
4	$\pi/2$	$\pi/4$	0	$\pi/2$	$-\pi/2$	$\pi/4$	$-\pi/4$
6	$\pi/4$	$\pi/2$	$\pi/2$	$\pi/2$	$\pi/2$	$-\pi/4$	$-\pi/8$
8	$-\pi/4$	$\pi/4$	$\pi/4$	$\pi/4$	$-\pi/4$	$\pi/4$	$-\pi/4$
9	$\pi/2$	$\pi/4$	0	$\pi/8$	$\pi/4$	$-\pi/4$	$-\pi/10$
10	$\pi/2$	$\pi/2$	$\pi/2$	$\pi/2$	$\pi/4$	$\pi/4$	$-\pi/4$
11	0	$\pi/8$	0	0	0	0	$-\pi/8$

Table B.2: Specifications of the point-to-point motions for the experiment related to Figure 4.11.

Time	q_1	q_2	q_3	q_4	q_5	q_6	q_7
(s)	(rad)	(rad)	(rad)	(rad)	(rad)	(rad)	(rad)
0	0	0.1	0	0	0	0	$-\pi/8$
2	0	$\pi/8$	0	0	0	0	$-\pi/8$
5	$\pi/2$	$\pi/4$	0	$\pi/2$	$-\pi/2$	$\pi/4$	$-\pi/4$
8	$\pi/4$	$\pi/2$	$\pi/2$	$\pi/2$	$\pi/2$	$-\pi/4$	$-\pi/8$
11	$-\pi/4$	$\pi/4$	$\pi/4$	$\pi/4$	$-\pi/4$	$\pi/4$	$-\pi/4$
14	$\pi/2$	$\pi/4$	0	$\pi/8$	$\pi/4$	$-\pi/4$	$-\pi/10$
17	$\pi/2$	$\pi/2$	$\pi/2$	$\pi/2$	$\pi/4$	$\pi/4$	$-\pi/4$
20	0	$\pi/8$	0	0	0	0	$-\pi/8$

Table B.3: Specifications of the point-to-point motions for the experiment related to Figure 4.12.

Time	q_1	q_2	q_3	q_4	q_5	q_6	q_7
(s)	(rad)	(rad)	(rad)	(rad)	(rad)	(rad)	(rad)
0	$-\pi/4$	$\pi/8$	0	$\pi/2$	$\pi/6$	$\pi/4$	$-\pi/8$
0.5	$\pi/12$	$\pi/8$	0	$\pi/4$	$\pi/6$	$\pi/8$	$-\pi/8$
1	$-\pi/4$	$\pi/8$	0	$\pi/2$	$\pi/6$	$\pi/4$	$-\pi/8$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
20	$\pi/12$	$\pi/8$	0	$\pi/4$	$\pi/6$	$\pi/8$	$-\pi/8$

Table B.4: Specifications of the point-to-point motions for the experiment related to Figure 4.13 (sawing motion).

References

[1] C. Abdallah, D. Dawson, P. Dorato, and M. Jamshidi. Survey of robust control for rigid robots. *IEEE Control Systems*, 11(2):24–30, Feb. 1991.

[2] A. Albu-Schaffer and G. Hirzinger. State feedback controller for flexible joint robots: a globally stable approach implemented on DLR's light-weight robots. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pages 1087–1093, 2000.

[3] A. Albu-Schäffer, C. Ott, and G. Hirzinger. A unified passivity-based control framework for position, torque and impedance control of flexible joint robots. *International Journal of Robotic Research*, 26(1):23–39, 2007.

[4] M. Althoff. An introduction to CORA 2015. In *Proc. of the Workshop on Applied Verification for Continuous and Hybrid Systems*, pages 120–151, 2015.

[5] B. Armstrong, O. Khatib, and J. Burdick. The explicit dynamic model and inertial parameters of the PUMA 560 arm. In *Proc. of the IEEE International Conference on Robotics and Automation*, volume 3, pages 510–518, 1986.

[6] Christopher G. Atkeson, Chae H. An, and John M. Hollerbach. Estimation of inertial parameters of manipulator loads and links. *The International Journal of Robotics Research*, 5(3):101–119, 1986.

