



SAPIENZA
UNIVERSITÀ DI ROMA

Faculty of Information Engineering,
Informatics, and Statistics

Dottorato in
Automatica, Bioingegneria e Ricerca Operativa

PhD Thesis

Numerical Solutions for Design and Dynamic Control of Compliant Robots

Academic Advisor
Prof. Alessandro De Luca

Candidate
Gabriele Buondonno

Ciclo XXX

*Assoone as he came home, streyght way Pygmalion did repayre
unto the Image of his wench, and leaning on the bed,
did kisse her. In hir body streyght a warmenesse seemd too spred.
He put his mouth againe to hers, and on her brest did lay
his hand. The Ivory waxed soft: and putting quyght away
all hardnesse, yeelded underneathe his fingars, as wee see
a peece of wax made soft ageinst the Sunne, or drawen too bee
in divers shapes by chaufing it betweene ones handes, and so
to serve to uses.*

Ovid, *Metamorphoses*, 10.280–286

Translation by Arthur Golding, 10.304–312

London, W. Seres, 1567

Acknowledgements

First and foremost, I would like to thank my advisor, Prof. Alessandro De Luca, for his supervision and help; overall, he has guided me for more than six years, from my bachelor's thesis up to this day, always being available, supportive, helpful, and a huge source of knowledge. Though I may now be moving to a new department and though I may have sillily missed his 60th-birthday celebration, I look forward to keep on with our cooperation. Besides him, I would like to thank all my colleagues and friends at DIAG, my home department in Rome, who were my fellow companions during the PhD and before that. In particular, thanks to Claudio Gaz, Massimo Cefalo and Emanuele Magrini for all the good time we spent and all the good work we did together.

I would also like to thank Prof. Jean-Paul Laumond, the unofficial co-advisor of this thesis, who warmly welcomed me in LAAS, gave me the great opportunity to visit it, and granted me the trust to stay there as a full-time researcher. Huge thanks to Nicolas Mansard, Justin Carpentier and Guilhem Saurel for the work we did together, and thanks to all members of the Gepetto team [*sic*] for always being helpful and friendly.

Also, I would like to thank Profs. Antonio Bicchi and Claudio Melchiorri for their time spent reviewing this work and for their encouraging comments.

The pursuit of a doctoral degree is a long and complicated process, made of many different experiences and lots of work, not all of which ends up in the final thesis, despite being just as valuable as that which does. Therefore, I feel obliged to mention all the people who contributed to my professional development, even though this may not be apparent from the present manuscript.

In the first place, thanks to all those in KUKA Corporate Research for allowing me to visit them, and to Matthias Kurze in particular for being a great person to work and also hang out with; although my stay only lasted two weeks, I still regard it as very significant and I remember it with pleasure.

Also, thanks to Antonio Franchi and his group for a profitable cooperation. The work done with them was very compelling, and sure it brought me great satisfaction and thought me valuable lessons. My warmest appreciation certainly goes to Burak Yüksel: what started out as an excellent work collaboration developed into a great friendship and, besides being pleased he shared with me his energy and his insights, I am so grateful to him for being close to me in times of trouble.

Finally, I would like to thank my parents, my brother Fabio and my friends for always being supportive, always comforting me, and always being so patient with my whimsicalities. Thanks to Fabio again for hosting me in Barcellona. Thanks to Adilya for saving me multiple times. Thanks to Ceren for making my time sweet while I was working on this thesis. Thanks to Sophie and Ketı for giving me many practical demonstrations of how difficult a researcher's life can be, but also fun and interesting at the same time.

Abstract

This work is focused on the development of numerical methods for the design and control of robots, with particular emphasis on joint elasticity. First, a general methodology is presented that is able to solve the problem of computing the inverse dynamics of a serial robot manipulator with an arbitrarily large number of elastic joints in a recursive numerical way. The solution algorithm is a generalized version of the standard Newton-Euler approach. The algorithm is presented with numerous extensions and variants, including the extension to variable-stiffness technologies and control applications. Then, an optimization framework is introduced for the design and analysis of biped walkers characterized by elastic joints, with comparative results demonstrating the scope of application of joint compliance in bipedal walking.

Contents

Abstract	v
Motivation	1
Part I Inverse Dynamics Algorithms for Compliant Manipulators	7
1 Introduction	9
1.1 Contribution	9
1.2 Rigid robots and RNEA	11
1.2.1 Kinematic conventions	11
1.2.2 Dynamic modeling	13
1.2.3 Recursive Newton-Euler Algorithm	14
2 EJNEA	17
2.1 Series Elastic Actuators	17
2.1.1 Inverse dynamics	19
2.1.2 Feedback linearization control	21
2.2 The Algorithm EJNEA	22
2.2.1 Forward recursion	23
2.2.2 Backward recursion	25
2.3 Application to Control	27
2.4 Modified Denavit-Hartenberg notation	30
2.5 Spatial vector notation	32
3 Extensions and Variants	35
3.1 Nonlinear springs	35
3.1.1 Inverse dynamics	36
3.1.2 The algorithm N-EJNEA	37
3.2 Variable Stiffness Actuators	37

CONTENTS

3.2.1	Modeling	39
3.2.2	Inverse dynamics	40
3.2.3	Special cases	42
3.2.4	Feedback linearization control	45
3.2.5	The algorithm VSA-NEA	46
3.2.6	Variant for feedback linearization	47
3.3	Friction	48
4	Results	51
4.1	EJNEA	51
4.1.1	Validation	51
4.1.2	Inverse dynamics of KUKA LWR4+	52
4.2	VSA-NEA	56
4.2.1	Inverse dynamics	56
4.2.2	Control	58
 Part II Optimization-Based Actuation Design for Compliant Humanoids		 61
5	Introduction	63
5.1	Contribution	63
6	Framework	67
6.1	Legged robots	67
6.2	Dynamic modeling	69
6.3	The framework	70
6.3.1	Rigid robots	71
6.3.2	Series elastic actuators	73
6.3.3	Parallel elastic actuators	73
6.3.4	Optimization techniques	73
6.3.5	Implementation	74
7	Results	77
7.1	Squared torque cost function	78
7.1.1	Compass walker	78
7.1.2	Full walker	79
7.2	Cost of transport objective function	80
7.2.1	Compass walker	80
7.2.2	Full walker	81
7.3	Serial vs. parallel actuations	82

Conclusions	85
Appendices	91
A Derivation of EJNEA	93
A.1 Forward recursion	94
A.1.1 Angular jerk	94
A.1.2 Angular snap	95
A.1.3 Linear jerk	95
A.1.4 Linear snap	96
A.2 Backward recursion	97
A.2.1 Forces and moments acting on the centers of mass . . .	97
A.2.2 Transmitted forces and moments	99
A.2.3 Elastic torques and motor torques	99
A.3 Modified Denavit-Hartenberg notation	100
B Nonlinear Spring Solutions	103
B.1 Cubic spring	103
B.2 Quadratic spring	107
C Dynamic Model of SEA 2R robot	111
References	115

MOTIVATION

MOTIVATION

Motivation

Ever since the early days of robotics, biology has often served as a source of inspiration for the development of new solutions to challenges faced by researchers in their work. One of the latest and more successful innovations which were brought by this approach is the introduction of physical compliance in robotic mechanisms, as seen in the muscle and tendon structure of animals.

Today, the use of compliant transmission elements in lightweight robotic devices is one of the most prominent trends in current research and applications. This is a shift of paradigm compared to the recent past, when robots were traditionally stiff and the presence of elasticity at the joint level was



Figure 1 The body structure of this frog is intrinsically compliant.



Figure 2 The KUKA LBR iiwa is a commercial lightweight manipulator with compliant joints. Image edited from [96].

considered only a parasitic and undesirable feature in industrial robots, mostly a side effect of using transmission belts or long shafts to relocate actuation, or of resorting to reduction elements with a high reduction ratio, such as the Harmonic Drive™ [95, 81]. Instead, compliant elements are now seen as an important and useful characteristic of robots. Often, they are intentionally introduced in order to achieve the (mutually non-exclusive) goals of realizing more natural and human-like robot behaviors, improving safety in human-robot interaction, optimizing energy consumption, or being capable of explosive tasks with limited actuation torque [40, 41].

MOTIVATION

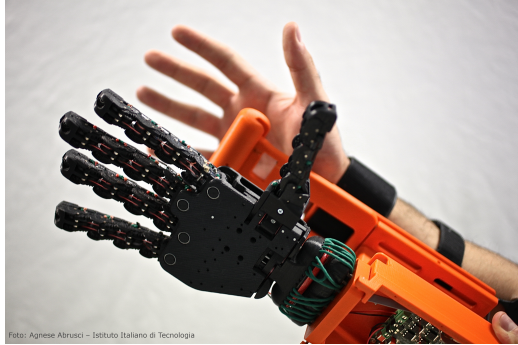


Figure 3 The Pisa/IIT SoftHand. Image from [99].

it has been noticed how intrinsic compliance can considerably simplify the task of interacting with the environment, as it allows the robot configuration to naturally adapt to unmodeled circumstances, demonstrated in grasping by the Pisa/IIT SoftHand [14].

The possibility to exploit elastic energy in order to achieve energy efficiency is also a growing trend. In fact, elastic components are capable of temporarily storing kinetic energy in the form of potential energy, to release it at a later stage, when it is more useful. This possibility, particularly advantageous during cyclical tasks, was demonstrated in [42]. Further studies showed how elastic joints can produce very explosive motions, if conveniently controlled [61].

For these reasons, there is a renewed interest in robots displaying non-negligible compliance at the joints, and it is foreseen that this kind of robots will gain even more importance in the next future.

A first technological response to these objectives is provided by *Series Elastic Actuators* (SEA) [64], as found in [81]. This is a new generation of actuators and transmission elements characterized by non-negligible elasticity of the transmission elements between the motors and the respective links. Their application is not only limited to manipulator arms, but it has been successfully applied to complex bipedal structures such as the TORO robot [27], developed by the German Aerospace Center (DLR). By contrast, in *Parallel Elastic Actuators* [5, 35] the motors directly drive the respective links through *rigid* transmission elements, but in addition the link is also connected to a fixed point through a spring.

The deployment of lightweight manipulators [43, 1, 7], intended for physical human-robot interaction, has turned joint elasticity into a beneficial advantage as mechanical absorbing layer for safety. In fact, in the event of a collision with a human being, an elastic joint would comply with the human's body before special-purpose collision detection and reaction software would take action [19]. On the control side,



Figure 4 The TORO robot. Image edited from [93].

Even beyond this, one can achieve simultaneous control of joint stiffness and link motion by taking inspiration from biomechanical analogies. For this, two motors are used at each robot joint, each of which is connected through compliant transmissions to the driven link. In the last ten years or so, there has been a consistent activity in the design of Variable Stiffness Actuation (VSA) [36]. For single-degree-of-freedom (single-DoF) devices, the two motors work either in antagonistic mode [54], collaborating in a similar way to both control objectives (as in the VSA-I and VSA-II of the University of Pisa [80, 67] and the MACCEPA [82]) or in serial mode, where the smaller, secondary motor is used only to modulate stiffness on line (as in the DLR VS and FVJ devices [91, 90] and the IIT AWAS series [47, 48]). These elementary devices, suitably combined, are currently paving the way to complete VSA-based manipulators and humanoids, sometimes considered the robots of the next generation. Multi-DoF examples include the IIT CompAct arm [50] and the VSA CubeBot modular series from Pisa, now commercialized as low-cost qbmove kits [97]. Recently, the tendency is to go even further and shift to *Variable Impedance Actuators* [83], characterized by controllable stiffness and damping parameters; an example is the complex bimanual robot David [94], formerly known as the DLR Hand-Arm System (HASy) [34]. As an astounding example of applications, this robot has been successfully programmed to such tasks as operating an electric drill to perform holes on a thick concrete surface, a job which would have been impossible with traditional stiff technologies for a human-sized robot, since vibrations would have quickly disrupted its mechanisms.

The advantages provided by joint elasticity do not need to take place at the expense of position accuracy, as long as the effects of joint elasticity are modeled and controlled. In fact, by suitable model-based control designs [20] it is possible to obtain a desired compliant behavior of the robot in response to contact forces, without giving away accuracy in the execution of reference trajectories, provided these are sufficiently smooth. In general, any study about elastic-joint robots cannot leave control out of consideration: clearly, intrinsic compliance is not an advantage *per se*, but it needs to be adequately exploited by control [8]. However, best-performing control algorithms may be very difficult to implement in practice, due to the great complexity of



Figure 5 A small-sized humanoid built out of qbmove kits. Image edited from [97].

MOTIVATION

the robot dynamic models, which may be exceptionally complicated for large robots. Therefore, generic and efficient tools for easily implementing these control schemes in real time are becoming necessary.

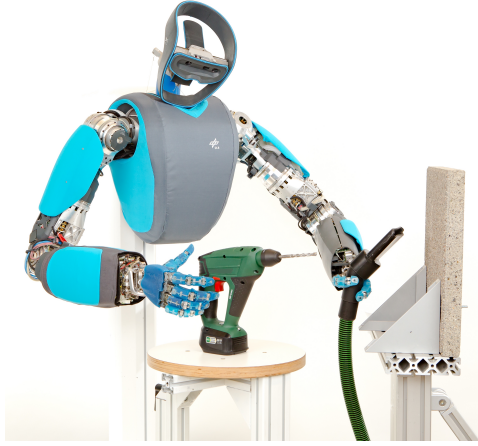


Figure 6 The robot David performing a drilling task. Image edited from [93].

Energetic efficiency in bipedal robot walking is an increasingly important line of research in robotics [16, 6]. Therefore, it is interesting to study the impact of elastic actuators on the energy consumption of robot walking. Traditionally, the design of energy-efficient walkers is a long and elaborated process, essentially left to the ability and experience of the designer. Concerning the sizing of the actuators, important work was done in the VIATORS project [100, 36]. The problem becomes even more sensitive if elastic elements are inserted into the dynamics, the stability analysis of the systems being highly complex. Because of this, the capabilities of VIA-equipped robots are often limited to very simple settings, making them unsuited for more general environments. Therefore, there is a need to develop software which may aid in the design.

In light of the ever-growing importance of robots equipped with elastic joints, a need arises to develop generic and efficient tools for addressing the control challenges posed by this new technology, when it is applied to complex robots with an arbitrary large number of joints. This work addresses the issue in two ways. In Part I different algorithms are introduced that allow the implementation of model-based control schemes of robots equipped with elastic joints. In Part II, an optimization framework is developed which is capable to automatize and speed up the design process of complex compliant bipedal robots, in particular the study of the different types of structures and actuators.

Part I

INVERSE DYNAMICS ALGORITHMS FOR COMPLIANT MANIPULATORS

PART I - COMPLIANT MANIPULATORS

Chapter 1

Introduction

In this Part, the problem of computing the inverse dynamics of a serial robot manipulator with elastic joints in a recursive numerical way is considered. The first solution algorithm, designed for Series Elastic Actuators, is EJNEA [11], a generalized version of the standard Newton-Euler approach. Numerous applications, extensions and variants are also introduced, among which VSA-NEA [12], a further generalization to Variable Stiffness Actuators. The developed tools, some of which are introduced for the first time in this work, are generic, easy to use, and do not require symbolic Lagrangian modeling and customization, thus being of particular interest when the number of joints becomes large.

1.1 Contribution

Robots with elastic joints need more complex modeling and control laws than rigid ones. Dynamic modeling requires the consideration of an extended set of variables in order to describe the distinct, dynamically related positions of the driving motors and of the driven links [20, 79]. Symbolic methods based on Lagrangian formulation have been developed for the systematic derivation of the dynamic model and for computing feedforward torque commands based on inverse dynamics [77, 44]. In turn, control algorithms aimed at high performance for the stable tracking of desired trajectories require extra measurements and a more complex design [24, 60]. In

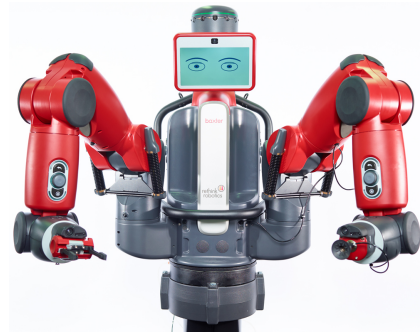


Figure 1.1 The complex compliant robot Baxter. Image from [98].

PART I - COMPLIANT MANIPULATORS

particular, under very reasonable simplifying assumptions, it has been shown that robots with elastic joints can be transformed into exactly linear and decoupled systems by means a nonlinear state feedback. This feedback linearization property is the equivalent of the so-called computed torque method for fully rigid robots. A fundamental difference, however, is that the resulting linear system turns out to be of fourth order (i.e., chains of four input-output integrators at each joint, rather than only two), with more critical stabilization issues. See Sec. 2.1 for a summary of these results.

Model-based control algorithms make heavy use of the Lagrangian dynamic model of the robot, which may become a particularly cumbersome task when for a complex robot (for instance, one with more than 6 joints). For robots equipped with elastic joints, generic and efficient tools for easily implementing these control schemes in real time are becoming necessary. To achieve this, it would be convenient if the implementation did not require the analytic development of a dynamic model in symbolic form.

Responding to this need, this work addresses two very basic problems in dynamics and control for the considered class of robots. The first is the extension to robots with joint elasticity of the classical Recursive Newton-Euler Algorithm (RNEA), originally proposed in [53] for rigid robots. This allows solving the inverse dynamics problem in a numerical way, which is efficient and exact (i.e., without resorting to approximate differentiation of quantities). Chapter 2 is concerned with achieving the same goal for robots equipped with Series Elastic Actuators. Inspired by the works [38, 37] on computing the command torque derivative in rigid robots, we present in Sec. 2.2 our complete Elastic Joints Newton-Euler Algorithm, or EJNEA in short. For a serial robot manipulator with N elastic joints, the algorithm runs still with a linear asymptotic complexity $O(N)$, as in the rigid case, but requires to set up recursions that involve higher order derivatives of motion and force variables. As a result, the feedforward command associated to a desired robot trajectory, defined in terms of link motion, can be obtained without the need to derive an explicit Lagrangian dynamic model in symbolic form. The algorithm is presented in many variants corresponding to the different kinematic conventions in use in the robotics community, so to ease its adoption.

The second problem is the implementation of a feedback linearization control law for robots with elastic joints that should avoid, just as in the rigid case, the need of evaluating at run time the complex expressions of the robot dynamic model, which was previously obtained in symbolic form. In Sec. 2.3, the EJNEA will serve again for the purpose of accurate and exponentially stable tracking of sufficiently smooth trajectories in robots with elastic joints, once it is complemented with a suitable numerical factorization of the link

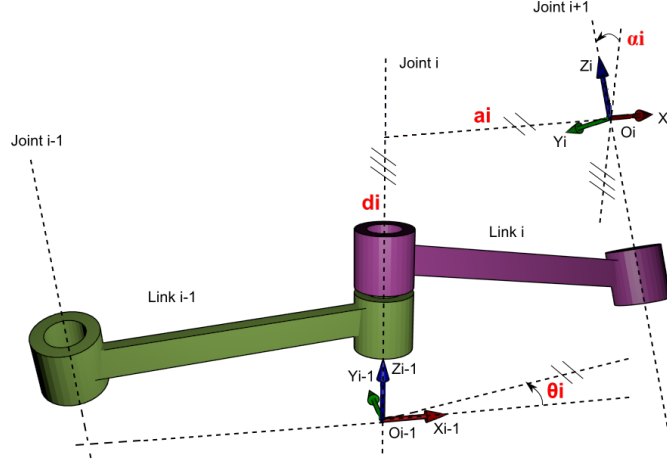


Figure 1.2 Standard Denavit-Hartenberg convention

inertia matrix. The resulting method runs in $O(N^3)$.

Finally, in Ch. 3, the algorithm is further extended to deal with different technologies and use cases: the case of nonlinear stiffness is briefly discussed in Sec. 3.1, with the straightforward extension N-EJNEA; a deep analysis of Variable Stiffness Actuators is conducted in Sec. 3.2, where the algorithm VSA-NEA is introduced; in Sec. 3.3, the way to deal with friction is detailed. Final results are presented in Ch. 4.