[7] S. Avila-Becerril, A. Loría, and E. Panteley. Global position-feedback tracking control of flexible-joint robots. In *Proc. of the American Control Conference*, pages 3008–3013, Jul. 2016.

[8] S. Avila-Becerril, A. Loría, and E. Panteley. A separation principle for underactuated lossless lagrangian systems. *IEEE Transactions on Automatic Control*, 62(10):5318–5323, 2016.

[9] L. Baccelliere, N. Kashiri, L. Muratore, A. Laurenzi, M. Kamedula, A. Margan, S. Cordasco, J. Malzahn, and N. Tsagarakis. Development of a human size and strength compatible bi-manual platform for realistic heavy manipulation tasks. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5594–5601, 2017.

[10] L. Bascetta and P. Rocco. Revising the robust-control design for rigid robot manipulators. *IEEE Transactuions on Robotics*, 26(1):180–187, Feb. 2010.

[11] J. Baur, J. Pfaff, H. Ulbrich, and T. Villgrattner. Design and development of a redundant modular multipurpose agricultural manipulator. In *Proc. of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 823–830, July 2012.

[12] P.R. Belanger. Estimation of angular velocity and acceleration from shaft encoder measurements. In *Proc. of the IEEE International Conference on Robotics and Automation*, volume 1, pages 585–592, 1992.

[13] P.R. Belanger, P. Dobrovolny, A. Helmy, and X. Zhang. Estimation of angular velocity and acceleration from shaft-encoder measurements. *International Journal of Robotics Research*, 17(11):1225–1233, 1998.

[14] B. Benhabib, G. Zak, and M.G. Lipton. A generalized kinematic modeling method for modular robots. *Journal of Robotic Systems*, 6(5):545–571, 1989.

[15] S. Bethge, J. Malzahn, N. Tsagarakis, and D. Caldwell. FloBaRoID – A Software Package for the Identification of Robot Dynamics Parameters. In *Proc. of the 26th International Conference on Robotics in Alpe-Adria-Danube Region*, 2017.

[16] Z.M. Bi, W.J. Zhang, I.M. Chen, and S.Y.T. Lang. Automated generation of the D-H parameters for configuration design of modular manipulators. *Robotics and Computer-Integrated Manufacturing*, 23:553–562, 2007.

[17] C. Guarino Lo Bianco. Evaluation of generalized force derivatives by means of a recursive Newton-Euler approach. *IEEE Transactions on Robotics*, 25(4):954–959, Aug. 2009.

[18] B. Brogliato and Carlos Canudas de Wit. *Theory of Robot Control*, chapter Joint space control, pages 59–114. Springer-Verlag London, 1996.

[19] B. Brogliato, R. Ortega, and R. Lozano. Global tracking controllers for flexible-joint manipulators: a comparative study. *Automatica*, 31(7):941–956, 1995.

[20] G. Buondonno and A. De Luca. A recursive Newton-Euler algorithm for robots with elastic joints and its application to control. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5526–5532, Sep. 2015.

[21] G. Buondonno and A. De Luca. Efficient computation of inverse dynamics and feedback linearization for VSA-based robots. *IEEE Robotics and Automation Letters*, 1(2):908–915, Jul. 2016.

[22] Paredis C. J. and Khosla P. Synthesis methodology for task based reconfiguration of modular manipulator systems. In *Proc. of the 6th International Symposium on Robotics Research*, pages 2–5, 1993.

[23] F. Caccavale, S. Chiaverini, and B. Siciliano. Second-order kinematic control of robot manipulators with jacobian damped least-squares inverse: theory and experiments. *IEEE/ASME Transactions on Mechatronics*, 2(3):188–194, 1997.

[24] F. Caccavale, C. Natale, B. Siciliano, and L. Villani. Resolved-acceleration control of robot manipulators: A critical review with experiments. *Robotica*, 16(5):565–573, 1998.

[25] M.G. Catalano, G. Grioli, M. Garabini, F. Bonomo, M. Mancinit, N. Tsagarakis, and A. Bicchi. VSA-CubeBot: A modular variable stiffness platform for multiple degrees of freedom robots. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 5090–5095, May 2011.

[26] I.-M. Chen and Guilin Yang. Automatic model generation for modular reconfigurable robot dynamics. *Journal of Dyn. Sys., Meas., Control*, 120(3):346–352, 1998.