1.2 Rigid robots and RNEA

Consider first a rigid robot manipulator with N Degrees of Freedom (DoFs). This corresponds to an open kinematic chain of N rigid links connected through N rigid joints. As standard practice, each link is assigned a reference frame, which is fixed in space with respect to the link itself. Further, the robot configuration is described through a set of *generalized coordinates*.

1.2.1 Kinematic conventions

In order to assign the linked-fixed reference frames and select the generalized coordinates, in this work we employ the *standard* Denavit-Hartenberg convention [25]. In this convention, kinematic frames are located at the end of the link they refer to. Frame placement for a pair of robot links is shown in Fig. 1.2. The position of one frame with respect to the other is described by four kinematic parameters, usually denoted as a_i , α_i , d_i and θ_i . Depending on whether joint i is revolute or prismatic, then θ_i or d_i will be a configura-

PART I - COMPLIANT MANIPULATORS

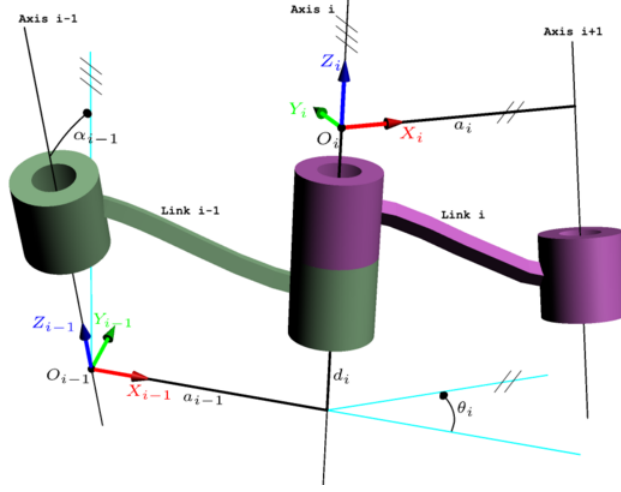


Figure 1.3 Modified Denavit-Hartenberg convention.

tion variable, while the other three parameters will be kinematic constants. The orientation of a frame i with respect to frame $i - 1$, and the position of the origin of frame i with respect to frame $i - 1$, expressed in frame i , are described respectively by the following rotation matrix and vector:

$${}^{i-1}\mathbf{R}_i = \begin{pmatrix} c\theta_i & -c\alpha_i s\theta_i & s\alpha_i s\theta_i \\ s\theta_i & c\alpha_i c\theta_i & -s\alpha_i c\theta_i \\ 0 & s\alpha_i & c\alpha_i \end{pmatrix} \quad {}^i\mathbf{p}_{i,i-1} = \begin{pmatrix} a_i \\ d_i s\alpha_i \\ d_i c\alpha_i \end{pmatrix}. \quad (1.1)$$

Here, we employed the compact notation $s\theta = \sin \theta$, $c\theta = \cos \theta$.

In contrast to this, the *modified* Denavit-Hartenberg convention was introduced by Craig [17]. In the modified convention, kinematic frames are located *at the beginning* of the links. This is shown in Fig. 1.3. The orientation of a frame i with respect to frame $i - 1$, and the position of the origin of frame i with respect to frame $i - 1$, this time expressed in frame $i - 1$, are described respectively by the following rotation matrix and vector

$${}^{i-1}\mathbf{R}_i = \begin{pmatrix} c\theta_i & -s\theta_i & 0 \\ s\theta_i c\alpha_{i-1} & c\theta_i c\alpha_{i-1} & -s\alpha_{i-1} \\ s\theta_i s\alpha_{i-1} & c\theta_i s\alpha_{i-1} & c\alpha_{i-1} \end{pmatrix} \quad {}^{i-1}\mathbf{p}_{i,i-1} = \begin{pmatrix} a_{i-1} \\ -d_i s\alpha_{i-1} \\ d_i c\alpha_{i-1} \end{pmatrix}. \quad (1.2)$$

Some numerical values and notational aspects may change between the two notations, but the two forms are equivalent to each other. In this work, unless explicitly stated, the *standard* convention is always employed.

1.2.2 Dynamic modeling

Once a suitable set of generalized coordinates is chosen, we denote its joint configuration as $\mathbf{q} \in \mathbb{R}^N$. Further, we denote the vector of actuation torques, exerted by the robot motors, as $\boldsymbol{\tau} \in \mathbb{R}^N$. Under standard assumptions [72], the Lagrangian dynamic model of the robot takes the form

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau}, \quad (1.3)$$

where $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{N \times N}$ is the inertia matrix, and is positive definite; $\mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^N$ contains centrifugal and Coriolis forces; $\mathbf{g}(\mathbf{q}) \in \mathbb{R}^N$ represents gravitational forces. The system admits a state space representation where the dynamic state is given as $(\mathbf{q} \ \dot{\mathbf{q}})$. For convenience, we will write compactly all nonlinear bias terms as

$$\mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}). \quad (1.4)$$

The dynamic model is central to the analysis, simulation and control of robot structures. The *inverse dynamics* problem is defined as follows: given a desired link trajectory $\mathbf{q}_d(t) \in \bar{C}^2$, i.e., twice differentiable (where the acceleration may be discontinuous), it is sought to compute the torques $\boldsymbol{\tau}_d$ that realize the desired motion. With a good knowledge of the dynamic model, the computation is straightforward

$$\boldsymbol{\tau}_d = \mathbf{M}(\mathbf{q}_d)\ddot{\mathbf{q}}_d + \mathbf{n}(\mathbf{q}_d, \dot{\mathbf{q}}_d). \quad (1.5)$$

By contrast, the *direct dynamics* problem consists in the following: given the robot state $(\mathbf{q} \ \dot{\mathbf{q}})$ and the input $\boldsymbol{\tau}$, it is desired finding the resulting acceleration. This is of interest mainly for performing simulations. The solution is straightforward from (1.3)

$$\ddot{\mathbf{q}} = \mathbf{M}^{-1}(\mathbf{q})[\boldsymbol{\tau} - \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}})]. \quad (1.6)$$

Now, theoretically, the torque values obtained from (1.5) should be sufficient to execute the desired trajectory $\mathbf{q}_d(t)$. However, in practice, the torque values obtained in (1.5) are only *nominal*. Even the slightest external disturbance or imprecision in the model may cause the robot to deviate from the desired motion, thus requiring different torques. This is the problem of dynamic control.

A very powerful control procedure is *feedback linearization*. This consists in obtaining from a nonlinear system such as (1.3) a *linear* response to some *virtual input*, which is defined by the user. This result allows to employ all control techniques developed in the field of linear system, and, in particular,

PART I - COMPLIANT MANIPULATORS

to obtain exponential error decay and input-output decoupling. In robotics, this technique is also known as *computed torque control* and takes the form

$$\boldsymbol{\tau} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}}_r + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) . \quad (1.7)$$

It can be verified that, plugging (1.7) into (1.3), we obtain the linearized system

$$\ddot{\mathbf{q}} = \ddot{\mathbf{q}}_r \quad (1.8)$$

where $\ddot{\mathbf{q}}_r$ is regarded as the virtual input. The control design is completed by choosing

$$\ddot{\mathbf{q}}_r = \ddot{\mathbf{q}}_d + \mathbf{K}_D(\dot{\mathbf{q}}_d - \dot{\mathbf{q}}) + \mathbf{K}_P(\mathbf{q}_d - \mathbf{q}) \quad (1.9)$$

where $\mathbf{K}_D$ and $\mathbf{K}_P$ are diagonal, positive definite gain matrices.

1.2.3 Recursive Newton-Euler Algorithm

In order to employ the dynamic model (1.3) directly, its terms need to be obtained analytically as symbolic functions of $\mathbf{q}$ and $\dot{\mathbf{q}}$. This needs to be done by hand, for each given robot; this is a long and tedious process, especially for complex manipulators.

A different approach to dynamics is the Newton-Euler method, based on the balance of forces and moments acting on each link. This leads to a set of *recursive* equations allowing a reformulation of the dynamic model which is liable to a generic algorithmic implementation, given some identifiable dynamic parameters. The algorithm is known as Recursive Newton-Euler Algorithm (RNEA), originally proposed in [53]. This method allows solving the inverse dynamics problem in a numerical way, which is efficient and exact, i.e., without resorting to approximate differentiation of quantities.

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) = \text{RNEA}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}) = \boldsymbol{\tau} . \quad (1.10)$$

The algorithm consists of a *forward recursion* and a *backward recursion*. During the forward recursion, the algorithm iterates the robot links from the base of the kinematic structure to its tip, computing each link's velocity and acceleration. During the backward recursion, the robot links are iterated from the tip to the base, propagating the forces and moments acting on the links. From these values, the torques exerted from the motors are extracted.

The dynamic parameters of interest are

- m_i mass of *augmented* link i (i.e., of the link plus the motor mounted on it)

- $\mathbf{p}_{c_i,i}$ position of the center of mass (CoM) of the augmented link i with respect to the origin of frame i
- $\mathbf{I}_i$ central inertia tensor of augmented link i

for a total of 10 dynamic parameters.

In canonical implementations of RNEA, all quantities are conveniently expressed in the *moving* reference frame of the considered link. This is very advantageous, since all dynamic parameters of the robot will be constant in these frames. In this way, computation results greatly simplified. Before proceeding, we recall the usual notation in presence of multiple reference frames. Subscripts represent the link that a quantity is referring to, while superscripts (prefixed) in vectors and matrices indicate the reference frame in which a quantity is expressed. No superscript denotes by default the base frame 0. The rotation matrix denoting the orientation of frame i in frame j is denoted as ${}^j\mathbf{R}_i$. Thus, for instance, the velocity of link i is expressed in the global frame as $\mathbf{v}_i$, while ${}^i\mathbf{v}_i = {}^i\mathbf{R}_0^0\mathbf{v}_i = \mathbf{R}_i^T\mathbf{v}_i$ represents the same quantity as expressed in the i -th frame. With $\hat{\mathbf{z}}_i$, we denote the z -axis unit vector of frame i . It can be easily seen that $\hat{\mathbf{z}}_0 = (0 \ 0 \ 1)^T$, ${}^i\hat{\mathbf{z}}_i = \hat{\mathbf{z}}_0$, $\hat{\mathbf{z}}_i = {}^0\mathbf{R}_i\hat{\mathbf{z}}_0$, and ${}^i\hat{\mathbf{z}}_{i-1} = {}^i\mathbf{R}_{i-1}{}^{i-1}\hat{\mathbf{z}}_{i-1} = {}^i\mathbf{R}_{i-1}\hat{\mathbf{z}}_0$. The final forward and backward recursions of RNEA are outlined next, for the standard Denavit-Hartenberg notation.

The following equations represent the forward recursion, and need to be propagated from the robot base to the tip, for $i = 1, \dots, N$

$${}^i\boldsymbol{\omega}_i = {}^i\mathbf{R}_{i-1}({}^{i-1}\boldsymbol{\omega}_{i-1} + \dot{\theta}_i\hat{\mathbf{z}}_0) \quad (1.11)$$

$${}^i\boldsymbol{\gamma}_i = {}^i\mathbf{R}_{i-1}({}^{i-1}\boldsymbol{\gamma}_{i-1} + \ddot{\theta}_i\hat{\mathbf{z}}_0 + {}^{i-1}\boldsymbol{\omega}_{i-1} \times \dot{\theta}_i\hat{\mathbf{z}}_0) \quad (1.12)$$

$$\begin{aligned} {}^i\mathbf{a}_i &= {}^i\mathbf{R}_{i-1}{}^{i-1}\mathbf{a}_{i-1} + {}^i\boldsymbol{\omega}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{i,i-1}) \\ &\quad + {}^i\boldsymbol{\gamma}_i \times {}^i\mathbf{p}_{i,i-1} + \ddot{d}_i{}^i\hat{\mathbf{z}}_{i-1} + 2\dot{d}_i({}^i\boldsymbol{\omega}_i \times {}^i\hat{\mathbf{z}}_{i-1}) \end{aligned} \quad (1.13)$$

$${}^i\mathbf{a}_{c_i} = {}^i\mathbf{a}_i + {}^i\boldsymbol{\omega}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{c_i,i}) + {}^i\boldsymbol{\gamma}_i \times {}^i\mathbf{p}_{c_i,i} \quad (1.14)$$

while, next, the backward recursion is performed by propagating the following for $i = N, \dots, 1$

$${}^i\mathbf{F}_i = m_i{}^i\mathbf{a}_{c_i} \quad (1.15)$$

$${}^i\mathbf{N}_i = {}^i\mathbf{I}_i{}^i\boldsymbol{\gamma}_i + {}^i\boldsymbol{\omega}_i \times ({}^i\mathbf{I}_i{}^i\boldsymbol{\omega}_i) \quad (1.16)$$

$${}^i\mathbf{f}_i = {}^i\mathbf{R}_{i+1}{}^{i+1}\mathbf{f}_{i+1} + {}^i\mathbf{F}_i \quad (1.17)$$

$${}^i\mathbf{n}_i = {}^i\mathbf{R}_{i+1}{}^{i+1}\mathbf{n}_{i+1} + {}^i\mathbf{p}_{c_i,i} \times {}^i\mathbf{F}_i + {}^i\mathbf{p}_{i,i-1} \times {}^i\mathbf{f}_i + {}^i\mathbf{N}_i \quad (1.18)$$

$$\tau_i = \begin{cases} {}^i\mathbf{n}_i^T {}^i\hat{\mathbf{z}}_{i-1} & \text{if joint } i \text{ is revolute} \\ {}^i\mathbf{f}_i^T {}^i\hat{\mathbf{z}}_{i-1} & \text{if joint } i \text{ is prismatic} \end{cases} \quad (1.19)$$

where the following quantities are defined

PART I - COMPLIANT MANIPULATORS

- $\boldsymbol{\omega}_i$ angular velocity of frame i
- $\boldsymbol{\gamma}_i$ angular acceleration of frame i
- $\boldsymbol{a}_i$ acceleration of the origin of frame i
- $\boldsymbol{a}_{c_i}$ acceleration of the CoM of link i
- $\boldsymbol{F}_i$ total force acting on the CoM of link i
- $\boldsymbol{N}_i$ total moment acting on link i
- $\boldsymbol{f}_i$ total force propagated from link $i - 1$ to link i
- $\boldsymbol{n}_i$ total moment propagated from link $i - 1$ to link i .

Thus, based on the above-established notation, ${}^i\boldsymbol{\gamma}_i = {}^i\boldsymbol{R}_0^0\boldsymbol{\gamma}_i = \boldsymbol{R}_i^T\dot{\boldsymbol{\omega}}_i$.

The forward recursion is initialized with ${}^0\boldsymbol{\omega}_0 = \mathbf{0}$, ${}^0\boldsymbol{\gamma}_0 = \mathbf{0}$, and ${}^0\boldsymbol{a} = -\vec{\boldsymbol{g}}$. Here, $\vec{\boldsymbol{g}}$ is the gravity acceleration vector expressed in frame 0, which is constant, and is included in order to account for gravitational effects. The above equations are valid both for revolute and prismatic joints, with $\theta_i = q_i$ in the former case, and $d_i = q_i$ in the latter. Moreover, if joint i is prismatic, then $\dot{\theta}_i = \ddot{\theta}_i = 0$; if it is revolute, $\dot{d}_i = \ddot{d}_i = 0$; the actual implementation of the algorithm may take into account the joint type in order to avoid multiplications by zero and achieve considerable simplifications.

For the initialization step of the backward recursion, $\boldsymbol{f}_{N+1}$ and $\boldsymbol{n}_{N+1}$ are, respectively, the forces and torques exerted by the end-effector on the environment, i.e., the *opposite* of the external forces and torques acting on the end-effector. If present, these should be passed to the algorithm as an additional input, otherwise they should be set to zero. If the external forces/torques are already expressed in frame N , then ${}^N\boldsymbol{R}_{N+1}$ will be the 3×3 identity matrix.

Once the RNEA is implemented, then, the solution to the inverse dynamics problem can be obtained straightforwardly

$$\boldsymbol{\tau}_d = \text{RNEA}(\boldsymbol{q}_d, \dot{\boldsymbol{q}}_d, \ddot{\boldsymbol{q}}_d) . \quad (1.20)$$

On the other hand, feedback linearization is also very easy to achieve:

$$\boldsymbol{\tau} = \text{RNEA}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}_r) . \quad (1.21)$$

Thus, in the rigid case, both the inverse dynamics and the feedback linearizing control can be computed in an efficient numerical way using directly the RNEA, without any relevant modification.

Chapter 2

EJNEA

2.1 Series Elastic Actuators

In the presence of joint elasticity, the link positions are distinct from the motor positions. Therefore, a robot with N elastic joints will require $2N$ generalized coordinates [20]

$$\Theta = \begin{pmatrix} \mathbf{q} \\ \boldsymbol{\theta}_m \end{pmatrix} \in \mathbb{R}^{2N}, \quad (2.1)$$

where $\boldsymbol{\theta}_m \in \mathbb{R}^N$ is the vector of motor positions, *after* the reduction gears. The difference $\boldsymbol{\phi} \in \mathbb{R}^N$ between the motor position and joint position is termed *deflection*

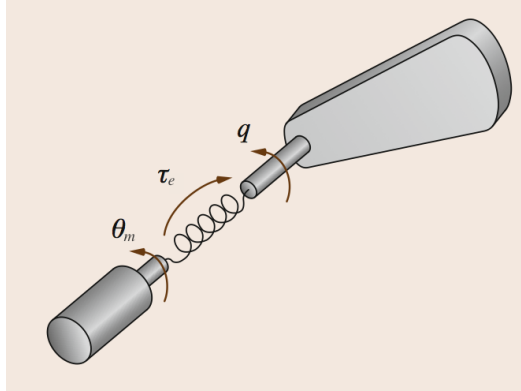


Figure 2.1 An elastic joint. Image edited from [20].

$$\boldsymbol{\phi} = \boldsymbol{\theta}_m - \mathbf{q} \quad (2.2)$$

The dynamic model is usually subject to the following standard assumptions:

1. Linear elasticity. This assumption is satisfied if the joint deflections are kept small. Later in this work, we will see how it is possible to relax this prerequisite.
2. The motors' mass distribution is symmetric around their rotation axes. This is always verified, since the motors would not be mechanically feasible if they were designed otherwise.

PART I - COMPLIANT MANIPULATORS

3. Each motor is located *before* the link it actuates. This is a design characteristic of the robot, which influences the robot's dynamic properties.

Under these assumptions, the standard Lagrangian model can be derived.

The total potential energy is given by the sum of the gravitational one and the elastic one

$$\mathcal{U} = \mathcal{U}_g(\mathbf{q}) + \frac{1}{2} \boldsymbol{\phi}^T \boldsymbol{\Sigma} \boldsymbol{\phi} \quad (2.3)$$

where $\boldsymbol{\Sigma} > 0$ is the constant, diagonal joint stiffness matrix.

In order to derive the total kinetic energy, we need to take the motor rotation into account. The rotor angular velocity, expressed in its own frame (where the z -axis coincides with the rotation axis), is expressed as

$${}^{m_i} \boldsymbol{\omega}_{m_i} = \mathbf{J}_{m,i} \dot{\mathbf{q}} + \begin{pmatrix} 0 \\ 0 \\ n_i \dot{\theta}_{mi} \end{pmatrix} \quad (2.4)$$

where n_i is the i -th reduction ratio.

From this, the total kinetic energy can be computed as

$$\begin{aligned} \mathcal{T} = & \frac{1}{2} \dot{\mathbf{q}}^T \mathbf{M}_L(\mathbf{q}) \dot{\mathbf{q}} + \frac{1}{2} \dot{\boldsymbol{\theta}}_m^T \mathbf{B} \dot{\boldsymbol{\theta}}_m \\ & + \frac{1}{2} \dot{\mathbf{q}}^T [\mathbf{M}_M(\mathbf{q}) + \mathbf{S}(\mathbf{q}) \mathbf{B}^{-1} \mathbf{S}^T(\mathbf{q})] \dot{\mathbf{q}} + \dot{\mathbf{q}}^T \mathbf{S}(\mathbf{q}) \dot{\boldsymbol{\theta}}_m \end{aligned} \quad (2.5)$$

where $\mathbf{M}_L \in \mathbb{R}^{N \times N}$ is the inertia matrix of the robot links, $\mathbf{B} \in \mathbb{R}^{N \times N}$ is the constant, diagonal, positive definite matrix with the drive inertia moments (i.e. the inertias of the motor rotors' around their own spinning axes, multiplied by the squared reduction ratios), $\mathbf{M}_M \in \mathbb{R}^{N \times N}$ contains the rotors' masses and all other rotors' inertial components, while $\mathbf{S} \in \mathbb{R}^{N \times N}$ encodes the inertial couplings between rotors and joints.

Applying Lagrange equations, the full dynamic model is given as

$$\begin{pmatrix} \mathbf{M}(\mathbf{q}) \mathbf{S}(\mathbf{q}) \\ \mathbf{S}^T(\mathbf{q}) \mathbf{B} \end{pmatrix} \begin{pmatrix} \ddot{\mathbf{q}} \\ \ddot{\boldsymbol{\theta}}_m \end{pmatrix} + \begin{pmatrix} \mathbf{c}_1(\mathbf{q}, \dot{\mathbf{q}}, \dot{\boldsymbol{\theta}}_m) + \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) + \boldsymbol{\Sigma}(\mathbf{q} - \boldsymbol{\theta}_m) \\ \mathbf{c}_2(\mathbf{q}, \dot{\mathbf{q}}) + \boldsymbol{\Sigma}(\boldsymbol{\theta}_m - \mathbf{q}) \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \boldsymbol{\tau} \end{pmatrix} \quad (2.6)$$

where $\mathbf{M} = \mathbf{M}_M + \mathbf{M}_L + \mathbf{S} \mathbf{B}^{-1} \mathbf{S}^T$, and $\mathbf{c}_1$ and $\mathbf{c}_2$ are additional Coriolis terms due to the inertial coupling between rotors and links. Additional friction terms can be included, but we will leave them out of consideration for the moment. The state of the system is $4N$ -dimensional and is given, e.g., by $(\mathbf{q} \ \boldsymbol{\theta}_m \ \dot{\mathbf{q}} \ \dot{\boldsymbol{\theta}}_m)$. Further details can be found in [20].

The above model, though being the most complete, is quite difficult to employ directly. This is mostly due to the dynamic couplings between rotors

and links. Some notable results were achieved (for instance, it was shown in [23, 24] that it can be dynamically feedback linearized) but in general it is preferred to work with a slightly simplified model, which is easier to work with.

The usual simplifying assumption was introduced by Spong [73]. It consists, simply, in *neglecting* all dynamic couplings between rotors and links. More formally, this can be stated as

4. The angular velocity of the rotors is only due to the spinning around their own rotation axis.

Mathematically:

$${}^{m_i}\boldsymbol{\omega}_{m_i} = \begin{pmatrix} 0 \\ 0 \\ n_i \dot{\theta}_{m_i} \end{pmatrix}$$

This assumption, although not obvious, was proved to be very reasonable in practice, mostly because of the high reduction ratios, which make the other components negligible. This leads to the kinetic energy being written

$$\mathcal{T} = \frac{1}{2} \dot{\mathbf{q}}^T \mathbf{M}_L(\mathbf{q}) \dot{\mathbf{q}} + \frac{1}{2} \dot{\boldsymbol{\theta}}_m^T \mathbf{B} \dot{\boldsymbol{\theta}}_m + \frac{1}{2} \dot{\mathbf{q}}^T \mathbf{M}_M(\mathbf{q}) \dot{\mathbf{q}} \quad (2.7)$$

where $\mathbf{M}_M(\mathbf{q})$ contains the robot masses and CoM positions. Therefore, the model transforms to

$$\mathbf{M}(\mathbf{q}) \ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) + \boldsymbol{\Sigma}(\mathbf{q} - \boldsymbol{\theta}_m) = \mathbf{0} \quad (2.8a)$$

$$\mathbf{B} \ddot{\boldsymbol{\theta}}_m + \boldsymbol{\Sigma}(\boldsymbol{\theta}_m - \mathbf{q}) = \boldsymbol{\tau} \quad (2.8b)$$

where $\mathbf{M}(\mathbf{q}) = \mathbf{M}_L(\mathbf{q}) + \mathbf{M}_M(\mathbf{q})$. This is called the *reduced model* of the robot. From now on, we will always refer to this model, neglecting (2.6). Equations (2.8a) and (2.8b) are referred to as *link equation* and *motor equation*, respectively.

2.1.1 Inverse dynamics

Given a desired link trajectory $\mathbf{q}_d(t) \in \bar{C}^4$, i.e., four times differentiable (at most with discontinuous fourth derivative), it is possible to use (2.8) for computing the nominal torque that realizes the desired motion. In the first place, we define the joint elastic torque as

$$\boldsymbol{\tau}_e = \boldsymbol{\Sigma}(\boldsymbol{\theta}_m - \mathbf{q}). \quad (2.9)$$

PART I - COMPLIANT MANIPULATORS

Therefore, we can rewrite (2.8a) as

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) = \boldsymbol{\tau}_e. \quad (2.10)$$

Now, we can differentiate (2.10) once¹

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{M}}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}}) = \dot{\boldsymbol{\tau}}_e \quad (2.11)$$

and twice

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + 2\dot{\mathbf{M}}(\mathbf{q})\ddot{\mathbf{q}} + \ddot{\mathbf{M}}(\mathbf{q})\ddot{\mathbf{q}} + \ddot{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}}) = \ddot{\boldsymbol{\tau}}_e. \quad (2.12)$$

From (2.9), we have

$$\dot{\boldsymbol{\tau}}_e = \boldsymbol{\Sigma}(\dot{\boldsymbol{\theta}}_m - \dot{\mathbf{q}}) \quad (2.13)$$

$$\ddot{\boldsymbol{\tau}}_e = \boldsymbol{\Sigma}(\ddot{\boldsymbol{\theta}}_m - \ddot{\mathbf{q}}). \quad (2.14)$$

Thus, the motor velocities and accelerations can be expressed as

$$\dot{\boldsymbol{\theta}}_m = \dot{\mathbf{q}} + \boldsymbol{\Sigma}^{-1}\dot{\boldsymbol{\tau}}_e \quad (2.15)$$

$$\ddot{\boldsymbol{\theta}}_m = \ddot{\mathbf{q}} + \boldsymbol{\Sigma}^{-1}\ddot{\boldsymbol{\tau}}_e \quad (2.16)$$

where $\dot{\boldsymbol{\tau}}_e$ and $\ddot{\boldsymbol{\tau}}_e$ are computed as in the left-hand side of (2.11) and (2.12). Also, from (2.8b)

$$\boldsymbol{\tau} = \mathbf{B}\ddot{\boldsymbol{\theta}}_m + \boldsymbol{\tau}_e. \quad (2.17)$$

Combining equations and plugging everywhere $\mathbf{q} = \mathbf{q}_d(t)$, we get the final expression of the desired torque in terms of the desired link trajectory only:

$$\begin{aligned} \boldsymbol{\tau}_d = & \mathbf{B}\boldsymbol{\Sigma}^{-1}[\mathbf{M}(\mathbf{q}_d)\ddot{\mathbf{q}}_d + 2\dot{\mathbf{M}}(\mathbf{q}_d)\ddot{\mathbf{q}}_d + \ddot{\mathbf{M}}(\mathbf{q}_d)\ddot{\mathbf{q}}_d + \ddot{\mathbf{n}}(\mathbf{q}_d, \dot{\mathbf{q}}_d)] \\ & + [\mathbf{M}(\mathbf{q}_d) + \mathbf{B}]\ddot{\mathbf{q}}_d + \mathbf{n}(\mathbf{q}_d, \dot{\mathbf{q}}_d). \end{aligned} \quad (2.18)$$

Notice that, when $\boldsymbol{\Sigma}$ goes to infinity, an equivalent rigid model can be obtained:

$$\boldsymbol{\tau}_d = [\mathbf{M}(\mathbf{q}_d) + \mathbf{B}]\ddot{\mathbf{q}}_d + \mathbf{n}(\mathbf{q}_d, \dot{\mathbf{q}}_d) \quad (2.19)$$

Comparing this with (1.5), we can see that the two models are equivalent, with the only formal difference that in (2.19) the drive inertia moments are considered explicitly, while in (1.5) they were “hidden” in $\mathbf{M}(\mathbf{q})$.

¹With $\dot{\mathbf{M}}(\mathbf{q})$, we mean $\frac{d}{dt}[\mathbf{M}(\mathbf{q})]$. Obviously, this quantity depends on $\dot{\mathbf{q}}$ as well as on $\mathbf{q}$, but we avoid notation overloading. Similar comments apply to $\dot{\mathbf{n}}$ and to higher-order time derivatives.

2.1.2 Feedback linearization control

From (2.8)–(2.17), it is immediate to see that the nonlinear state feedback law

$$\begin{aligned} \boldsymbol{\tau} = & \mathbf{B}\boldsymbol{\Sigma}^{-1}[\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}}_r + 2\dot{\mathbf{M}}(\mathbf{q})\dot{\mathbf{q}} + \ddot{\mathbf{M}}(\mathbf{q})\mathbf{q} + \ddot{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}})] \\ & + [\mathbf{M}(\mathbf{q}) + \mathbf{B}]\ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) \end{aligned} \quad (2.20)$$

yields $\ddot{\mathbf{q}} = \ddot{\mathbf{q}}_r$. In order to (exponentially) stabilize the trajectory error to zero, we choose

$$\begin{aligned} \ddot{\mathbf{q}}_r = & \ddot{\mathbf{q}}_d + \mathbf{K}_3(\ddot{\mathbf{q}}_d - \ddot{\mathbf{q}}) + \mathbf{K}_2(\dot{\mathbf{q}}_d - \dot{\mathbf{q}}) \\ & + \mathbf{K}_1(\dot{\mathbf{q}}_d - \dot{\mathbf{q}}) + \mathbf{K}_0(\mathbf{q}_d - \mathbf{q}), \end{aligned} \quad (2.21)$$

where $\mathbf{K}_0, \dots, \mathbf{K}_3$ are diagonal matrices, with their diagonal elements $K_{\cdot,i}$ being such that the polynomials

$$p_i(s) = s^4 + K_{3,i}s^3 + K_{2,i}s^2 + K_{1,i}s + K_{0,i}, \quad i = 1, \dots, N, \quad (2.22)$$

are Hurwitz, i.e. all their roots have negative real part, implying that all gains $K_{\cdot,i}$ are necessarily (but not sufficiently) positive. Each of these roots corresponds to an eigenvalue of the controlled system.

The careful reader may have recognized a difficulty in applying the feedback linearization control law in case of elastic joints. The control law makes apparent use of the actual values of $\ddot{\mathbf{q}}$ and $\dot{\mathbf{q}}$, which are hard or even impossible to measure directly. On the other hand, obtaining these values by multiple on-line numerical differentiation of position measurements would introduce excessive noise and delays in a discrete-time implementation. However, a solution to these problems is easily found.

Usually, elasticity-aware designed robots have a rich sensory suite built in. In our case, the essential capability is that of retrieving $(\mathbf{q} \ \dot{\mathbf{q}} \ \boldsymbol{\tau}_e \ \dot{\boldsymbol{\tau}}_e)$ from sensor data. Indeed, from (2.9) and (2.13), it is clear that this is an alternative state representation for elastic-joints robots. The motor position $\boldsymbol{\theta}_m$ is easily measurable, by encoders or other common methods such as Hall sensors. The joint torque $\boldsymbol{\tau}_e$ can also be measured, for instance using strain gauges. Direct measures of the link position $\mathbf{q}$ may also be obtained, for instance using capacitive sensors or potentiometers. All these sensed quantities provides a redundant set of measurements, since they are related through (2.9), making it is easy to compute one of them from the other two. In practice, it is very difficult to achieve a useful resolution and precision when measuring $\mathbf{q}$ [1]; therefore, computing it from $\boldsymbol{\theta}_m$ and $\boldsymbol{\tau}_e$ might be preferable to using its direct measure. Sensor fusion could provide even better estimates for all three quantities [2]. Measures of $\dot{\boldsymbol{\theta}}_m$ may also be available from sensors, such as

PART I - COMPLIANT MANIPULATORS

tachometers. Otherwise its value, as well as that of $\dot{\mathbf{q}}$ and $\dot{\boldsymbol{\tau}}_e$, may be obtained from numerical time differentiation. Thus, full state estimation would require only one level of differentiation, just as in the rigid case (where $\dot{\mathbf{q}}$ is often obtained from $\mathbf{q}$); this is still acceptable for practical applications. Further, $\dot{\mathbf{q}}$, $\dot{\boldsymbol{\theta}}_m$ and $\dot{\boldsymbol{\tau}}_e$ are also related to each other via (2.13), which may be exploited.

Once $\boldsymbol{\tau}_e$ and $\dot{\boldsymbol{\tau}}_e$ are known, the value of $\ddot{\mathbf{q}}$ and $\ddot{\mathbf{j}}$ can be computed from (2.8a) and (2.11) as

$$\ddot{\mathbf{q}} = \mathbf{M}^{-1}(\mathbf{q}) [\boldsymbol{\tau}_e - \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}})] \quad (2.23)$$

$$\ddot{\mathbf{j}} = \mathbf{M}^{-1}(\mathbf{q}) \left[\dot{\boldsymbol{\tau}}_e - \left(\dot{\mathbf{M}}(\mathbf{q}) \ddot{\mathbf{q}} + \dot{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}}) \right) \right] \quad (2.24)$$

where (2.23) is evaluated first to obtain $\ddot{\mathbf{q}}$, so that its value can be used in (2.24) to obtain $\ddot{\mathbf{j}}$. Incidentally, the above equations point out that $(\mathbf{q} \ \dot{\mathbf{q}} \ \ddot{\mathbf{q}} \ \ddot{\mathbf{j}})$ is an equivalent state representation for the reduced model (2.8), which is not the case for the full model (2.6).

It can be noticed that (2.23) is formally identical to (1.6), the expression of the direct dynamics for rigid robots. This is in contrast with the rigid case, where no inversion of the inertia matrix is needed for feedback linearization control.

2.2 The Algorithm EJNEA

An algorithm capable of computing numerically $\boldsymbol{\tau}_d$ from a desired $\mathbf{q}_d$ and its first four derivatives, and which implements (2.18) without the need of deriving first the symbolic model, can be designed as a direct extension of the standard RNEA. In the following, we will drop for compactness the subscript d (of desired trajectory). As shown in Sec. 2.1.1, it is necessary to compute first $\boldsymbol{\tau}_e$, then $\dot{\boldsymbol{\tau}}_e$, and finally exploit (2.16) and (2.17) to obtain the nominal command torque $\boldsymbol{\tau}$.

Indeed, $\boldsymbol{\tau}_e$ can be computed using the standard RNEA. The only difference in this case is that the input parameters do not include the data about the drive inertia moments. The main difficulty lies in computing $\dot{\boldsymbol{\tau}}_e$ and for this, higher-order dynamic terms need to be considered. In particular, the recursive algorithm will necessarily involve two more levels of differentiation than the standard RNEA, with the need of setting up recursions for higher-order time derivatives of motion variables. In particular, we are computing the *jerk* (also known as *jolt*) and the *snap* (or *jounce*), which are, respectively, the third and fourth time derivative of position. As usual, all quantities will be conveniently expressed in the *moving* reference frame of the considered

link. This is very advantageous, since all dynamic parameters of the robot will be constant in these frames. The algorithm is obtained by symbolically differentiating the Newton-Euler equations; full derivation in Appendix A.

2.2.1 Forward recursion

The following equations need to be propagated from the robot base to the tip, for $i = 1, \dots, N$:

$${}^i\omega_i = {}^iR_{i-1}({}^{i-1}\omega_{i-1} + \dot{\theta}_i \hat{z}_0) \quad (2.25)$$

$${}^i\gamma_i = {}^iR_{i-1}({}^{i-1}\gamma_{i-1} + \ddot{\theta}_i \hat{z}_0 + {}^{i-1}\omega_{i-1} \times \dot{\theta}_i \hat{z}_0) \quad (2.26)$$

$$\begin{aligned} {}^i\iota_i = {}^iR_{i-1}[{}^{i-1}\iota_{i-1} + \ddot{\theta}_i \hat{z}_0 + {}^{i-1}\gamma_{i-1} \times \dot{\theta}_i \hat{z}_0 \\ + {}^{i-1}\omega_{i-1} \times (2\ddot{\theta}_i \hat{z}_0 + {}^{i-1}\omega_{i-1} \times \dot{\theta}_i \hat{z}_0)] \end{aligned} \quad (2.27)$$

$$\begin{aligned} {}^i\varsigma_i = {}^iR_{i-1}\{ {}^{i-1}\varsigma_{i-1} + \ddot{\theta}_i \hat{z}_0 + 3 {}^{i-1}\gamma_{i-1} \times \ddot{\theta}_i \hat{z}_0 \\ + 3 {}^{i-1}\omega_{i-1} \times (\ddot{\theta}_i \hat{z}_0 + {}^{i-1}\omega_{i-1} \times \dot{\theta}_i \hat{z}_0) \\ + 2 {}^{i-1}\gamma_{i-1} \times ({}^{i-1}\omega_{i-1} \times \dot{\theta}_i \hat{z}_0) \\ + {}^{i-1}\omega_{i-1} \times [{}^{i-1}\omega_{i-1} \times ({}^{i-1}\omega_{i-1} \times \dot{\theta}_i \hat{z}_0) \\ + {}^{i-1}\gamma_{i-1} \times \dot{\theta}_i \hat{z}_0] + {}^{i-1}\iota_{i-1} \times \dot{\theta}_i \hat{z}_0 \} \end{aligned} \quad (2.28)$$

$$\begin{aligned} {}^i\mathbf{a}_i = {}^iR_{i-1}{}^{i-1}\mathbf{a}_{i-1} + {}^i\omega_i \times ({}^i\omega_i \times {}^i\mathbf{p}_{i,i-1}) \\ + {}^i\gamma_i \times {}^i\mathbf{p}_{i,i-1} + \ddot{d}_i {}^i\hat{z}_{i-1} + 2\dot{d}_i ({}^i\omega_i \times {}^i\hat{z}_{i-1}) \end{aligned} \quad (2.29)$$

$${}^i\mathbf{a}_{c_i} = {}^i\mathbf{a}_i + {}^i\omega_i \times ({}^i\omega_i \times {}^i\mathbf{p}_{c_i,i}) + {}^i\gamma_i \times {}^i\mathbf{p}_{c_i,i} \quad (2.30)$$

$$\begin{aligned} {}^i\mathbf{j}_i = {}^iR_{i-1}{}^{i-1}\mathbf{j}_{i-1} + 2 {}^i\gamma_i \times ({}^i\omega_i \times {}^i\mathbf{p}_{i,i-1}) \\ + {}^i\omega_i \times [{}^i\gamma_i \times {}^i\mathbf{p}_{i,i-1} + {}^i\omega_i \times ({}^i\omega_i \times {}^i\mathbf{p}_{i,i-1})] \\ + {}^i\iota_i \times {}^i\mathbf{p}_{i,i-1} + \ddot{d}_i {}^i\hat{z}_{i-1} + 3\dot{d}_i {}^i\omega_i \times {}^i\hat{z}_{i-1} \\ + 3\dot{d}_i [{}^i\gamma_i \times {}^i\hat{z}_{i-1} + {}^i\omega_i \times ({}^i\omega_i \times {}^i\hat{z}_{i-1})] \end{aligned} \quad (2.31)$$

$$\begin{aligned} {}^i\mathbf{j}_{c_i} = {}^i\mathbf{j}_i + {}^i\iota_i \times {}^i\mathbf{p}_{c_i,i} + 2 {}^i\gamma_i \times ({}^i\omega_i \times {}^i\mathbf{p}_{c_i,i}) \\ + {}^i\omega_i \times [{}^i\gamma_i \times {}^i\mathbf{p}_{c_i,i} + {}^i\omega_i \times ({}^i\omega_i \times {}^i\mathbf{p}_{c_i,i})] \end{aligned} \quad (2.32)$$

$$\begin{aligned} {}^i\mathbf{s}_i = {}^iR_{i-1}{}^{i-1}\mathbf{s}_{i-1} + {}^i\varsigma_i \times {}^i\mathbf{p}_{i,i-1} + 3 {}^i\iota_i \times ({}^i\omega_i \times {}^i\mathbf{p}_{i,i-1}) \\ + 3 {}^i\gamma_i \times [{}^i\gamma_i \times {}^i\mathbf{p}_{i,i-1} + {}^i\omega_i \times ({}^i\omega_i \times {}^i\mathbf{p}_{i,i-1})] \\ + {}^i\omega_i \times [{}^i\iota_i \times {}^i\mathbf{p}_{i,i-1} + 2 {}^i\gamma_i \times ({}^i\omega_i \times {}^i\mathbf{p}_{i,i-1})] \\ + {}^i\omega_i \times \{ {}^i\omega_i \times [{}^i\gamma_i \times {}^i\mathbf{p}_{i,i-1} + {}^i\omega_i \times ({}^i\omega_i \times {}^i\mathbf{p}_{i,i-1})] \} \\ + \ddot{d}_i {}^i\hat{z}_{i-1} + 4\dot{d}_i {}^i\iota_i \times {}^i\hat{z}_{i-1} + 8\dot{d}_i {}^i\gamma_i \times ({}^i\omega_i \times {}^i\hat{z}_{i-1}) \\ + 4\dot{d}_i {}^i\omega_i \times [{}^i\gamma_i \times {}^i\hat{z}_{i-1} + {}^i\omega_i \times ({}^i\omega_i \times {}^i\hat{z}_{i-1})] \\ + 6\dot{d}_i [{}^i\gamma_i \times {}^i\hat{z}_{i-1} + {}^i\omega_i \times ({}^i\omega_i \times {}^i\hat{z}_{i-1})] + 4\ddot{d}_i {}^i\omega_i \times {}^i\hat{z}_{i-1} \end{aligned} \quad (2.33)$$