[27] I.-M. Chen, S.H. Yeo, G. Chen, and G. Yang. Kernel for modular robot applications: Automatic modeling techniques. *International Journal of Robotics Research*, 18(2):225–242, 1999.

[28] S. Chiaverini and B. Siciliano. The unit quaternion: A useful tool for inverse kinematics of robot manipulators. *Systems Analisis, Modelling and Simulation*, 35(1):45–60, 1999.

[29] Wankyun Chung, Li-Chen Fu, and Su-Hau Hsu. *Handbook of Robotics*, book section Motion Control, pages 133–159. Springer, 2008.

[30] M.J. Corless and G. Leitmann. Continuous state feedback guaranteeing uniform ultimate boundedness for uncertain dynamic systems. *IEEE Transactions on Automatic Control*, 26(5):1139–1144, 1981.

[31] A. De Luca. Decoupling and feedback linearization of robots with mixed rigid/elastic joints. *International Journal of Robust and Nonlinear Control*, 8(11):965–977, 1998.

[32] A. De Luca. Feedforward/feedback laws for the control of flexible robots. In *Proc. of the IEEE International Conference on Robotics and Automation*, volume 1, pages 233–240, 2000.

[33] A. De Luca and W. Book. *Handbook of Robotics*, book section Robots with Flexible Elements, pages 287–320. Springer, 2008.

[34] A. De Luca and L. Ferrajoli. A modified Newton-Euler method for dynamic computations in robot fault detection and control. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 3359–3364, 2009.

[35] A. De Luca and F. Flacco. A PD-type regulator with exact gravity cancellation for robots with flexible joints. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 317–323, May 2011.

[36] A. De Luca and R. Mattone. Actuator failure detection and isolation using generalized momenta. In *Proc. of the IEEE International Conference on Robotics and Automation*, volume 1, pages 634–639, Sept 2003.

[37] A. De Luca, G. Oriolo, and B. Siciliano. Robot redundancy resolution at the acceleration level. *Laboratory Robotics and Automation*, 4(2):97–106, January 1992.

[38] A. De Luca, B. Siciliano, and L. Zollo. PD control with on-line gravity compensation for robots with elastic joints: Theory and experiments. *Automatica*, 41(10):1809–1819, 2005.

[39] A. De Luca and P. Tomei. *Theory of Robot Control*, chapter Elastic joints, pages 176–217. Springer-Verlag London, 1996.

[40] Carlos Canudas de Wit, Bruno Siciliano, and Georges Bastin, editors. *Theory of Robot Control*. Springer-Verlag London, 1996.

[41] J. Denavit and R.S. Hartenberg. A kinematic notation of lower-pair mechanisms based on matrices. *ASME Journal of Applied Mechanics*, 22:215–221, 1955.

[42] J. Englsberger, A. Werner, C. Ott, B. Henze, M. A. Roa, G. Garofalo, R. Burger, A. Beyer, O. Eiberger, K. Schmid, and A. Albu-Schäffer. Overview of the torque-controlled humanoid robot toro. In *Proc. of the IEEE-RAS International Conference on Humanoid Robots*, pages 916–923, Nov 2014.

[43] R. Featherstone and D. Orin. Robot dynamics: equations and algorithms. In *Proc. of the IEEE International Conference on Robotics and Automation*, volume 1, pages 826–834, 2000.

[44] A. Giusti. https://github.com/andreagiusti/otfctrlmodrob. Software repository OTFCtrlModRob.

[45] S. Gutman. Uncertain dynamical systems–A Lyapunov min-max approach. *IEEE Transactions on Automatic Control*, 24(3):437–443, 1979.

[46] S. Haddadin, F. Huber, and A. Albu-Schäffer. Optimal control for exploiting the natural dynamics of variable stiffness robots. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 3347–3354, May 2012.

[47] G. Hirzinger, N. Sporer, A. Albu-Schäffer, M. Hahnle, R. Krenn, A. Pascucci, and M. Schedl. DLR's torque-controlled light weight robot III-are we reaching the technological limits now? In *Proc. of the IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, volume 2, pages 1710–1716 vol.2, 2002.

[48] J.M. Hollerbach. A recursive lagrangian formulation of maniputator dynamics and a comparative study of dynamics formulation complexity. *IEEE Transactions on Systems, Man and Cybernetics*, 10(11):730–736, Nov 1980.

[49] J.M. Hollerbach. Dynamic scaling of manipulator trajectories. *ASME Journal of Dynamic Systems, Measurement and Control*, 106:102–106, 1984.

[50] P. Hsu, J. Hauser, and S. Sastry. Dynamic control of redundant manipulators. In *1988 American Control Conference*, pages 2135–2139, June 1988.

[51] R. Hui, N. Kircanski, A.A. Goldenberg, C. Zhou, P. Kuzan, J. Wiercienski, D. Gershon, and P. Sinha. Design of the IRIS facility-a modular, reconfigurable and expandable robot test bed. In *Proc. of the IEEE International Conference on Robotics and Automation*, volume 3, pages 155–160, May 1993.

[52] Chen I-M. and Yim M. *Handbook of Robotics*, book section Modular Robots, pages 531–542. Springer, 2016.

[53] E. Icer and M. Althoff. Cost-optimal composition synthesis for modular robots. In *Proc. of the IEEE Conference on Control Applications*, pages 1408–1413, 2016.

[54] A. Jaritz and M.W. Spong. An experimental comparison of robust control algorithms on a direct drive manipulator. *IEEE Transactions on Control Systems Technology*, 4(6):627–640, 1996.

[55] Luc Jaulin, Michel Kieffer, Olivier Didrit, and Eric Walter. *Applied interval analysis: with examples in parameter and state estimation, robust control and robotics.* Springer Verlag, 2001.

[56] R. Kelly. Global positioning of robot manipulators via pd control plus a class of nonlinear integral actions. *IEEE Transactions on Automatic Control*, 43(7):934–938, Jul 1998.

[57] L. Kelmar and P.K. Khosla. Automatic generation of kinematics for a reconfigurable modular manipulator system. In *Proc. of the IEEE International Conference on Robotics and Automation*, volume 2, pages 663–668, Apr. 1988.

[58] M. Keppler, D. Lakatos, C. Ott, and A. Albu-Schäffer. A passivity-based controller for motion tracking and damping assignment for compliantly actuated robots. In *Proc. of the IEEE Conference on Decision and Control*, pages 1521–1528, Dec. 2016.

[59] M. Keppler, D. Lakatos, C. Ott, and A. Albu-Schäffer. A passivity-based approach for trajectory tracking and link-side damping of compliantly actuated robots. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 1079–1086, May 2016.

[60] Hassan K. Khalil. *Nonlinear Systems.* Prentice Hall, Upper Saddle River, New Jersey, 3 edition, 2002.

[61] T. Lamarche and W. H. Zhu. A virtual decomposition control based communication network for modular robots applications. In *Proc. of the International Conference on Computer Communications and Networks*, pages 1321–1326, 2007.

[62] W.W. Li, Z. Melek and C. Clark. Decentralized robust control of robot manipulators with harmonic drive transmission and application to modular and reconfigurable serial arms. *Robotics*, 27:291–302, 2009.

[63] Feng Lin and R.D. Brandt. An optimal control approach to robust control of robot manipulators. *IEEE Transactions on Robotics and Automation*, 14(1):69–77, 1998.

[64] G. Liu. Decomposition-based friction compensation of mechanical systems. *Mechatronics*, 12(5):755 – 769, 2002.

[65] Guangjun Liu, S. Abdul, and AA Goldenberg. Distributed modular and reconfigurable robot control with torque sensing. In *Proc. of the IEEE International Conference on Mechatronics and Automation*, pages 384–389, June 2006.

[66] Guangjun Liu, Yugang Liu, and A.A. Goldenberg. Design, analysis, and control of a spring-assisted modular and reconfigurable robot. *IEEE/ASME Transactions on Mechatronics*, 16(4):695–706, Aug. 2011.

[67] Ming Liu. Decentralized control of robot manipulators: nonlinear and adaptive approaches. *IEEE Transactions on Automatic Control*, 44(2):357–363, Feb. 1999.

[68] A. Loría and S. Avila-Becerril. Output-feedback global tracking control of robot manipulators with flexible joints. In *Proc. of the American Control Conference*, pages 4032–4037, June 2014.