PART I - COMPLIANT MANIPULATORS

$$\begin{aligned}
{}^i\mathbf{s}_{c_i} = & {}^i\mathbf{s}_i + {}^i\boldsymbol{\varsigma}_i \times {}^i\mathbf{p}_{c_i,i} + 3 {}^i\boldsymbol{\nu}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{c_i,i}) \\
& + 3 {}^i\boldsymbol{\gamma}_i \times [{}^i\boldsymbol{\gamma}_i \times {}^i\mathbf{p}_{c_i,i} + {}^i\boldsymbol{\omega}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{c_i,i})] \\
& + {}^i\boldsymbol{\omega}_i \times [{}^i\boldsymbol{\nu}_i \times {}^i\mathbf{p}_{c_i,i} + 2 {}^i\boldsymbol{\gamma}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{c_i,i})] \\
& + {}^i\boldsymbol{\omega}_i \times \{ {}^i\boldsymbol{\omega}_i \times [{}^i\boldsymbol{\gamma}_i \times {}^i\mathbf{p}_{c_i,i} + {}^i\boldsymbol{\omega}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{c_i,i})] \}.
\end{aligned} \tag{2.34}$$

The following symbols have been used:

- $\mathbf{p}_{c_i,i}$ position of the CoM of the augmented link i with respect to the origin of frame i
- $\boldsymbol{\omega}_i$ angular velocity of frame i
- $\boldsymbol{\gamma}_i$ angular acceleration of frame i
- $\boldsymbol{\nu}_i$ angular jerk of frame i
- $\boldsymbol{\varsigma}_i$ angular snap of frame i
- $\mathbf{a}_i$ acceleration of the origin of frame i
- $\mathbf{a}_{c_i}$ acceleration of the center of mass (CoM) of link i
- $\mathbf{j}_i$ jerk of the origin of frame i
- $\mathbf{j}_{c_i}$ jerk of the CoM of link i
- $\mathbf{s}_i$ snap of the origin of frame i
- $\mathbf{s}_{c_i}$ snap of the CoM of link i .

As in standard Newton-Euler, the initialization of the forward recursion is zero for all quantities, with the exception of ${}^0\mathbf{a}_0$, which is set to $-\vec{\mathbf{g}}$ to account for gravitational effects. The above equations are valid both for revolute and prismatic joints, with $\theta_i = q_i$ in the former case, and $\dot{d}_i = q_i$ in the latter. Moreover, if joint i is revolute, then $\dot{d}_i = \ddot{d}_i = \dddot{d}_i = \ddot{\ddot{d}}_i = 0$, while if it is prismatic $\dot{\theta}_i = \ddot{\theta}_i = \ddot{\ddot{\theta}}_i = \ddot{\ddot{\ddot{\theta}}}_i = 0$. In an actual implementation of the algorithm, the two different kinds of joints can be treated separately, so to achieve considerable simplifications.

2.2.2 Backward recursion

The following equations need to be propagated from the robot tip to the base, for $i = N, \dots, 1$:

$${}^i\mathbf{F}_i = m_i {}^i\mathbf{a}_{c_i} \quad (2.35)$$

$${}^i\dot{\mathbf{F}}_i = m_i {}^i\dot{\mathbf{j}}_{c_i} \quad (2.36)$$

$${}^i\ddot{\mathbf{F}}_i = m_i {}^i\mathbf{s}_{c_i} \quad (2.37)$$

$${}^i\mathbf{N}_i = {}^i\mathbf{I}_i {}^i\boldsymbol{\gamma}_i + {}^i\boldsymbol{\omega}_i \times ({}^i\mathbf{I}_i {}^i\boldsymbol{\omega}_i) \quad (2.38)$$

$${}^i\dot{\mathbf{N}}_i = {}^i\boldsymbol{\omega}_i \times ({}^i\mathbf{I}_i {}^i\boldsymbol{\gamma}_i + {}^i\mathbf{N}_i) + {}^i\mathbf{I}_i ({}^i\boldsymbol{\gamma}_i \times {}^i\boldsymbol{\omega}_i + {}^i\boldsymbol{\omega}_i \times {}^i\boldsymbol{\gamma}_i) + {}^i\boldsymbol{\gamma}_i \times {}^i\mathbf{I}_i {}^i\boldsymbol{\omega}_i \quad (2.39)$$

$${}^i\ddot{\mathbf{N}}_i = {}^i\boldsymbol{\gamma}_i \times ({}^i\mathbf{I}_i {}^i\boldsymbol{\gamma}_i + 2{}^i\mathbf{N}_i) + {}^i\boldsymbol{\omega}_i \times {}^i\mathbf{I}_i {}^i\boldsymbol{\omega}_i \quad (2.40)$$

$$\begin{aligned} & + {}^i\mathbf{I}_i [2({}^i\boldsymbol{\omega}_i \times {}^i\boldsymbol{\gamma}_i) + {}^i\boldsymbol{\omega}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\boldsymbol{\gamma}_i) + {}^i\boldsymbol{\gamma}_i \times {}^i\boldsymbol{\omega}_i] \\ & + {}^i\boldsymbol{\omega}_i \times [{}^i\boldsymbol{\omega}_i \times {}^i\mathbf{I}_i {}^i\boldsymbol{\gamma}_i + 2{}^i\mathbf{I}_i ({}^i\boldsymbol{\omega}_i \times {}^i\boldsymbol{\gamma}_i) + {}^i\dot{\mathbf{N}}_i] \end{aligned}$$

$${}^i\mathbf{f}_i = {}^i\mathbf{R}_{i+1} {}^{i+1}\mathbf{f}_{i+1} + {}^i\mathbf{F}_i \quad (2.41)$$

$${}^i\dot{\mathbf{f}}_i = {}^i\mathbf{R}_{i+1} {}^{i+1}\dot{\mathbf{f}}_{i+1} + {}^i\dot{\mathbf{F}}_i \quad (2.42)$$

$${}^i\ddot{\mathbf{f}}_i = {}^i\mathbf{R}_{i+1} {}^{i+1}\ddot{\mathbf{f}}_{i+1} + {}^i\ddot{\mathbf{F}}_i \quad (2.43)$$

$${}^i\mathbf{n}_i = {}^i\mathbf{R}_{i+1} {}^{i+1}\mathbf{n}_{i+1} + {}^i\mathbf{p}_{c_i,i} \times {}^i\mathbf{F}_i + {}^i\mathbf{p}_{i,i-1} \times {}^i\mathbf{f}_i + {}^i\mathbf{N}_i \quad (2.44)$$

$${}^i\dot{\mathbf{n}}_i = {}^i\mathbf{R}_{i+1} {}^{i+1}\dot{\mathbf{n}}_{i+1} + ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{c_i,i}) \times {}^i\mathbf{F}_i + {}^i\mathbf{p}_{c_i,i} \times {}^i\dot{\mathbf{F}}_i \quad (2.45)$$

$$+ ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{i,i-1} + \dot{d}_i {}^i\hat{\mathbf{z}}_{i-1}) \times {}^i\mathbf{f}_i + {}^i\mathbf{p}_{i,i-1} \times {}^i\dot{\mathbf{f}}_i + {}^i\dot{\mathbf{N}}_i$$

$${}^i\ddot{\mathbf{n}}_i = {}^i\mathbf{R}_{i+1} {}^{i+1}\ddot{\mathbf{n}}_{i+1} + 2({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{c_i,i}) \times {}^i\dot{\mathbf{F}}_i \quad (2.46)$$

$$+ [{}^i\boldsymbol{\gamma}_i \times {}^i\mathbf{p}_{c_i,i} + {}^i\boldsymbol{\omega}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{c_i,i})] \times {}^i\mathbf{F}_i$$

$$+ [{}^i\boldsymbol{\gamma}_i \times {}^i\mathbf{p}_{i,i-1} + {}^i\boldsymbol{\omega}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{i,i-1}) + \ddot{d}_i {}^i\hat{\mathbf{z}}_{i-1}$$

$$+ 2\dot{d}_i {}^i\boldsymbol{\omega}_i \times {}^i\hat{\mathbf{z}}_{i-1}] \times {}^i\mathbf{f}_i + {}^i\mathbf{p}_{c_i,i} \times {}^i\ddot{\mathbf{F}}_i + {}^i\ddot{\mathbf{N}}_i$$

$$+ 2({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{i,i-1} + \dot{d}_i {}^i\hat{\mathbf{z}}_{i-1}) \times {}^i\dot{\mathbf{f}}_i + {}^i\mathbf{p}_{i,i-1} \times {}^i\ddot{\mathbf{f}}_i$$

$$\tau_{ei} = \begin{cases} {}^i\mathbf{n}_i^T {}^i\hat{\mathbf{z}}_{i-1} & \text{if joint } i \text{ is revolute} \\ {}^i\mathbf{f}_i^T {}^i\hat{\mathbf{z}}_{i-1} & \text{if joint } i \text{ is prismatic} \end{cases} \quad (2.47)$$

$$\dot{\tau}_{ei} = \begin{cases} ({}^i\dot{\mathbf{n}}_i + {}^i\mathbf{n}_i \times {}^i\boldsymbol{\omega}_i)^T {}^i\hat{\mathbf{z}}_{i-1} & \text{revolute} \\ ({}^i\dot{\mathbf{f}}_i + {}^i\mathbf{f}_i \times {}^i\boldsymbol{\omega}_i)^T {}^i\hat{\mathbf{z}}_{i-1} & \text{prismatic} \end{cases} \quad (2.48)$$

$$\ddot{\tau}_{ei} = \begin{cases} [{}^i\ddot{\mathbf{n}}_i + 2({}^i\dot{\mathbf{n}}_i \times {}^i\boldsymbol{\omega}_i) + {}^i\mathbf{n}_i \times {}^i\boldsymbol{\gamma}_i \\ \quad + ({}^i\mathbf{n}_i \times {}^i\boldsymbol{\omega}_i) \times {}^i\boldsymbol{\omega}_i]^T {}^i\hat{\mathbf{z}}_{i-1} & \text{revolute} \\ [{}^i\ddot{\mathbf{f}}_i + 2({}^i\dot{\mathbf{f}}_i \times {}^i\boldsymbol{\omega}_i) + {}^i\mathbf{f}_i \times {}^i\boldsymbol{\gamma}_i \\ \quad + ({}^i\mathbf{f}_i \times {}^i\boldsymbol{\omega}_i) \times {}^i\boldsymbol{\omega}_i]^T {}^i\hat{\mathbf{z}}_{i-1} & \text{prismatic} \end{cases} \quad (2.49)$$

PART I - COMPLIANT MANIPULATORS

$$\dot{\theta}_{mi} = \frac{\dot{\tau}_{ei}}{\sigma_i} + \dot{q}_i \quad (2.50)$$

$$\ddot{\theta}_{mi} = \frac{\ddot{\tau}_{ei}}{\sigma_i} + \ddot{q}_i \quad (2.51)$$

$$\tau_i = B_i \ddot{\theta}_{mi} + \tau_{ei} \quad (2.52)$$

where we employ:

- σ_i stiffness of joint i
- B_i drive inertia moment of joint i
- m_i mass of augmented link i
- ${}^i\mathbf{I}_i$ central inertia tensor of link i , in frame i
- $\mathbf{F}_i$ total force acting on the CoM of link i
- $\mathbf{N}_i$ total torque acting on link i
- $\mathbf{f}_i$ total force exerted on link i by link $i - 1$
- $\mathbf{n}_i$ total torque exerted on link i by link $i - 1$.

It is important to remark that, while m_i and $\mathbf{p}_{c_i,i}$ represent the mass and CoM position of the *augmented* link, here ${}^i\mathbf{I}_i$ *only* accounts for the inertia properties of link i , without considering the motor, since the rotor's inertial components, other than the drive inertia moments, are neglected by Spong's assumption. Again, the algorithm is initialized with the external forces $\mathbf{f}_{N+1}$ and $\mathbf{n}_{N+1}$. While (2.48) and (2.50) are not strictly needed for the computation of the torques, they were nevertheless included into the equations, as they will turn out to be useful for various extensions and applications that will be presented in the following sections.

Summarizing, given a desired $\mathbf{q}_d$ and its first four derivatives (at any given instant of time), the use of first (2.25)–(2.34) and then (2.35)–(2.52) will provide the nominal inverse dynamics torque

$$\boldsymbol{\tau}_d = \text{EJNEA}(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d, \dddot{\mathbf{q}}_d, \mathbf{f}_d), \quad (2.53)$$

being the algorithm seen as a subroutine with five suitable input arguments.

2.3 Application to Control

Through the use of the EJNEA subroutine, it is possible to implement also a feedback linearization control law for robots with elastic joints. In fact, the required control torque (2.20) can be obtained by simply calling this subroutine with

$$\boldsymbol{\tau} = \text{EJNEA}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \ddot{\mathbf{q}}_r, \ddot{\mathbf{q}}_r), \quad (2.54)$$

where $\ddot{\mathbf{q}}_r$ is computed from (2.21), $\ddot{\mathbf{q}}$ from (2.23), and $\ddot{\mathbf{q}}$ from (2.24). In the rest of this section, we will focus on how to implement efficiently (2.23) and (2.24) in an algorithmic way.

As mentioned in Sec. 2.1.2, the evaluation of (2.23) corresponds to the task of solving the direct dynamics problem for a rigid robot. Several methods addressing this task are available. However, in robots with elastic joints a solution to the direct dynamics is required *also* for control purposes, not only for simulations. Therefore, real time performance will be the core aspect to consider.

The first issue is how to compute the generalized inertia matrix $\mathbf{M}(\mathbf{q})$ at the current value $\mathbf{q}$. In principle, this could be done by using the RNEA. If $\mathbf{e}_i$ is the i -th column of the $N \times N$ identity matrix, then the standard Newton-Euler algorithm can be called with $\ddot{\mathbf{g}} = \mathbf{0}$ (removing gravity) in the form

$$\mathbf{m}_i(\mathbf{q}) = \text{RNEA}_0(\mathbf{q}, \mathbf{0}, \mathbf{e}_i) \quad (2.55)$$

where $\mathbf{m}_i(\mathbf{q})$ is the i -th column of the robot inertia matrix $\mathbf{M}(\mathbf{q})$ and the subscript 0 denotes the absence of gravity. Repeating this computation for $i = 1, \dots, N$ would return the full inertia matrix in $O(N^2)$ complexity. However, this is definitely not the best way to proceed. A much faster method is the Composite Rigid Body Algorithm (CRBA), which was specifically designed for this purpose and first appeared in [86] as Method 3. Thus, we evaluate the inertia matrix as

$$\mathbf{M}(\mathbf{q}) = \text{CRBA}(\mathbf{q}). \quad (2.56)$$

The second issue involves the inversion of the obtained inertia matrix in order to solve for $\ddot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$. In general, inverting a (numeric) matrix is not the most efficient solution. Letting

$$\mathbf{A} = \mathbf{M}(\mathbf{q}) \quad (2.57)$$

$$\mathbf{b}_1 = \boldsymbol{\tau}_e - \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) \quad (2.58)$$

$$\mathbf{b}_2 = \dot{\boldsymbol{\tau}}_e - [\dot{\mathbf{M}}(\mathbf{q})\dot{\mathbf{q}} + \dot{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}})] \quad (2.59)$$

PART I - COMPLIANT MANIPULATORS

a better approach would be to solve the two linear systems

$$\mathbf{A}\ddot{\mathbf{q}} = \mathbf{b}_1 \quad (2.60)$$

$$\mathbf{A}\ddot{\ddot{\mathbf{q}}} = \mathbf{b}_2. \quad (2.61)$$

This approach is in fact rather standard in direct dynamics of rigid robots. The solution can be computed using any algorithm designed for solving linear systems. In particular, since $\mathbf{A} = \mathbf{M}(\mathbf{q})$ is always positive definite and symmetric, specialized methods could be used, such as using the $\mathbf{LL}^T$ (Cholesky) or $\mathbf{LDL}^T$ decompositions. Obviously, in our case, we need first to solve (2.60), then compute $\mathbf{b}_2$, and finally solve (2.61). Notice that $\mathbf{A}$, at a given configuration $\mathbf{q}$, needs to be computed and decomposed only once.

The last issue left is how to compute $\mathbf{b}_1$ and $\mathbf{b}_2$ algorithmically, using the obtained values of $\ddot{\mathbf{q}}$ and $\ddot{\ddot{\mathbf{q}}}$. This can be done by using EJNEA or, even better, smaller versions of it. By $\text{EJNEA}_2^*(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$ we denote a reduced version of EJNEA, returning $\boldsymbol{\tau}_e$. Note that this corresponds to the standard RNEA. Similarly, by $\text{EJNEA}_3^*(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \ddot{\ddot{\mathbf{q}}})$ we denote a reduced version returning $\dot{\boldsymbol{\tau}}_e$. Then:

$$\mathbf{b}_1 = \boldsymbol{\tau}_e - \text{EJNEA}_2^*(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{0}) \quad (2.62)$$

$$\mathbf{b}_2 = \dot{\boldsymbol{\tau}}_e - \text{EJNEA}_3^*(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbf{0}). \quad (2.63)$$

The computation of $\mathbf{b}_1$ through the use of RNEA is a standard step also in the direct dynamics of rigid robots. An even further optimization would consist in keeping a reference to the angular velocities ${}^i\boldsymbol{\omega}_i$ computed in EJNEA_2^* for later re-use in EJNEA_3^* and EJNEA, and also to the accelerations ${}^i\boldsymbol{\gamma}_i$, ${}^i\mathbf{a}_i$ and ${}^i\mathbf{a}_{c_i}$ computed in EJNEA_3^* for re-use in EJNEA. This avoids recomputing the values from scratch, since the velocities do not depend on $\ddot{\mathbf{q}}$, and the accelerations do not depend on $\ddot{\ddot{\mathbf{q}}}$. Notice that it is possible to implement simplified versions of EJNEA_2^* and EJNEA_3^* including the built-in assumptions $\ddot{\mathbf{q}} = \mathbf{0}$ and $\ddot{\ddot{\mathbf{q}}} = \mathbf{0}$, but the performance gain would be negligible.