[69] A. Loría and R. Ortega. On tracking control of rigid and flexible joints robots. *Appl. Math. Comput. Sci*, 5(2):101–113, 1995.

[70] A. De Luca and P. Lucibello. A general algorithm for dynamic feedback linearization of robots with elastic joints. In *Proc. of the IEEE International Conference on Robotics and Automation*, volume 1, pages 504–510 vol.1, May 1998.

[71] J.Y.S. Luh, M.W. Walker, and R.P.C. Paul. On-line computational scheme for mechanical manipulators. *ASME Journal of Dynamic Systems, Measurement and Control*, 102:468–474, 1980.

[72] T. Matsumaru. Design and control of the modular robot system: TOMMS. In *Proc. of the IEEE International Conference on Robotics and Automation*, volume 2, pages 2125–2131, May 1995.

[73] E. Meister, E. Nosov, and P. Levi. Automatic onboard and online modelling of modular and self-reconfigurable robots. In *Proc. of the IEEE Conference on Robotics, Automation and Mechatronics*, pages 91–96, 2013.

[74] W.W. Melek and A.A. Goldenberg. Neurofuzzy control of modular and reconfigurable robots. *IEEE/ASME Transactions on Mechatronics*, 8(3):381–389, Sept. 2003.

[75] R. Moore. *Methods and Applications of Interval Analysis.* Society for Industrial and Applied Mathematics, 1979.

[76] Ramon E. Moore, R. Baker Kearfott, and Michael J. Cloud. *Introduction to Interval Analysis.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2009.

[77] A. Müller. Recursive second-order inverse dynamics for serial manipulators. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 2483–2489, 2017.

[78] C. Natale, B. Siciliano, and L. Villani. Spatial impedance control of redundant manipulators. In *Proc. of the IEEE International Conference on Robotics and Automation*, volume 3, pages 1788–1793, 1999.

[79] Alan V. Oppenheim, Ronald W. Schafer, and John R. Buck. *Discrete-time Signal Processing.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1999.

[80] R. Ortega, R. Kelly, and A. Loria. A class of output feedback globally stabilizing controllers for flexible joints robots. *IEEE Transactions on Robotics and Automation*, 11(5):766–770, Oct 1995.

[81] R. Ortega, A. J. Van Der Schaft, I. Mareels, and B. Maschke. Putting energy back in control. *IEEE Control Systems*, 21(2):18–33, Apr. 2001.

[82] Romeo Ortega, Antonio Loría, Per Johan Nicklasson, and Hebertt Sira-Ramirez. *Passivity-based Control of Euler-Lagrange Systems.* Springer-Verlag London, 1998.

[83] C. Ott, A. Albu-Schaffer, A. Kugi, S. Stamigioli, and G. Hirzinger. A passivity based cartesian impedance controller for flexible joint robots - part i: torque feedback and

gravity compensation. In *Proc. of the IEEE International Conference on Robotics and Automation*, volume 3, pages 2659–2665 Vol.3, Apr. 2004.

[84] S. Ozgoli and H. D. Taghirad. A survey on the control of flexible joint robots. *Asian Journal of Control*, 8(4):332–344, 2006.

[85] C. J. J. Paredis, H. B. Brown, and P. K. Khosla. A rapidly deployable manipulator system. In *Proc. of the IEEE International Conference on Robotics and Automation*, volume 2, pages 1434–1439, 1996.

[86] F. C. Park, J. E. Bobrow, and S. R. Ploen. A Lie group formulation of robot dynamics. *International Journal of Robotics Research*, 14(6):609–618, 1995.

[87] Jonghoon Park and Wan Kyun Chung. Analytic nonlinear H_∞ inverse-optimal control for Euler-Lagrange system. *IEEE Transactions on Robotics and Automation*, 16(6):847–854, 2000.

[88] Donald L Peiper. The kinematics of manipulators under computer control. Technical report, Stanford Univ Ca Dept Of Computer Science, 1968.

[89] G. A. Pratt and M. M. Williamson. Series elastic actuators. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 399–406, Aug. 1995.

[90] Z. Qu and J. Dorsey. Robust tracking control of robots by a linear feedback law. *IEEE Transactions on Automatic Control*, 36(9):1081–1084, Sep 1991.

[91] S.M. Rump. INTLAB - INTerval LABoratory. In Tibor Csendes, editor, *Developments in Reliable Computing*, pages 77–104. Kluwer Academic Publishers, Dordrecht, 1999. `http://www.ti3.tuhh.de/rump/`.

[92] S.Chiaverini, G. Oriolo, and Ian D. Walker. *Handbook of Robotics*, book section Kinematically Redundant Manipulators, pages 245–268. Springer, 2008.

[93] D. Schmitz, P. Khosla, and T. Kanade. The CMU reconfigurable modular manipulator system. Technical report, Inst. Software Res., Carnegie Mellon Univ., Pittsburgh, PA, USA, CMU-RI-TR-88-7, 1988.

[94] L. Sciavicco, B. Siciliano, and L. Villani. Lagrange and Newton-Euler dynamic modelling of a gear-driven rigid robot manipulator with inclusion of motor inertia effects. *Advanced Robotics*, 10(3):317–334, 1996.

[95] B. Siciliano. A closed-loop inverse kinematic scheme for on-line joint-based robot control. *Robotica*, 8(3):231–243, 1990.

[96] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. *Robotics: Modelling, Planning and Control*. Springer, 2009.

[97] J.-J. E. Slotine. The robust control of robot manipulators. *The International Journal of Robotics Research*, 4(2):49–64, 1985.

[98] J.-J. E. Slotine and W. Li. On the adaptive control of robot manipulators. *International Journal of Robotics Research*, 6(3):49–59, 1987.

[99] M. W. Spong. Modeling and control of elastic joint robots. *Journal of dynamic systems, measurement, and control*, 109(4):310–318, 1987.

[100] M. W. Spong. Control of flexible joint robots: a survey. Technical report, Coordinated Science Laboratory, University of Illinois, Feb. 1990.

[101] M. W. Spong, J. Y. Hung, S. A. Bortoff, and F. Ghorbel. A comparison of feedback linearization and singular perturbation techniques for the control of flexible joint robots. In *Proc. of the American Control Conference*, pages 25–30, Jun. 1989.

[102] M. W. Spong, Seth Hutchinson, and M. Vidyasagar. *Robot Modeling and Control*. Wiley, 2006.

[103] M. W. Spong, K. Khorasani, and P. Kokotovic. An integral manifold approach to the feedback control of flexible joint robots. *IEEE Journal on Robotics and Automation*, 3(4):291–300, Aug. 1987.

[104] M.W. Spong. On the robust control of robot manipulators. *IEEE Transactions on Automatic Control*, 37(11):1782–1786, 1992.

[105] M.W. Spong and M. Vidyasagar. Robust linear compensator design for nonlinear robotic control. *IEEE Journal of Robotics and Automation*, 3(4):345–351, Aug. 1987.

[106] Y. Tang, M. Tomizuka, G. Guerrero, and G. Montemayor. Decentralized robust control of mechanical systems. *IEEE Transactions on Automatic Control*, 45(4):771–776, 2000.

[107] P. Tomei. A simple PD controller for robots with elastic joints. *IEEE Transactions on Automatic Control*, 36(10):1208–1213, Oct. 1991.

[108] S. Traversaro, S. Brossette, A. Escande, and F. Nori. Identification of fully physical consistent inertial parameters using optimization on manifolds. In *Proc of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5446–5451, Oct. 2016.

[109] N. G. Tsagarakis, M. Laffranchi, B. Vanderborght, and D. G. Caldwell. A compact soft actuator unit for small scale human friendly robots. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 4356–4362, May 2009.

[110] M.-W. Ueberle. *Design, Control, and Evaluation of a Family of Kinesthetic Haptic Interfaces*. PhD thesis, Technische Universität München, 2006.

[111] M. Wenz. *Automatische Konfiguration der Bewegungssteuerung von Industrierobotern*. PhD thesis, Universität Fridericiana zu Karlsruhe, 2008.

[112] C. Wright, A. Buchan, B. Brown, J. Geist, M. Schwerin, D. Rollinson, M. Tesch, and H. Choset. Design and architecture of the unified modular snake robot. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 4347–4354, 2012.

[113] L. Wu, R. Crawford, and J. Roberts. An analytic approach to converting POE parameters into D-H parameters for serial-link robots. *IEEE Robotics and Automation Letters*, 2(4):2174–2179, Oct. 2017.