Summarizing, the necessary steps are the following, where the computational complexity of each step is highlighted:

- one run of CRBA to get $\mathbf{A} = \mathbf{M}(\mathbf{q}) - O(N^2)$
- one decomposition $\mathbf{A} = \mathbf{LDL}^T - O(N^3)$
- one run of EJNEA_2^* to get $\mathbf{b}_1 - O(N)$
- solve the linear system (2.60), using the $\mathbf{LDL}^T - O(N^2)$
- one run of EJNEA_3^* to get $\mathbf{b}_2 - O(N)$

- solve the linear system (2.61), using the $\mathbf{LDL}^T - O(N^2)$
- one run of EJNEA to obtain the input torque $\boldsymbol{\tau} - O(N)$

Therefore, the total computational cost grows asymptotically as $O(N^3)$.

Another possibility is to resort to a completely different way to compute the direct dynamics, namely Featherstone's Articulated Body Algorithm (ABA) [29, Ch. 6] [30, Ch. 7]. This algorithm, contrarily to the traditional approach followed in this work, is usually formulated in terms of screw theory, on which more words will be spent in Sec. 2.5. For rigid robots, it allows to obtain $\ddot{\mathbf{q}}$ directly and in $O(N)$ time, without computing and inverting $\mathbf{M}(\mathbf{q})$, and can be used for elastic joint robots as

$$\ddot{\mathbf{q}} = \text{ABA}(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\tau}_e). \quad (2.64)$$

The first version of ABA, appeared in [28], was assuming $\dot{\mathbf{q}} = \mathbf{0}$ and no gravity action. In this way, the full direct dynamics could still be computed as

$$\ddot{\mathbf{q}} = \mathbf{M}^{-1}(\mathbf{q})\mathbf{b}_1 = \text{ABA}_0(\mathbf{q}, \mathbf{0}, \mathbf{b}_1) \quad (2.65)$$

where $\mathbf{b}_1$ is defined in (2.58) and the subscript 0 denotes the absence of gravity. In principle, this first version of the algorithm could also work for computing $\ddot{\mathbf{q}}$, by simply using $\mathbf{b}_2$ instead of $\mathbf{b}_1$ as

$$\ddot{\mathbf{q}} = \mathbf{M}^{-1}(\mathbf{q})\mathbf{b}_2 = \text{ABA}_0(\mathbf{q}, \mathbf{0}, \mathbf{b}_2) \quad (2.66)$$

while yet another possibility would be extending ABA to the third order using a differentiation procedure similar to the one employed for RNEA in this work. In either case, using ABA would make the feedback linearization computational cost fully linear with respect to the number of joints.

The choice of whether using ABA or CRBA-based methods for controlling robots with elastic joints depends on the number of joints. Although the computational cost of using ABA is linear with respect to the number of joints, in practice the real cost is much larger than that of employing CRBA for simple structures, since the multiplicative constants are much higher in the first case. For rigid robots, ABA becomes convenient for computing the direct dynamics when $N \geq 9$. For elastic-joint robots, the threshold is higher, since $\mathbf{M}(\mathbf{q})$ needs to be computed and decomposed only once with the proposed algorithm, while ABA would need to run twice.

Finally, it should be noticed that the applications of EJNEA in control settings are not limited to feedback linearization or to feedforward torque components. Indeed, a skillful selection of the input arguments can lead to the implementation of different control schemes. For instance, this is what was done in [33] to implement a passivity-based controller.

2.4 Modified Denavit-Hartenberg notation

In Sec. 2.2, the standard Denavit-Hartenberg notation, described in Sec. 1.2.1, was employed to present EJNEA.

As also mentioned in Sec. 1.2.1, another widespread notation for the robot kinematics is the so-called modified Denavit-Hartenberg convention, introduced by Craig [17]. Different specialists may use one convention or the other, mostly as a matter of personal use. This is sometimes a source of confusion, especially when professionals have to reference works employing a convention they are not used to. However, the two conventions are equivalent to each other in terms of representational power. Clearly, as a consequence of employing the modified convention, some description parameters of the robot will be different from the standard ones. This is true not only for the kinematic Denavit-Hartenberg parameters (which may include the joint coordinates $\mathbf{q}$), but also for the dynamic parameters ${}^i\mathbf{p}_{c_i,i}$ and ${}^i\mathbf{I}_i$, since the reference frames will be different.

Here, the equivalent EJNEA algorithm for the modified convention is presented for the first time. Hints to its derivation are given in Appendix A, the rest being dropped, as it is analogous to that for the standard convention.

$${}^i\boldsymbol{\omega}_i = {}^i\mathbf{R}_{i-1} ({}^{i-1}\boldsymbol{\omega}_{i-1} + \dot{\theta}_i {}^{i-1}\hat{\mathbf{z}}_i) \quad (2.67)$$

$${}^i\boldsymbol{\gamma}_i = {}^i\mathbf{R}_{i-1} ({}^{i-1}\boldsymbol{\gamma}_{i-1} + \ddot{\theta}_i {}^{i-1}\hat{\mathbf{z}}_i + {}^{i-1}\boldsymbol{\omega}_{i-1} \times \dot{\theta}_i {}^{i-1}\hat{\mathbf{z}}_i) \quad (2.68)$$

$$\begin{aligned} {}^i\boldsymbol{\iota}_i = {}^i\mathbf{R}_{i-1} [& {}^{i-1}\boldsymbol{\iota}_{i-1} + \ddot{\theta}_i {}^{i-1}\hat{\mathbf{z}}_i + {}^{i-1}\boldsymbol{\gamma}_{i-1} \times \dot{\theta}_i {}^{i-1}\hat{\mathbf{z}}_i \\ & + {}^{i-1}\boldsymbol{\omega}_{i-1} \times (2\ddot{\theta}_i {}^{i-1}\hat{\mathbf{z}}_i + {}^{i-1}\boldsymbol{\omega}_{i-1} \times \dot{\theta}_i {}^{i-1}\hat{\mathbf{z}}_i)] \end{aligned} \quad (2.69)$$

$$\begin{aligned} {}^i\boldsymbol{\varsigma}_i = {}^i\mathbf{R}_{i-1} \{ & {}^{i-1}\boldsymbol{\varsigma}_{i-1} + \ddot{\theta}_i {}^{i-1}\hat{\mathbf{z}}_i + 3 {}^{i-1}\boldsymbol{\gamma}_{i-1} \times \ddot{\theta}_i {}^{i-1}\hat{\mathbf{z}}_i \\ & + 3 {}^{i-1}\boldsymbol{\omega}_{i-1} \times (\ddot{\theta}_i {}^{i-1}\hat{\mathbf{z}}_i + {}^{i-1}\boldsymbol{\omega}_{i-1} \times \ddot{\theta}_i {}^{i-1}\hat{\mathbf{z}}_i) \\ & + 2 {}^{i-1}\boldsymbol{\gamma}_{i-1} \times (\dot{\theta}_i {}^{i-1}\hat{\mathbf{z}}_i + {}^{i-1}\boldsymbol{\omega}_{i-1} \times \dot{\theta}_i {}^{i-1}\hat{\mathbf{z}}_i) \\ & + {}^{i-1}\boldsymbol{\omega}_{i-1} \times [{}^{i-1}\boldsymbol{\omega}_{i-1} \times ({}^{i-1}\boldsymbol{\omega}_{i-1} \times \dot{\theta}_i {}^{i-1}\hat{\mathbf{z}}_i) \\ & + {}^{i-1}\boldsymbol{\gamma}_{i-1} \times \dot{\theta}_i {}^{i-1}\hat{\mathbf{z}}_i] + {}^{i-1}\boldsymbol{\iota}_{i-1} \times \dot{\theta}_i {}^{i-1}\hat{\mathbf{z}}_i \} \end{aligned} \quad (2.70)$$

$$\begin{aligned} {}^i\mathbf{a}_i = {}^i\mathbf{R}_{i-1} [& {}^{i-1}\mathbf{a}_{i-1} + {}^{i-1}\boldsymbol{\omega}_{i-1} \times ({}^{i-1}\boldsymbol{\omega}_{i-1} \times {}^{i-1}\mathbf{p}_{i,i-1}) \\ & + {}^{i-1}\boldsymbol{\gamma}_{i-1} \times {}^{i-1}\mathbf{p}_{i,i-1} + \ddot{d}_i {}^{i-1}\hat{\mathbf{z}}_i + 2\dot{d}_i ({}^{i-1}\boldsymbol{\omega}_{i-1} \times {}^{i-1}\hat{\mathbf{z}}_i)] \end{aligned} \quad (2.71)$$

$${}^i\mathbf{a}_{c_i} = {}^i\mathbf{a}_i + {}^i\boldsymbol{\omega}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{c_i,i}) + {}^i\boldsymbol{\gamma}_i \times {}^i\mathbf{p}_{c_i,i} \quad (2.72)$$

$$\begin{aligned} {}^i\mathbf{j}_i = {}^i\mathbf{R}_{i-1} \{ & {}^{i-1}\mathbf{j}_{i-1} + 2 {}^{i-1}\boldsymbol{\gamma}_{i-1} \times ({}^{i-1}\boldsymbol{\omega}_{i-1} \times {}^{i-1}\mathbf{p}_{i,i-1}) \\ & + {}^{i-1}\boldsymbol{\omega}_{i-1} \times [{}^{i-1}\boldsymbol{\gamma}_{i-1} \times {}^{i-1}\mathbf{p}_{i,i-1} + {}^{i-1}\boldsymbol{\omega}_{i-1} \times ({}^{i-1}\boldsymbol{\omega}_{i-1} \times {}^{i-1}\mathbf{p}_{i,i-1})] \\ & + {}^{i-1}\boldsymbol{\iota}_{i-1} \times {}^{i-1}\mathbf{p}_{i,i-1} + \ddot{d}_i {}^{i-1}\hat{\mathbf{z}}_i + 3\dot{d}_i {}^{i-1}\boldsymbol{\omega}_{i-1} \times {}^{i-1}\hat{\mathbf{z}}_i \\ & + 3\dot{d}_i [{}^{i-1}\boldsymbol{\gamma}_{i-1} \times {}^{i-1}\hat{\mathbf{z}}_i + {}^{i-1}\boldsymbol{\omega}_{i-1} \times ({}^{i-1}\boldsymbol{\omega}_{i-1} \times {}^{i-1}\hat{\mathbf{z}}_i)] \} \end{aligned} \quad (2.73)$$

$${}^i\mathbf{j}_{c_i} = {}^i\mathbf{j}_i + {}^i\boldsymbol{\nu}_i \times {}^i\mathbf{p}_{c_i,i} + 2 {}^i\boldsymbol{\gamma}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{c_i,i}) \quad (2.74)$$

$$\begin{aligned} {}^i\mathbf{s}_i = & {}^i\mathbf{R}_{i-1} \left\{ {}^{i-1}\mathbf{s}_{i-1} + {}^{i-1}\boldsymbol{\varsigma}_{i-1} \times {}^{i-1}\mathbf{p}_{i,i-1} + 3 {}^{i-1}\boldsymbol{\nu}_{i-1} \times ({}^{i-1}\boldsymbol{\omega}_{i-1} \times {}^{i-1}\mathbf{p}_{i,i-1}) \right. \\ & + 3 {}^{i-1}\boldsymbol{\gamma}_{i-1} \times [{}^{i-1}\boldsymbol{\gamma}_{i-1} \times {}^{i-1}\mathbf{p}_{i,i-1} + {}^{i-1}\boldsymbol{\omega}_{i-1} \times ({}^{i-1}\boldsymbol{\omega}_{i-1} \times {}^{i-1}\mathbf{p}_{i,i-1})] \\ & + {}^{i-1}\boldsymbol{\omega}_{i-1} \times [{}^{i-1}\boldsymbol{\nu}_{i-1} \times {}^{i-1}\mathbf{p}_{i,i-1} + 2 {}^{i-1}\boldsymbol{\gamma}_{i-1} \times ({}^{i-1}\boldsymbol{\omega}_{i-1} \times {}^{i-1}\mathbf{p}_{i,i-1})] \\ & + {}^{i-1}\boldsymbol{\omega}_{i-1} \times [{}^{i-1}\boldsymbol{\omega}_{i-1} \times ({}^{i-1}\boldsymbol{\gamma}_{i-1} \times {}^{i-1}\mathbf{p}_{i,i-1})] \\ & + {}^{i-1}\boldsymbol{\omega}_{i-1} \times \{ {}^{i-1}\boldsymbol{\omega}_{i-1} \times [{}^{i-1}\boldsymbol{\omega}_{i-1} \times ({}^{i-1}\boldsymbol{\omega}_{i-1} \times {}^{i-1}\mathbf{p}_{i,i-1})] \} \\ & + \ddot{d}_i {}^{i-1}\hat{\mathbf{z}}_i + 4\dot{d}_i {}^{i-1}\boldsymbol{\nu}_{i-1} \times {}^{i-1}\hat{\mathbf{z}}_i + 8\dot{d}_i {}^{i-1}\boldsymbol{\gamma}_{i-1} \times ({}^{i-1}\boldsymbol{\omega}_{i-1} \times {}^{i-1}\hat{\mathbf{z}}_i) \\ & + 4\dot{d}_i {}^{i-1}\boldsymbol{\omega}_{i-1} \times [{}^{i-1}\boldsymbol{\gamma}_{i-1} \times {}^{i-1}\hat{\mathbf{z}}_i + {}^{i-1}\boldsymbol{\omega}_{i-1} \times ({}^{i-1}\boldsymbol{\omega}_{i-1} \times {}^{i-1}\hat{\mathbf{z}}_i)] \\ & + 6\ddot{d}_i [{}^{i-1}\boldsymbol{\gamma}_{i-1} \times {}^{i-1}\hat{\mathbf{z}}_i + {}^{i-1}\boldsymbol{\omega}_{i-1} \times ({}^{i-1}\boldsymbol{\omega}_{i-1} \times {}^{i-1}\hat{\mathbf{z}}_i)] \\ & \left. + 4\ddot{d}_i {}^{i-1}\boldsymbol{\omega}_{i-1} \times {}^{i-1}\hat{\mathbf{z}}_i \right\} \end{aligned} \quad (2.75)$$

$$\begin{aligned} {}^i\mathbf{s}_{c_i} = & {}^i\mathbf{s}_i + {}^i\boldsymbol{\varsigma}_i \times {}^i\mathbf{p}_{c_i,i} + 3 {}^i\boldsymbol{\nu}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{c_i,i}) \\ & + 3 {}^i\boldsymbol{\gamma}_i \times [{}^i\boldsymbol{\gamma}_i \times {}^i\mathbf{p}_{c_i,i} + {}^i\boldsymbol{\omega}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{c_i,i})] \\ & + {}^i\boldsymbol{\omega}_i \times [{}^i\boldsymbol{\nu}_i \times {}^i\mathbf{p}_{c_i,i} + 2 {}^i\boldsymbol{\gamma}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{c_i,i})] \\ & + {}^i\boldsymbol{\omega}_i \times \{ {}^i\boldsymbol{\omega}_i \times [{}^i\boldsymbol{\gamma}_i \times {}^i\mathbf{p}_{c_i,i} + {}^i\boldsymbol{\omega}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{c_i,i})] \} \end{aligned} \quad (2.76)$$

$${}^i\mathbf{F}_i = m_i {}^i\mathbf{a}_{c_i} \quad (2.77)$$

$${}^i\dot{\mathbf{F}}_i = m_i {}^i\dot{\mathbf{j}}_{c_i} \quad (2.78)$$

$${}^i\ddot{\mathbf{F}}_i = m_i {}^i\ddot{\mathbf{s}}_{c_i} \quad (2.79)$$

$${}^i\mathbf{N}_i = {}^i\mathbf{I}_i {}^i\boldsymbol{\gamma}_i + {}^i\boldsymbol{\omega}_i \times ({}^i\mathbf{I}_i {}^i\boldsymbol{\omega}_i) \quad (2.80)$$

$${}^i\dot{\mathbf{N}}_i = {}^i\boldsymbol{\omega}_i \times ({}^i\mathbf{I}_i {}^i\boldsymbol{\gamma}_i + {}^i\mathbf{N}_i) + {}^i\mathbf{I}_i ({}^i\boldsymbol{\gamma}_i \times {}^i\boldsymbol{\omega}_i + {}^i\boldsymbol{\nu}_i) + {}^i\boldsymbol{\gamma}_i \times {}^i\mathbf{I}_i {}^i\boldsymbol{\omega}_i \quad (2.81)$$

$$\begin{aligned} {}^i\ddot{\mathbf{N}}_i = & {}^i\boldsymbol{\gamma}_i \times ({}^i\mathbf{I}_i {}^i\boldsymbol{\gamma}_i + 2 {}^i\mathbf{N}_i) + {}^i\boldsymbol{\nu}_i \times {}^i\mathbf{I}_i {}^i\boldsymbol{\omega}_i \\ & + {}^i\mathbf{I}_i [2 ({}^i\boldsymbol{\nu}_i \times {}^i\boldsymbol{\omega}_i) + {}^i\boldsymbol{\omega}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\boldsymbol{\gamma}_i) + {}^i\boldsymbol{\varsigma}_i] \\ & + {}^i\boldsymbol{\omega}_i \times [{}^i\boldsymbol{\omega}_i \times {}^i\mathbf{I}_i {}^i\boldsymbol{\gamma}_i + 2 {}^i\mathbf{I}_i ({}^i\boldsymbol{\nu}_i \times {}^i\boldsymbol{\omega}_i) + {}^i\dot{\mathbf{N}}_i] \end{aligned} \quad (2.82)$$

$${}^i\mathbf{f}_i = {}^i\mathbf{R}_{i+1} {}^{i+1}\mathbf{f}_{i+1} + {}^i\mathbf{F}_i \quad (2.83)$$

$${}^i\dot{\mathbf{f}}_i = {}^i\mathbf{R}_{i+1} {}^{i+1}\dot{\mathbf{f}}_{i+1} + {}^i\dot{\mathbf{F}}_i \quad (2.84)$$

$${}^i\ddot{\mathbf{f}}_i = {}^i\mathbf{R}_{i+1} {}^{i+1}\ddot{\mathbf{f}}_{i+1} + {}^i\ddot{\mathbf{F}}_i \quad (2.85)$$

$${}^i\mathbf{n}_i = {}^i\mathbf{R}_{i+1} {}^{i+1}\mathbf{n}_{i+1} + {}^i\mathbf{p}_{c_i,i} \times {}^i\mathbf{F}_i + {}^i\mathbf{p}_{i+1,i} \times {}^i\mathbf{f}_{i+1} + {}^i\mathbf{N}_i \quad (2.86)$$

$$\begin{aligned} {}^i\dot{\mathbf{n}}_i = & {}^i\mathbf{R}_{i+1} {}^{i+1}\dot{\mathbf{n}}_{i+1} + ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{c_i,i}) \times {}^i\mathbf{F}_i + {}^i\mathbf{p}_{c_i,i} \times {}^i\dot{\mathbf{F}}_i \\ & + ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{i+1,i} + \dot{d}_{i+1} {}^i\hat{\mathbf{z}}_{i+1}) \times {}^i\mathbf{f}_{i+1} + {}^i\mathbf{p}_{i+1,i} \times {}^i\dot{\mathbf{f}}_{i+1} + {}^i\dot{\mathbf{N}}_i \end{aligned} \quad (2.87)$$