[114] K. Wurst. The conception and construction of a modular robot system. In *Proc. 16th Int. Sym. Industrial Robotics (ISIR)*, pages 37–44, 1986.

[115] Pan Xinan, Wang Hongguang, Jiang Yong, and Xiao Jizhong. Automatic kinematic modelling of a modular reconfigurable robot. *Transactions of the Institute of Measurement and Control*, 35(7):922–932, 2013.

[116] Guilin Yang and I-Ming Chen. Task-based optimization of modular robot configurations: minimized degree-of-freedom approach. *Mechanism and Machine Theory*, 35(4):517–540, 2000.

[117] M. Yim, K. Roufas, D. Duff, Y. Zhang, C. Eldershaw, and S. Homans. Modular reconfigurable robots in space applications. *Autonomous Robots*, 14(2-3):225–237, 2003.

[118] J. S. Yuan. Closed-loop manipulator control using quaternion feedback. *IEEE Journal on Robotics and Automation*, 4(4):434–440, 1988.

[119] M. J. A. Zeestraten, S. Calinon, and D. G. Caldwell. Variable duration movement encoding with minimal intervention control. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 497–503, 2016.

[120] S. Zenieh and M. Corless. Simple robust r - α tracking controllers for uncertain fully-actuated mechanical systems. *ASME Journal of Dynamic Systems, Measurement, and Control*, 119(4):821–825, 1997.

[121] M. Zhu and Y. Li. Decentralized adaptive fuzzy control for reconfigurable manipulators. In *Proc. of the IEEE Conference on Robotics, Automation and Mechatronics*, pages 404–409, Sept. 2008.

[122] W.-H. Zhu. *Virtual decomposition control: toward hyper degrees of freedom robots*, volume 60. Springer Science & Business Media, 2010.

[123] W.-H. Zhu, T. Lamarche, E. Dupuis, D. Jameux, P. Barnard, and Guangjun Liu. Precision control of modular robot manipulators: The VDC approach with embedded FPGA. *IEEE Transactions on Robotics*, 29(5):1162–1179, Oct. 2013.

Own Publications

[124] A. Giusti and M. Althoff. Automatic centralized controller design for modular and reconfigurable robot manipulators. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3268–3275, 2015.

[125] A. Giusti and M. Althoff. Ultimate robust performance control of rigid robot manipulators using interval arithmetic. In *Proc. of the American Control Conference*, pages 2995–3001, 2016.

[126] A. Giusti and M. Althoff. Efficient computation of interval-arithmetic-based robust controllers for rigid robots. In *Proc. of the First IEEE International Conference on Robotic Computing*, pages 129–135, 2017.

[127] A. Giusti and M. Althoff. On-the-fly control design of modular robot manipulators. *IEEE Transactions on Control Systems Technology*, preprint, 2017.

[128] A. Giusti, J. Malzahn, N.G. Tsagarakis, and M. Althoff. Combined inverse-dynamics/passivity-based control for robots with elastic joints. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 5281–5288, 2017.

[129] A. Giusti, J. Malzahn, N.G. Tsagarakis, and M. Althoff. On the combine inverse-dynamics/passivity-based control of elastic-joint robots. *IEEE Transactions on Robotics (conditionally accepted)*, 2018.

[130] A. Giusti, M.J.A. Zeestraten, E. Icer, A. Pereira, D.G. Caldwell, S. Calinon, and M. Althoff. Towards flexible automation driven by demonstration. *IEEE Robotics & Automation Magazine (under review)*, 2018.

[131] F. Hisch, A. Giusti, and M. Althoff. Robust control of continuum robots using interval arithmetic. In *Proc. of the 20th IFAC World Congress (IFAC-PapersOnLine)*, pages 5660–5665, 2017.

[132] E. Icer, A. Giusti, and M. Althoff. A task-driven algorithm for configuration synthesis of modular robots. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 5203–5209, 2016.

[133] M. Wagner, S.B. Liu, A. Giusti, and M. Althoff. Interval-arithmetic-based trajectory scaling and collision detection for robots with uncertain dynamics. In *Proc. of the Second IEEE International Conference on Robotic Computing (accepted)*, 2018.