PART I - COMPLIANT MANIPULATORS

$$\begin{aligned}
{}^i\ddot{\mathbf{n}}_i &= {}^i\mathbf{R}_{i+1}{}^{i+1}\ddot{\mathbf{n}}_{i+1} + 2({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{c_i,i}) \times {}^i\dot{\mathbf{F}}_i \\
&+ [{}^i\boldsymbol{\gamma}_i \times {}^i\mathbf{p}_{c_i,i} + {}^i\boldsymbol{\omega}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{c_i,i})] \times {}^i\mathbf{F}_i \\
&+ [{}^i\boldsymbol{\gamma}_i \times {}^i\mathbf{p}_{i+1,i} + {}^i\boldsymbol{\omega}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{i+1,i}) + \ddot{d}_{i+1} {}^i\hat{\mathbf{z}}_{i+1} \\
&+ 2\dot{d}_{i+1} {}^i\boldsymbol{\omega}_i \times {}^i\hat{\mathbf{z}}_{i+1}] \times {}^i\mathbf{f}_{i+1} + {}^i\mathbf{p}_{c_i,i} \times {}^i\ddot{\mathbf{F}}_i + {}^i\ddot{\mathbf{N}}_i \\
&+ 2({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{p}_{i+1,i} + \dot{d}_{i+1} {}^i\hat{\mathbf{z}}_{i+1}) \times {}^i\dot{\mathbf{f}}_{i+1} + {}^i\mathbf{p}_{i+1,i} \times {}^i\ddot{\mathbf{f}}_{i+1}
\end{aligned} \tag{2.88}$$

$$\tau_{ei} = \begin{cases} {}^i\mathbf{n}_i^T \hat{\mathbf{z}}_0 & \text{revolute} \\ {}^i\mathbf{f}_i^T \hat{\mathbf{z}}_0 & \text{prismatic} \end{cases} \tag{2.89}$$

$$\dot{\tau}_{ei} = \begin{cases} ({}^i\dot{\mathbf{n}}_i + {}^i\mathbf{n}_i \times {}^i\boldsymbol{\omega}_i)^T \hat{\mathbf{z}}_0 & \text{revolute} \\ ({}^i\dot{\mathbf{f}}_i + {}^i\mathbf{f}_i \times {}^i\boldsymbol{\omega}_i)^T \hat{\mathbf{z}}_0 & \text{prismatic} \end{cases} \tag{2.90}$$

$$\ddot{\tau}_{ei} = \begin{cases} [{}^i\ddot{\mathbf{n}}_i + 2({}^i\dot{\mathbf{n}}_i \times {}^i\boldsymbol{\omega}_i) + {}^i\mathbf{n}_i \times {}^i\boldsymbol{\gamma}_i \\ \quad + ({}^i\mathbf{n}_i \times {}^i\boldsymbol{\omega}_i) \times {}^i\boldsymbol{\omega}_i]^T \hat{\mathbf{z}}_0 & \text{revolute} \\ [{}^i\ddot{\mathbf{f}}_i + 2({}^i\dot{\mathbf{f}}_i \times {}^i\boldsymbol{\omega}_i) + {}^i\mathbf{f}_i \times {}^i\boldsymbol{\gamma}_i \\ \quad + ({}^i\mathbf{f}_i \times {}^i\boldsymbol{\omega}_i) \times {}^i\boldsymbol{\omega}_i]^T \hat{\mathbf{z}}_0 & \text{prismatic} \end{cases} \tag{2.91}$$

$$\dot{\theta}_{mi} = \frac{\dot{\tau}_{ei}}{\sigma_i} + \dot{q}_i \tag{2.92}$$

$$\ddot{\theta}_{mi} = \frac{\ddot{\tau}_{ei}}{\sigma_i} + \ddot{q}_i \tag{2.93}$$

$$\tau_i = B_i \ddot{\theta}_{mi} + \tau_{ei} \tag{2.94}$$

where ${}^i\mathbf{f}_{i+1} = {}^i\mathbf{R}_{i+1}{}^{i+1}\mathbf{f}_{i+1}$.

As it can be noticed, (2.67)–(2.94) are very similar to (2.25)–(2.52), even identical in some cases. Obviously, the two variants of the algorithm, applied to the same robot, with coherent dynamic parameters, equivalent configuration $\mathbf{q}$ and same inputs $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$, will return the same output $\boldsymbol{\tau}$.

2.5 Spatial vector notation

Both versions of the algorithm presented above employ one form of the Denavit-Hartenberg notation. Kinematic quantities are represented with traditional $\mathbb{R}^3$ vectors, keeping linear and angular velocities as distinct entities, and holding masses, CoM positions and inertia tensors as separate parameters. A radically different approach is resorting to screw theory and Lie groups [57]. In this framework, linear and angular velocities are represented as unified $\mathbb{R}^6$ *spacial vectors*, while all inertial properties of a body are contained in a 6×6 inertia matrix. Arguably, this allows a more compact notation for representing dynamic equations. Also, the Denavit-Hartenberg convention is not needed anymore, and more general kinds of linkage, other

than revolute or prismatic joints, can be represented. Perhaps the best-known robotics works employing this notation are Featherstone's books [29, 30], which present all important dynamic algorithms, including RNEA, using screw theory.

Details about the spatial vector notation are out of the scope of this thesis. However, it is noteworthy to point out that the results presented in this work inspired a rewriting of the recursive fourth-order inverse dynamics, i.e. the core part of EJNEA, in terms of screw theory [56]. This contribution complements the existing algorithms presented in Featherstone's books.

PART I - COMPLIANT MANIPULATORS

Chapter 3

Extensions and Variants

3.1 Nonlinear springs

In the previous sections, linear elasticity was assumed. However, this hypothesis is restrictive: in general, nonlinearities are always present to some degrees, and sometimes they are even purposely introduced in Series Elastic Actuators [76]. Thus, in this section, it is shown how the linear elasticity assumption can be dropped. The resulting extended algorithm is called N-EJNEA, standing for Nonlinear EJNEA.

In the nonlinear case, the general model takes the form

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) = \boldsymbol{\tau}_e(\boldsymbol{\phi}) \quad (3.1a)$$

$$\mathbf{B}\ddot{\boldsymbol{\theta}}_m + \boldsymbol{\tau}_e(\boldsymbol{\phi}) = \boldsymbol{\tau}. \quad (3.1b)$$

Here, the flexibility torque $\boldsymbol{\tau}_e$ is a general vector expression of $\boldsymbol{\phi}$, decoupled for each joint. Its analytic formulation can be obtained from the expression of the total elastic energy $\mathcal{U}_e$ of the system:

$$\boldsymbol{\tau}_e = - \left(\frac{\partial \mathcal{U}_e}{\partial \mathbf{q}} \right)^T. \quad (3.2)$$

The stiffness matrix is physically defined as

$$\boldsymbol{\Sigma}(\boldsymbol{\phi}) = \frac{\partial \boldsymbol{\tau}_e}{\partial \boldsymbol{\phi}} = \frac{\partial \boldsymbol{\tau}_e}{\partial \boldsymbol{\theta}_m} = - \frac{\partial \boldsymbol{\tau}_e}{\partial \mathbf{q}}. \quad (3.3)$$

Notice that $\boldsymbol{\Sigma}$ is now a function of $\boldsymbol{\phi}$, which was not the case for linear springs. Since the motors dynamics are decoupled, it is still diagonal. For later use, we arrange the elements of its diagonal in vector $\boldsymbol{\sigma}(\boldsymbol{\phi}) \in \mathbb{R}^N$.

3.1.1 Inverse dynamics

When tackling the inverse dynamics problem for nonlinear elastic robots, the fundamental difference with respect to the linear case is that the stiffness Σ is not given a priori, but it needs to be computed. Further, it is not constant in time, which introduces additional terms. These considerations lead to the necessity of performing the required operations in a very precise order.

First of all, notice how (2.10)–(2.12) are still valid in this context. Thus, the first step is to apply these equations to obtain τ_e , $\dot{\tau}_e$ and $\ddot{\tau}_e$ from the desired joint trajectory.

Next, from the value of τ_e , we can compute the associated joint deflections ϕ . This amounts to the resolution of N nonlinear equations, which may be done analytically for simple cases, or numerically for more complex ones. This is an important difference with respect to the linear case.

Once the values of ϕ are obtained, we can compute Σ using the underlying spring model. Now, differentiating τ_e with respect to time, from (3.3) we obtain

$$\dot{\tau}_e = \Sigma \dot{\phi} \quad (3.4)$$

which is identical to (2.13), with the only difference that Σ has now been computed from ϕ . From this, $\dot{\phi}$ can be easily obtained by inverting Σ .

Exploiting the joint decoupling property of τ_e , we define

$$\mathbf{Z} = \frac{\partial \sigma}{\partial \phi} \quad (3.5)$$

which is again diagonal and computable from the spring model. This allows us to differentiate (3.4) as

$$\ddot{\tau}_e = \Sigma \ddot{\phi} + \mathbf{Z} \dot{\phi}^2 \quad (3.6)$$

where $\dot{\phi}$ is squared element-wise. This is easily rearranged as

$$\ddot{\theta}_m = \ddot{\mathbf{q}} + \Sigma^{-1}(\ddot{\tau}_e - \mathbf{Z} \dot{\phi}^2) \quad (3.7)$$

which is the nonlinear counterpart of (2.16). Finally, the value of τ can be obtained from (3.1b).

Notice that, for feedback linearization control, hardly anything changes with respect to the linear case. Assuming $\mathbf{q}$, $\dot{\mathbf{q}}$, ϕ and $\dot{\phi}$ are known, then Σ , $\mathbf{Z}$, τ_e and $\dot{\tau}_e$ can be computed from the spring model directly beforehand, so that $\ddot{\mathbf{q}}$ and $\ddot{\ddot{\mathbf{q}}}$ can be obtained in exactly the same way as in the linear case. Thus, the final control expression is practically identical to (2.20)

$$\begin{aligned} \tau = & \mathbf{B}\Sigma^{-1}(\phi)[\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}}_r + 2\dot{\mathbf{M}}(\mathbf{q})\dot{\mathbf{q}} + \ddot{\mathbf{M}}(\mathbf{q})\mathbf{q} + \ddot{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{Z}\dot{\phi}^2] \\ & + [\mathbf{M}(\mathbf{q}) + \mathbf{B}]\ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) \end{aligned} \quad (3.8)$$

where the dependency of Σ from ϕ was emphasized. Notably, since ϕ is already known, there is no need of solving the nonlinear equations.

3.1.2 The algorithm N-EJNEA

After the above discussion, extending the inverse dynamics algorithm to the case of nonlinear stiffness is straightforward. In the first place, equations (2.25)–(2.49) are still valid. Then, the only equations that need to be suitably replaced are (2.50)–(2.52). This is done as follows

$$\tau_{ei} \mapsto \phi_i \quad (3.9)$$

$$\sigma_i = \sigma_i(\phi_i) \quad (3.10)$$

$$\zeta_i = \frac{\partial \sigma_i}{\partial \phi_i} \quad (3.11)$$

$$\dot{\phi}_i = \frac{\dot{\tau}_{ei}}{\sigma_i} \quad (3.12)$$

$$\dot{\theta}_{mi} = \dot{q}_i + \dot{\phi}_i \quad (3.13)$$

$$\ddot{\theta}_{mi} = \ddot{q}_i + \frac{\ddot{\tau}_{ei} - \zeta_i \dot{\phi}_i^2}{\sigma_i} \quad (3.14)$$

$$\tau_i = B_i \ddot{\theta}_{mi} + \tau_{ei} \quad (3.15)$$

where (3.9) represents the computation of resolution of ϕ_i from τ_{ei} , while σ_i is the i -th element of σ , and ζ_i is the i -th diagonal element of $\mathbf{Z}$.

Feedback linearization can be simply achieved by applying N-EJNEA directly, with no modifications, once $\ddot{\mathbf{q}}$ and $\ddot{\ddot{\mathbf{q}}}$ have been obtained. Clearly, (3.9) and (3.12) can be skipped. The same result can be obtained just as directly by using regular EJNEA, as

$$\boldsymbol{\tau} = \text{EJNEA}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \ddot{\ddot{\mathbf{q}}}, \ddot{\ddot{\mathbf{q}}}_r) - \mathbf{B}\Sigma^{-1}\mathbf{Z}\dot{\boldsymbol{\phi}}^2 \quad (3.16)$$

where Σ and $\mathbf{Z}$ are precomputed from ϕ , and the resulting value of Σ is passed as a parameter to EJNEA.

3.2 Variable Stiffness Actuators

Variable Stiffness Actuators are fundamentally different from regular ones. The very definition of the inverse dynamics problem takes a new form, namely computing the motor torque commands that smoothly execute desired link

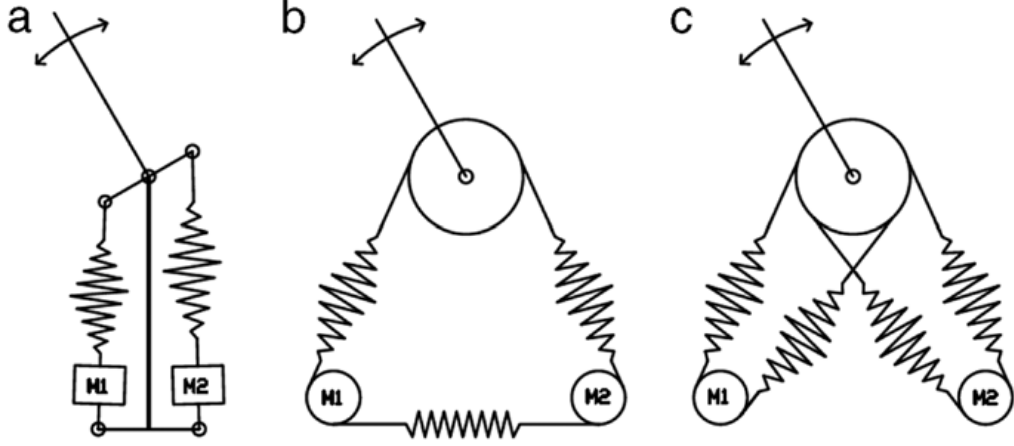


Figure 3.1 Some possible configurations of the motors for VSAs. Image adapted from [83].

and stiffness trajectories in nominal conditions. As mentioned in the Introduction, this requires *two* motors for each robot joint instead of one. As it turns out, the compliant transmission elements connecting each motor to the driven link necessarily need to be nonlinear, for the stiffness to be controllable. Notice that, technically, a single nonlinear spring is enough to make the stiffness variable, since it is not constant, but not to make it controllable for a given position. Variable Stiffness Actuators are broadly divided in two categories. The most prominent and widespread one is the antagonistic architecture. For this type of actuator the two motors, usually equally powerful, are independently connected to the actuated link through elastic connections; as they pull the link stronger in opposite directions, the stiffness increases. The second kind is the serial architecture, where the smaller, secondary motor is used only to modulate stiffness on line. Notably, a large class of single-DoF or multi-DoF VSA-based robots with complex nonlinear dynamics turns out to satisfy, under mild conditions, the necessary and sufficient conditions for feedback linearization and input-output decoupling [62, 22], a result that generalizes from the rigid case and from the case of robots with finite, but constant joint stiffness. For actuators of the antagonistic type, feedback linearization control was developed in [63].

In this section, the inverse dynamics problem for general multi-DoF VSA-based robots with antagonistic actuation is carefully analyzed. Next, the Variable Stiffness Actuation Newton-Euler Algorithm (VSA-NEA), itself a generalized version of EJNEA, is introduced and discussed.

3.2.1 Modeling

In robots having Variable Stiffness Actuation, since two actuators are present at each joint, the coordinate space is *tripled* in size as compared to the rigid case. The robot configuration is then given as

$$\Theta = \begin{pmatrix} \mathbf{q} \\ \boldsymbol{\theta}_{m1} \\ \boldsymbol{\theta}_{m2} \end{pmatrix} \in \mathbb{R}^{3N}, \quad (3.17)$$

where $\boldsymbol{\theta}_{m1}$ and $\boldsymbol{\theta}_{m2}$ are the motor positions after the reduction. Here, we will focus on the antagonistic type of actuators. Under the usual assumptions, the dynamic model is given as

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) = \boldsymbol{\tau}_{e1}(\boldsymbol{\phi}_1) + \boldsymbol{\tau}_{e2}(\boldsymbol{\phi}_2) \quad (3.18a)$$

$$\mathbf{B}_1\ddot{\boldsymbol{\theta}}_{m1} + \boldsymbol{\tau}_{e1}(\boldsymbol{\phi}_1) = \boldsymbol{\tau}_1 \quad (3.18b)$$

$$\mathbf{B}_2\ddot{\boldsymbol{\theta}}_{m2} + \boldsymbol{\tau}_{e2}(\boldsymbol{\phi}_2) = \boldsymbol{\tau}_2, \quad (3.18c)$$

where $\boldsymbol{\tau}_{e1}$ and $\boldsymbol{\tau}_{e2}$ are the flexibility torques, which are *nonlinear* functions of $\boldsymbol{\phi}_1$ and $\boldsymbol{\phi}_2$, decoupled for each joint. The nonlinearity in this case is fundamental, since it is what allows the controllability of the stiffness. The $6N$ -dimensional state of the system is given by $(\mathbf{q}, \boldsymbol{\theta}_{m1}, \boldsymbol{\theta}_{m2}, \dot{\mathbf{q}}, \dot{\boldsymbol{\theta}}_{m1}, \dot{\boldsymbol{\theta}}_{m2})$. The total flexibility torque acting on the links is

$$\boldsymbol{\tau}_e = \boldsymbol{\tau}_{e1}(\boldsymbol{\phi}_1) + \boldsymbol{\tau}_{e2}(\boldsymbol{\phi}_2) \quad (3.19)$$

The flexibility torques and the stiffness matrices are analytically obtained as:

$$\boldsymbol{\tau}_{ek} = \left(\frac{\partial \mathcal{U}_e}{\partial \boldsymbol{\phi}_k} \right)^T, \quad k = 1, 2, \quad \boldsymbol{\tau}_e = - \left(\frac{\partial \mathcal{U}_e}{\partial \mathbf{q}} \right)^T. \quad (3.20)$$

$$\boldsymbol{\Sigma}_k(\boldsymbol{\phi}_k) = \frac{\partial \boldsymbol{\tau}_{ek}}{\partial \boldsymbol{\phi}_k}, \quad k = 1, 2, \quad \boldsymbol{\Sigma}(\Theta) = - \frac{\partial \boldsymbol{\tau}_e}{\partial \mathbf{q}}. \quad (3.21)$$

The stiffness matrices are again diagonal, therefore their elements can be arranged in vectors $\boldsymbol{\sigma}_1 \in \mathbb{R}^N$, $\boldsymbol{\sigma}_2 \in \mathbb{R}^N$ and $\boldsymbol{\sigma} \in \mathbb{R}^N$. From (3.19) and (3.21), it follows

$$\boldsymbol{\Sigma} = \boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2 \quad (3.22)$$

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}_1(\boldsymbol{\phi}_1) + \boldsymbol{\sigma}_2(\boldsymbol{\phi}_2) \quad (3.23)$$

3.2.2 Inverse dynamics

For Variable Stiffness Actuators, the inverse dynamics problem is slightly different from the previous cases, since in general we also have a desired *stiffness* trajectory. Therefore, the task becomes the following: given a desired four-times differentiable link trajectory $\mathbf{q}_d(t)$ and a desired twice-differentiable stiffness trajectory $\boldsymbol{\sigma}_d(t)$ (with $\ddot{\mathbf{q}}_d$ and $\ddot{\boldsymbol{\sigma}}_d$ possibly piecewise continuous), we will compute the nominal input torques $\boldsymbol{\tau}_{d1}(t)$ and $\boldsymbol{\tau}_{d2}(t)$ that realize this combined task trajectory. Both reference trajectories $\mathbf{q}_d(t)$ and $\boldsymbol{\Sigma}_d(t)$ are chosen by the user, and thus all their needed time derivatives are known by design. In the rest of this section, we will omit time dependence and subscript d for notational simplicity.

As in the previous cases, the first step is to compute $\boldsymbol{\tau}_e$, $\dot{\boldsymbol{\tau}}_e$ and $\ddot{\boldsymbol{\tau}}_e$ from (2.10)–(2.12). Next, from the desired values of $\boldsymbol{\tau}_e$ and $\boldsymbol{\sigma}$, we need to compute the associated joint deflections $\boldsymbol{\phi}_1$ and $\boldsymbol{\phi}_2$. These are obtained by solving the following system of $2N$ nonlinear equations

$$\begin{cases} \boldsymbol{\tau}_e = \boldsymbol{\tau}_{e1}(\boldsymbol{\phi}_1) + \boldsymbol{\tau}_{e2}(\boldsymbol{\phi}_2) \\ \boldsymbol{\sigma} = \boldsymbol{\sigma}_1(\boldsymbol{\phi}_1) + \boldsymbol{\sigma}_2(\boldsymbol{\phi}_2), \end{cases} \quad (3.24)$$

where $\boldsymbol{\tau}_{e1}(\cdot)$, $\boldsymbol{\tau}_{e2}(\cdot)$, $\boldsymbol{\sigma}_1(\cdot)$, and $\boldsymbol{\sigma}_2(\cdot)$ are seen as functions expressed in their analytical forms. In general, system (3.24) needs to be solved numerically, exploiting the joint decoupling property which causes the split into N separate 2×2 subsystems (still nonlinear, though).

Next, we compute matrices $\boldsymbol{\Sigma}_1$, $\boldsymbol{\Sigma}_2$, $\mathbf{Z}_1$, $\mathbf{Z}_2$, $\mathbf{Z}$, $\mathbf{H}_1$ and $\mathbf{H}_2$. These quantities are all obtained from the underlying spring model, using the solutions of (3.24). Matrices $\mathbf{Z}_1$, $\mathbf{Z}_2$, $\mathbf{Z}$, $\mathbf{H}_1$, and $\mathbf{H}_2$ are diagonal, defined as

$$\mathbf{Z}_k = \frac{\partial \boldsymbol{\sigma}_k}{\partial \boldsymbol{\phi}_k} = \text{diag}(\boldsymbol{\zeta}_k) \quad (3.25)$$

$$\mathbf{Z} = \mathbf{Z}_1 + \mathbf{Z}_2 \quad (3.26)$$

$$\mathbf{H}_k = \frac{\partial \boldsymbol{\zeta}_k}{\partial \boldsymbol{\phi}_k} \quad (3.27)$$

the diagonal elements of all these matrices being denoted as $\sigma_{k,i}$, $\zeta_{k,i}$, ζ_i , and $\eta_{k,i}$, with $k = 1, 2$ and $i = 1 \dots N$.

From (3.19) and (3.21), we have then (with dependencies omitted)

$$\dot{\boldsymbol{\tau}}_e = \boldsymbol{\Sigma}_1 \dot{\boldsymbol{\phi}}_1 + \boldsymbol{\Sigma}_2 \dot{\boldsymbol{\phi}}_2. \quad (3.28)$$

Exploiting the decentralized property of $\boldsymbol{\tau}_{e1}$ and $\boldsymbol{\tau}_{e2}$ and using (3.25), we can differentiate (3.23) and obtain

$$\dot{\boldsymbol{\sigma}} = \mathbf{Z}_1 \dot{\boldsymbol{\phi}}_1 + \mathbf{Z}_2 \dot{\boldsymbol{\phi}}_2. \quad (3.29)$$

CHAPTER 3. EXTENSIONS AND VARIANTS

Putting (3.28) and (3.29) together, we set up the linear system

$$\begin{pmatrix} \Sigma_1 & \Sigma_2 \\ Z_1 & Z_2 \end{pmatrix} \begin{pmatrix} \dot{\phi}_1 \\ \dot{\phi}_2 \end{pmatrix} = \mathcal{A}(\phi_1, \phi_2) \begin{pmatrix} \dot{\phi}_1 \\ \dot{\phi}_2 \end{pmatrix} = \begin{pmatrix} \dot{\tau}_e \\ \dot{\sigma} \end{pmatrix}, \quad (3.30)$$

where $\mathcal{A}(\phi_1, \phi_2)$ is the *decoupling* matrix [22], in which we have emphasized the dependency from ϕ_1 and ϕ_2 . Provided the decoupling matrix is nonsingular, system (3.30) can be easily solved for $\dot{\phi}_1$ and $\dot{\phi}_2$ as

$$\begin{pmatrix} \dot{\phi}_1 \\ \dot{\phi}_2 \end{pmatrix} = \begin{pmatrix} \Sigma_1 & \Sigma_2 \\ Z_1 & Z_2 \end{pmatrix}^{-1} \begin{pmatrix} \dot{\tau}_e \\ \dot{\sigma} \end{pmatrix}. \quad (3.31)$$

We now go one level of differentiation further. Differentiating (3.28) and rearranging, we obtain

$$\Sigma_1 \ddot{\theta}_{m1} + \Sigma_2 \ddot{\theta}_{m2} = \ddot{\tau}_e - Z_1 \dot{\phi}_1^2 - Z_2 \dot{\phi}_2^2 + \Sigma \ddot{q} = \beta_1 \quad (3.32)$$

where vectors $\dot{\phi}_1$ and $\dot{\phi}_2$ are squared component-wise. Using (3.27), we can differentiate also (3.29) and rearrange it as

$$Z_1 \ddot{\theta}_{m1} + Z_2 \ddot{\theta}_{m2} = \ddot{\sigma} - H_1 \dot{\phi}_1^2 - H_2 \dot{\phi}_2^2 + Z \ddot{q} = \beta_2 \quad (3.33)$$

Equations (3.32) and (3.33) result in another linear system

$$\begin{pmatrix} \Sigma_1 & \Sigma_2 \\ Z_1 & Z_2 \end{pmatrix} \begin{pmatrix} \ddot{\theta}_{m1} \\ \ddot{\theta}_{m2} \end{pmatrix} = \mathcal{A}(\phi_1, \phi_2) \begin{pmatrix} \ddot{\theta}_{m1} \\ \ddot{\theta}_{m2} \end{pmatrix} = \begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix}, \quad (3.34)$$

which has the same coefficient matrix $\mathcal{A}$ that appears in (3.30), making only one inversion necessary to obtain also $\ddot{\theta}_{m1}$ and $\ddot{\theta}_{m2}$. We obtain finally τ_1 and τ_2 from (3.18b) and (3.18c). Also, τ_{e1} and τ_{e2} are evaluated from the spring model using the computed values of ϕ_1 and ϕ_2 , while $\dot{\theta}_{m1} = \dot{\phi}_1 + \dot{q}$ and $\dot{\theta}_{m2} = \dot{\phi}_2 + \dot{q}$. Some remarks are now in order.

- Computation will be terminated if (3.24) has no solution. This means that the desired combined motion-stiffness task is not attainable by the given actuator model. Instead, the possible existence of multiple solutions is not a source of problems. In fact, when the system is solved for a (discretized) time sequence of desired data, at each instant the solution is found by a numerical search, starting from the result obtained at the previous time step. Since the search is a local process, the new solution will always be close to the older one, achieving continuity.

PART I - COMPLIANT MANIPULATORS

- If $\mathcal{A}$ is not invertible, the system is in a dynamic singularity, restricting the desired motion-stiffness task that can be achieved. Thanks to the joint decoupling property, $\mathcal{A}$ can be rearranged into a block-diagonal matrix with 2×2 blocks, which can be inverted separately from each other. Thus matrix $\mathcal{A}$ is singular if and only if at least one of these N blocks is singular. For instance, for a joint i with two identical antagonistic springs, symmetric for torsion and tension, this will always happen when $\phi_{1,i} = \phi_{2,i}$.
- If the initial values of ϕ_1 and ϕ_2 are known, we can avoid the explicit resolution of (3.24) at each time instant. Instead, we can solve (3.30) and integrate then for ϕ_1 and ϕ_2 . Caution must be taken, however, due to error drifts caused by numerical integration.

3.2.3 Special cases

In this section, we show how system (3.24) can be solved for some specific types of springs.

The first case is that of a *cubic* flexible torque profile. The cubic model is one of the most commonly used approximations for complex spring characteristics, since the cubic term is the lowest-order correction which can be added to the linear model in order to represent a generic function more closely, while maintaining a positive-definite elastic potential energy. Assuming the two springs acting on each joint are identical to each other, then, for the k -th transmission element of the i -th joint

$$\mathcal{U}_{ek,i} = \frac{1}{2}K_i\phi_{k,i}^2 + \frac{1}{4}K_{ci}\phi_{k,i}^4 \quad (3.35)$$

which, remembering (3.20), results in the joint torque

$$\tau_{ek,i} = \frac{\partial \mathcal{U}_{ek,i}}{\partial \phi_{k,i}} = K_i\phi_{k,i} + K_{ci}\phi_{k,i}^3. \quad (3.36)$$

so that the full multi-DoF actuation model is given as

$$\tau_{e1}(\phi_1) = \mathbf{K}\phi_1 + \mathbf{K}_c\phi_1^3 \quad (3.37a)$$

$$\tau_{e2}(\phi_2) = \mathbf{K}\phi_2 + \mathbf{K}_c\phi_2^3 \quad (3.37b)$$

where ϕ^3 indicates element-wise elevation to the third power, while $\mathbf{K} = \text{diag}\{K_i\} \in \mathbb{R}^{N \times N}$ and $\mathbf{K}_c = \text{diag}\{K_{ci}\} \in \mathbb{R}^{N \times N}$ are constant, diagonal,

CHAPTER 3. EXTENSIONS AND VARIANTS

positive definite matrices. Therefore, for $k = 1, 2$ and $i = 1, \dots, N$, we have

$$\sigma_{k,i} = K_i + 3K_{ci} \phi_{k,i}^2 \quad (3.38)$$

$$\zeta_{k,i} = 6K_{ci} \phi_{k,i} \quad (3.39)$$

$$\eta_{k,i} = 6K_{ci} \quad (3.40)$$

yielding, for each joint, the system

$$\tau_{ei} = K_i(\phi_{1,i} + \phi_{2,i}) + K_{ci}(\phi_{1,i}^3 + \phi_{2,i}^3) \quad (3.41a)$$

$$\sigma_i = 2K_i + 3K_{ci}(\phi_{1,i}^2 + \phi_{2,i}^2) \quad (3.41b)$$

which is the joint-wise expression of (3.24) for the cubic case. The minimum possible stiffness for joint i is thus $\sigma_i = 2K_i$. In order to avoid singularity problems, this lower bound should never be reached. We solve now (3.24) as follows. Defining from (3.41b)

$$R_i^2 = \phi_{1,i}^2 + \phi_{2,i}^2 = \frac{\sigma_i - 2K_i}{3K_{ci}} \quad (3.42)$$

it can be seen that all solutions are parametrized by a scalar $\xi_i \in (-\pi, \pi]$ such that $\phi_{1,i} = R_i \cos \xi_i$ and $\phi_{2,i} = R_i \sin \xi_i$. The value of R_i can be chosen as positive, without loss of generality. Replacing in (3.41a) we get

$$f(\xi_i) = (\cos \xi_i + \sin \xi_i) + \frac{\sigma_i - 2K_i}{3K_i}(\cos^3 \xi_i + \sin^3 \xi_i) = \frac{\tau_{ei}}{K_i R_i} \quad (3.43)$$

which is a single trigonometric equation in ξ_i . It can be noticed that choosing $\phi_{1,i} = R_i \sin \xi_i$ and $\phi_{2,i} = R_i \cos \xi_i$ would have resulted in exactly the same equation, yielding equivalent specular solutions for the two deflections.

Equation (3.43) cannot be solved analytically. However, it is sufficiently smooth, and thus easily solvable by a numerical root finder. Moreover, we can describe quickly verifiable necessary and sufficient conditions for solutions to exist. Specifically, the elastic torque modulus $|\tau_{ei}|$ must not exceed some given maximum values, which are function of the stiffness σ_i . The values are obtained by finding the points where $f(\xi_i)$ is maximized.

For $2K_i \leq \sigma_i \leq 4K_i$, a global maximum is found as

$$|\tau_{ei}|_{max1} = \frac{\sqrt{2}}{6} R_i (4K_i + \sigma_i) \quad (3.44)$$

while for $\sigma_i > 4K_i$, two points are found such that

$$|\tau_{ei}|_{max2} = \sqrt{\frac{1}{K_{ci}} \left(\frac{\sigma_i}{3}\right)^3} \quad (3.45)$$

while $|\tau_{ei}|_{max1} < |\tau_{ei}|_{max2}$ becomes a local minimum. Therefore, assuming $\sigma_i \geq 2K_i$

PART I - COMPLIANT MANIPULATORS

- At least one solution can always be found for $|\tau_{ei}| \leq |\tau_{ei}|_{max1}$. They are exactly two if the inequality holds strictly, while it is exactly one if $|\tau_{ei}| = |\tau_{ei}|_{max1}$ and $\sigma \leq 4K_i$. For $\sigma \leq 4K_i$, $|\tau_{ei}|$ cannot be larger than $|\tau_{ei}|_{max1}$ and no other solutions exist.
- If $\sigma > 4K_i$, additional solutions exist for $|\tau_{ei}|_{max1} \leq |\tau_{ei}| \leq |\tau_{ei}|_{max2}$. There are exactly four solutions if both inequalities hold strictly, exactly three if $|\tau_{ei}| = |\tau_{ei}|_{max1}$, and exactly two if $|\tau_{ei}| = |\tau_{ei}|_{max2}$.

For $\sigma > 4K_i$, then it is possible to find continuous solutions when $|\tau_{ei}|$ increases from below $|\tau_{ei}|_{max1}$ up to $|\tau_{ei}|_{max2}$. However, when we switch from $|\tau_{ei}| \geq |\tau_{ei}|_{max1}$ to $|\tau_{ei}| < |\tau_{ei}|_{max1}$, a jump discontinuity in the solutions might arise, depending on the solution “branch” which is being followed. Notice that there is no upper bound on σ_i . However, in the case of the cubic model (and also more in general), it is recommended to set a limit, in order to keep spring deformations reasonably limited; a good value is $\sigma_i = 4K_i$, which is twice the minimum. This also avoids the problem of the discontinuities. Detailed derivation of the elastic torque bounds can be found in Appendix B.1.

An even lower-order spring profile is the *quadratic* one. However in this case, for the potential energy to be positive definite, an absolute value must be inserted

$$\mathcal{U}_{ek,i} = \frac{1}{2}K_i\phi_{k,i}^2 + \frac{1}{3}K_{qi}|\phi_{k,i}|^3 \quad (3.46)$$

resulting in the joint torque

$$\tau_{ek,i} = K_i\phi_{k,i} + K_{qi}\text{sign}(\phi_{k,i})\phi_{k,i}^2 \quad (3.47)$$

where conventionally $\text{sign}(0) = 0$. The full actuation model is

$$\boldsymbol{\tau}_{e1}(\boldsymbol{\phi}_1) = \mathbf{K}\boldsymbol{\phi}_1 + \mathbf{K}_q\text{sign}(\boldsymbol{\phi}_1)\boldsymbol{\phi}_1^2 \quad (3.48a)$$

$$\boldsymbol{\tau}_{e2}(\boldsymbol{\phi}_2) = \mathbf{K}\boldsymbol{\phi}_2 + \mathbf{K}_q\text{sign}(\boldsymbol{\phi}_2)\boldsymbol{\phi}_2^2 \quad (3.48b)$$

with constant, diagonal, positive definite $\mathbf{K} \in \mathbb{R}^{N \times N}$ and $\mathbf{K}_q \in \mathbb{R}^{N \times N}$, and the sign function evaluated element-wise. Then, for $k = 1, 2$ and $i = 1, \dots, N$

$$\sigma_{k,i} = K_i + 2K_{qi}|\phi_{k,i}| \quad (3.49)$$

$$\zeta_{k,i} = 2K_{qi}\text{sign}(\phi_{k,i}) \quad (3.50)$$

$$\eta_{k,i} = 0. \quad (3.51)$$

As a technical remark, it should be noticed that (3.47) and (3.49) are always valid expressions of $\tau_{ek,i}$ and $\sigma_{k,i}$, while for (3.50) and (3.51) the notation

CHAPTER 3. EXTENSIONS AND VARIANTS

was slightly abused, since these partial derivatives are technically not defined for $\phi_{k,i} = 0$ due to the non-differentiability of $\sigma_{k,i}$ in that point. However, we will see this is not a problem. For the i -th joint, we obtain the system

$$\tau_{ei} = K_i(\phi_{1,i} + \phi_{2,i}) + K_{qi}(\text{sign}(\phi_{1,i})\phi_{1,i}^2 + \text{sign}(\phi_{2,i})\phi_{2,i}^2) \quad (3.52a)$$

$$\sigma_i = 2K_i + 2K_{qi}(|\phi_{1,i}| + |\phi_{2,i}|) \quad (3.52b)$$

where it can be seen that, also in this case, it is always $\sigma_i \geq 2K_i$. This system can be solved if and only if

$$|\tau_{ei}| \leq \frac{\sigma_i(\sigma_i - 2K_i)}{4K_{qi}} \quad (3.53)$$

with two possible pairs of solutions

$$\phi_{1,i} = \frac{\tau_{ei}}{\sigma_i} \pm \frac{\sigma_i - 2K_i}{4K_{qi}} \quad (3.54a)$$

$$\phi_{2,i} = \frac{\tau_{ei}}{\sigma_i} \mp \frac{\sigma_i - 2K_i}{4K_{qi}} \quad (3.54b)$$

which are opposite in sign when (3.53) holds. It can be verified that one of these values can be equal to zero if and only if the equality holds for (3.53), thereby ruling out the problem of $\zeta_{k,i}$ and $\eta_{k,i}$ being undefined if the upper bound for $|\tau_{ei}|$ is avoided. Alternative solutions can be found when additional conditions hold. When

$$\frac{\sigma_i^2 - 4K_i^2}{8K_{qi}} \leq |\tau_{ei}| \leq \frac{\sigma_i(\sigma_i - 2K_i)}{4K_{qi}} \quad (3.55)$$

then two alternative pairs of solutions can be found. For each of these pairs, both deflection values are nonnegative for $\tau_{ei} > 0$ and nonpositive when $\tau_{ei} < 0$. These solutions are only continuous with (3.54) when $|\tau_{ei}|$ reaches its upper bound, and are therefore of limited interest. Their exact values, together with all derivations, can be found in Appendix B.2. No other solutions exist.

3.2.4 Feedback linearization control

The same procedure outlined in Sec. 3.2.2 can be used, with minor modifications, to implement a feedback linearization law for VSA robots. With the desired trajectories $\mathbf{q}_d(t)$ and $\boldsymbol{\sigma}_d(t)$, we compute the linear control signals $\ddot{\mathbf{q}}_r$

PART I - COMPLIANT MANIPULATORS

and $\ddot{\sigma}_r$ from the *actual* values of $\mathbf{q}$, $\dot{\mathbf{q}}$, $\ddot{\mathbf{q}}$, $\ddot{\mathbf{q}}_r$, σ , and $\dot{\sigma}$

$$\ddot{\mathbf{q}}_r = \ddot{\mathbf{q}}_d + \mathbf{K}_3(\ddot{\mathbf{q}}_d - \ddot{\mathbf{q}}) + \mathbf{K}_2(\dot{\mathbf{q}}_d - \dot{\mathbf{q}}) + \mathbf{K}_1(\mathbf{q}_d - \mathbf{q}) + \mathbf{K}_0(\dot{\mathbf{q}}_d - \dot{\mathbf{q}}) \quad (3.56a)$$

$$\ddot{\sigma}_r = \ddot{\sigma}_d + \mathbf{K}_{\sigma_1}(\dot{\sigma}_d - \dot{\sigma}) + \mathbf{K}_{\sigma_0}(\sigma_d - \sigma), \quad (3.56b)$$

where all six gain matrices are diagonal with their diagonal elements being such that the two characteristic polynomials

$$p_i(s) = s^4 + K_{3,i}s^3 + K_{2,i}s^2 + K_{1,i}s + K_{0,i} \quad (3.57a)$$

$$p_{\sigma i}(s) = s^2 + K_{\sigma 1,i}s + K_{\sigma 0,i} \quad (3.57b)$$

are both Hurwitz. Then, the motor torques achieving feedback linearization control are obtained using the previous inverse dynamics computations, by only replacing $\ddot{\mathbf{q}}_d$ with $\ddot{\mathbf{q}}_r$ and $\ddot{\sigma}_d$ with $\ddot{\sigma}_r$. Use of high gains and possible addition of an integral action to $\ddot{\mathbf{q}}_r$ and $\ddot{\sigma}_r$ (with similar requisites on the gain coefficients) can contribute to the controller accuracy and robustness to dynamic modeling errors.

Since the control scheme requires the current values of $\ddot{\mathbf{q}}$, $\ddot{\mathbf{q}}_r$, σ , and $\dot{\sigma}$, it is necessary to compute these quantities from measured ones. We assume *full state* measurements are available, with the values of $\mathbf{q}$, ϕ_1 , and ϕ_2 given, and those of $\dot{\mathbf{q}}$, $\dot{\phi}_1$, and $\dot{\phi}_2$ obtained by numerical differentiation. Now, the values of τ_{e1} , τ_{e2} , Σ_1 , Σ_2 , $\mathbf{Z}_1$, and $\mathbf{Z}_2$ (as well as of $\mathbf{H}_1$ and $\mathbf{H}_2$) can be immediately computed from ϕ_1 and ϕ_2 . Next, τ_e , σ , $\dot{\tau}_e$ and $\dot{\sigma}$ are obtained from (3.19), (3.23), (3.28), and (3.29), respectively. An alternative to obtain these quantities is by using an estimator, as shown in [31] for σ ; for practical purposes, this is actually preferable to the crude application of the model. Finally, $\ddot{\mathbf{q}}$ and $\ddot{\mathbf{q}}_r$ are obtained from (2.23) and (2.24), as in the constant-stiffness case.

Notice that the resolution of (3.24) is no longer necessary. This means that no numerical root finding is required for feedback linearization, which also eliminates the possibility of failure at this step. Finally, equations (2.10), (2.11) and (3.31) are also skipped as such.

3.2.5 The algorithm VSA-NEA

In the following, the final inverse dynamics algorithm for robots equipped with Variable Stiffness Actuators is presented. We refer to it as VSA-NEA, i.e. Variable Stiffness Actuators Newton-Euler Algorithm. It can be obtained by replacing (2.50)–(2.52) with the following

$$\langle \tau_{ei}, \sigma_i \rangle \mapsto \langle \phi_{1,i}, \phi_{2,i} \rangle \quad (3.58)$$

$$\tau_{e1,i} = \tau_{e1,i}(\phi_{1,i}) \quad (3.59)$$

$$\tau_{e2,i} = \tau_{e2,i}(\phi_{2,i}) \quad (3.60)$$

$$\sigma_{1,i} = \sigma_{1,i}(\phi_{1,i}) \quad (3.61)$$

$$\sigma_{2,i} = \sigma_{2,i}(\phi_{2,i}) \quad (3.62)$$

$$\zeta_{1,i} = \frac{\partial \sigma_{1,i}}{\partial \phi_{1,i}} \quad (3.63)$$

$$\zeta_{2,i} = \frac{\partial \sigma_{2,i}}{\partial \phi_{2,i}} \quad (3.64)$$

$$\zeta_i = \zeta_{1,i} + \zeta_{2,i} \quad (3.65)$$

$$\eta_{1,i} = \frac{\partial^2 \sigma_{1,i}}{\partial \phi_{1,i}^2} \quad (3.66)$$

$$\eta_{2,i} = \frac{\partial^2 \sigma_{2,i}}{\partial \phi_{2,i}^2} \quad (3.67)$$

$$A_i^{-1} = \frac{1}{\sigma_{1,i}\zeta_{2,i} - \sigma_{2,i}\zeta_{1,i}} \begin{pmatrix} \zeta_{2,i} & -\sigma_{2,i} \\ -\zeta_{1,i} & \sigma_{1,i} \end{pmatrix} \quad (3.68)$$

$$\begin{pmatrix} \dot{\phi}_1 \\ \dot{\phi}_2 \end{pmatrix} = A_i^{-1} \begin{pmatrix} \dot{\tau}_{ei} \\ \dot{\sigma}_i \end{pmatrix} \quad (3.69)$$

$$\dot{\theta}_{m1,i} = \dot{\phi}_{1,i} + \dot{q}_i \quad (3.70)$$

$$\dot{\theta}_{m2,i} = \dot{\phi}_{2,i} + \dot{q}_i \quad (3.71)$$

$$\beta_{1,i} = \ddot{\tau}_{ei} - \zeta_{1,i}\dot{\phi}_{1,i}^2 - \zeta_{2,i}\dot{\phi}_{2,i}^2 + \sigma_i\ddot{q}_i \quad (3.72)$$

$$\beta_{2,i} = \ddot{\sigma}_i - \eta_{1,i}\dot{\phi}_{1,i}^2 - \eta_{2,i}\dot{\phi}_{2,i}^2 + \zeta_i\ddot{q}_i \quad (3.73)$$

$$\begin{pmatrix} \ddot{\theta}_{m1,i} \\ \ddot{\theta}_{m2,i} \end{pmatrix} = A_i^{-1} \begin{pmatrix} \beta_{1,i} \\ \beta_{2,i} \end{pmatrix} \quad (3.74)$$

$$\tau_{1,i} = B_{1,i}\ddot{\theta}_{m1,i} + \tau_{e1,i} \quad (3.75)$$

$$\tau_{2,i} = B_{2,i}\ddot{\theta}_{m2,i} + \tau_{e2,i} \quad (3.76)$$

with obvious meaning of the symbols.

3.2.6 Variant for feedback linearization

A variant of the VSA-NEA algorithm can be used to compute a feedback linearization control, as explained in Sec. 3.2.4. Equations (3.59)–(3.67) are evaluated first in a separate initial loop; next, (3.19), (3.23), (3.28) and (3.29)

are evaluated so to obtain τ_e , σ , $\dot{\tau}_e$ and $\dot{\sigma}$, respectively. From these values, $\ddot{\mathbf{q}}$ and $\ddot{\mathbf{j}}$ are obtained as explained in Sec. 2.3. Finally, the standard version of VSA-NEA is run, replacing $\ddot{\mathbf{q}}$ and $\ddot{\sigma}$ with $\ddot{\mathbf{q}}_r$ and $\ddot{\sigma}_r$, respectively; during execution, equations (2.47)–(2.48), (3.58)–(3.67) and (3.69) can be skipped, since their outputs have already been computed in the previous steps.

3.3 Friction

In the previous sections, the presence of energy-dissipating elements, i.e. of friction, was not considered. In this section, we will describe how they can be included in inverse dynamics computations. Here, we will focus on the case of constant, linear stiffness, the discussion for nonlinear and variable stiffness being analogous.

In general, friction forces are quite difficult to model. The most common simplifying assumption is that they act *locally* on each joint, in a decoupled fashion. This excludes, notably, the viscous friction of the robot structure with the air, so to concentrate on dissipative phenomena acting within the robot electromechanical components. Therefore, friction can be present at the motor side or at the link side. Common kinds of friction are *viscous friction*, which is proportional to velocity, and *dry friction*, which is constant in module. Another possible source of dissipation can be on the connecting elastic element, in which case we speak of *spring damping* or *viscoelastic friction*. Each case will be treated separately.

The simplest case concerns motor-side friction. This may be caused, in particular, by the electromagnetic interaction between rotor and stator of the motor. The presence of viscous and dry friction on the motor side is expressed by adding the dissipative terms to (2.8b) as

$$\mathbf{B}\ddot{\boldsymbol{\theta}}_m + \mathbf{D}_m\dot{\boldsymbol{\theta}}_m + \mathbf{F}_m \text{sign}(\dot{\boldsymbol{\theta}}_m) + \boldsymbol{\tau}_e = \boldsymbol{\tau} \quad (3.77)$$

where $\mathbf{F}_m \in \mathbb{R}^{N \times N}$ and $\mathbf{D}_m \in \mathbb{R}^{N \times N}$ are diagonal, positive definite matrices containing, respectively, dry and viscous friction coefficients, and the sign function is evaluated element-wise. Thus, it becomes necessary to compute $\dot{\boldsymbol{\theta}}_m$. This is done by first computing $\dot{\tau}_e$ through (2.11), and then using (2.15). The final result is simply plugged into (3.77) together with $\ddot{\boldsymbol{\theta}}_m$ and $\boldsymbol{\tau}_e$ to obtain $\boldsymbol{\tau}$. In the implementation of the algorithm, now (2.48) and (2.50), which were not needed in absence of motor friction, have to be included, while (2.52) is replaced by

$$\tau_i = B_i\ddot{\theta}_{mi} + D_{mi}\dot{\theta}_{mi} + F_{mi} \text{sign}(\dot{\theta}_{mi}) + \tau_{ei} . \quad (3.78)$$

CHAPTER 3. EXTENSIONS AND VARIANTS

Notice that the computation may be performed just as easily by using a friction-free implementation of EJNEA, by simply doing

$$\boldsymbol{\tau} = \text{EJNEA}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \ddot{\ddot{\mathbf{q}}}, \ddot{\ddot{\ddot{\mathbf{q}}}}) + \mathbf{D}_m \dot{\boldsymbol{\theta}}_m + \mathbf{F}_m \text{sign}(\dot{\boldsymbol{\theta}}_m) \quad (3.79)$$

as long as the computation of $\dot{\boldsymbol{\theta}}_m$, or even just $\dot{\boldsymbol{\tau}}_e$, is included in the algorithm and the resulting value is returned as an extra output. Indeed, any model of motor-side friction can be straightforwardly included in (3.78) or (3.79). Incidentally, we notice that friction acting on the motor side can be fully compensated via control by measuring $\boldsymbol{\theta}_m$.

Friction on the link side is primarily caused by mechanical contact between consecutive links. It should be noticed that a careful mechanical design of the robot could reduce this dissipative action to zero, for instance introducing ball bearings as in the DLR lightweight robot [1]. Otherwise, viscous and dry friction can be modeled by adding the appropriate terms to (2.10), obtaining

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{D}_l \dot{\mathbf{q}} + \mathbf{F}_l \text{sign}(\dot{\mathbf{q}}) = \boldsymbol{\tau}_e \quad (3.80)$$

where $\mathbf{F}_l \in \mathbb{R}^{N \times N}$ and $\mathbf{D}_l \in \mathbb{R}^{N \times N}$ are the diagonal positive definite matrices of dry and viscous friction coefficients on the link side. However, in the case of link-side friction, it is necessary to approximate the model to a smooth function, in order to differentiate the link equations. This is done as

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{D}_l \dot{\mathbf{q}} + \tilde{\mathbf{F}}(\dot{\mathbf{q}}) = \boldsymbol{\tau}_e \quad (3.81)$$

where $\tilde{\mathbf{F}}(\dot{\mathbf{q}})$ is a smooth approximation of the sign function, for instance, a hyperbolic tangent. Differentiating (3.80) twice, we obtain

$$\mathbf{M}(\mathbf{q})\ddot{\ddot{\mathbf{q}}} + \dot{\mathbf{M}}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}}) + (\mathbf{D}_l + \frac{\partial \tilde{\mathbf{F}}}{\partial \dot{\mathbf{q}}})\ddot{\mathbf{q}} = \dot{\boldsymbol{\tau}}_e \quad (3.82)$$

$$\mathbf{M}(\mathbf{q})\ddot{\ddot{\ddot{\mathbf{q}}}} + 2\dot{\mathbf{M}}(\mathbf{q})\ddot{\ddot{\mathbf{q}}} + \ddot{\mathbf{M}}(\mathbf{q})\ddot{\mathbf{q}} + \ddot{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}}) + (\mathbf{D}_l + \frac{\partial \tilde{\mathbf{F}}}{\partial \dot{\mathbf{q}}})\ddot{\ddot{\mathbf{q}}} + \frac{\partial^2 \tilde{\mathbf{F}}}{\partial \dot{\mathbf{q}}^2} \ddot{\mathbf{q}}^2 = \ddot{\boldsymbol{\tau}}_e \quad (3.83)$$

where notation was slightly abused for $\frac{\partial^2 \tilde{\mathbf{F}}}{\partial \dot{\mathbf{q}}^2}$. Once $\boldsymbol{\tau}_e$, $\ddot{\boldsymbol{\tau}}_e$ and possibly $\dot{\boldsymbol{\tau}}_e$ are computed from (3.80), (3.83) and (3.82), computation goes on as normal. In the algorithm implementation, equations (2.47)–(2.49) are replaced with

$$\boldsymbol{\tau}_{ei} = {}^i\mathbf{n}_i^T {}^i\hat{\mathbf{z}}_{i-1} + D_{li}\dot{q}_i + F_{li} \text{sign}(\dot{q}_i) \quad (3.84)$$

$$\dot{\boldsymbol{\tau}}_{ei} = ({}^i\dot{\mathbf{n}}_i + {}^i\mathbf{n}_i \times {}^i\boldsymbol{\omega}_i)^T {}^i\hat{\mathbf{z}}_{i-1} + (D_{li} + \frac{\partial \tilde{F}_i}{\partial \dot{q}_i})\ddot{q}_i \quad (3.85)$$

$$\begin{aligned} \ddot{\boldsymbol{\tau}}_{ei} = & [{}^i\ddot{\mathbf{n}}_i + 2({}^i\dot{\mathbf{n}}_i \times {}^i\boldsymbol{\omega}_i) + {}^i\mathbf{n}_i \times {}^i\boldsymbol{\gamma}_i + ({}^i\mathbf{n}_i \times {}^i\boldsymbol{\omega}_i) \times {}^i\boldsymbol{\omega}_i]^T {}^i\hat{\mathbf{z}}_{i-1} \\ & + (D_{li} + \frac{\partial \tilde{F}_i}{\partial \dot{q}_i})\ddot{\ddot{q}}_i + \frac{\partial^2 \tilde{F}_i}{\partial \dot{q}_i^2} \ddot{q}_i^2. \end{aligned} \quad (3.86)$$

PART I - COMPLIANT MANIPULATORS

For prismatic joints, the formulas are analogous. Notice that, in the case of link-side friction, it is more difficult to compute friction effects starting from friction-free values of $\boldsymbol{\tau}$. For linear stiffness, this can still be done relatively easily by adding $\mathbf{B}\boldsymbol{\Sigma}^{-1}[(\mathbf{D}_l + \frac{\partial \tilde{\mathbf{F}}}{\partial \dot{\mathbf{q}}})\ddot{\mathbf{q}} + \frac{\partial^2 \tilde{\mathbf{F}}}{\partial \dot{\mathbf{q}}^2}\dot{\mathbf{q}}^2]$; otherwise, it becomes impossible, as it would essentially amount to repeating steps (3.9)–(3.15) for nonlinear stiffness, or (3.58)–(3.76) for VSA. Finally, it should be noticed that, due to non-collocation, link-side friction cannot be cancelled completely.

The inclusion of viscoelastic friction changes the nature of the problem considerably. Its effects can be modeled by modifying (2.8) as

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) + \boldsymbol{\Sigma}(\mathbf{q} - \boldsymbol{\theta}_m) + \mathbf{D}(\dot{\mathbf{q}} - \dot{\boldsymbol{\theta}}_m) = \mathbf{0} \quad (3.87a)$$

$$\mathbf{B}\ddot{\boldsymbol{\theta}}_m + \boldsymbol{\Sigma}(\boldsymbol{\theta}_m - \mathbf{q}) + \mathbf{D}(\dot{\boldsymbol{\theta}}_m - \dot{\mathbf{q}}) = \boldsymbol{\tau} \quad (3.87b)$$

where $\mathbf{D} \in \mathbb{R}^{N \times N}$ is the positive-definite diagonal matrix of spring damping coefficients. The inverse dynamics is profoundly affected from this change. It can be seen that, by differentiating (3.87a) once, the term $\ddot{\boldsymbol{\theta}}_m$ appears immediately, together with $\dot{\boldsymbol{\theta}}_m$. This gives rise to the following differential equation

$$\mathbf{D}\ddot{\boldsymbol{\theta}}_m + \boldsymbol{\Sigma}\dot{\boldsymbol{\theta}}_m = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{M}}(\mathbf{q})\dot{\mathbf{q}} + \dot{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{D}\dot{\mathbf{q}} + \boldsymbol{\Sigma}\dot{\mathbf{q}} \quad (3.88)$$

which needs to be solved for $\dot{\boldsymbol{\theta}}_m(t)$ and its derivative $\ddot{\boldsymbol{\theta}}_m(t)$ with given values of $\mathbf{q}(t)$, $\dot{\mathbf{q}}(t)$, $\ddot{\mathbf{q}}(t)$ and $\ddot{\mathbf{q}}(t)$, starting from an arbitrary value of $\boldsymbol{\theta}_m(0)$. Here, the EJNEA₃^{*} (from Sec. 2.3) may be of help for computing the right-hand side of (3.88) at each time instant. Once the value of $\ddot{\boldsymbol{\theta}}_m$ is obtained for a given time instant, the desired torque is obtained as

$$\boldsymbol{\tau} = \mathbf{B}\ddot{\boldsymbol{\theta}}_m + \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}). \quad (3.89)$$

On the other hand, an exactly feedback linearizing control law [21] is given as

$$\begin{aligned} \boldsymbol{\tau} = & \mathbf{B}\mathbf{D}^{-1}[\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}}_r + \dot{\mathbf{M}}(\mathbf{q})\dot{\mathbf{q}} + \dot{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}}) + \boldsymbol{\Sigma}(\dot{\mathbf{q}} - \dot{\boldsymbol{\theta}}_m)] \\ & + [\mathbf{M}(\mathbf{q}) + \mathbf{B}]\ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) \end{aligned} \quad (3.90)$$

where again EJNEA₃^{*} can be used to compute the terms. However, it should be noticed that such a control algorithm will be difficultly implementable if the values of $\mathbf{D}$ are very low, which is usually the case. Therefore, a dynamic compensator [21] is usually preferable.

Chapter 4

Results

In this Chapter, many numerical simulations are performed in order to test the proposed solutions. Results are shown on EJNEA and VSA-NEA, for both inverse dynamics and control settings. The MATLAB[®] implementation of VSA-NEA and the source code for its pertaining simulations are available¹.

4.1 EJNEA

4.1.1 Validation

The algorithm EJNEA was validated on a simple 2R planar robot moving on a vertical plane, shown in Fig. 4.1.

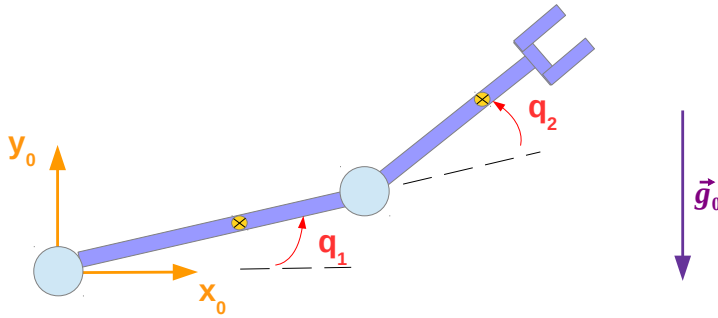


Figure 4.1 Planar 2R robot used for validation

For this robot, the Lagrangian symbolic model was manually computed, and it was employed in order to perform an inverse dynamics computation over a given joint trajectory, using (2.18). The imposed joint trajectory and the resulting link and motor torques are shown in Fig. 4.2.

¹<http://ieeexplore.ieee.org/document/7399670/media>

PART I - COMPLIANT MANIPULATORS

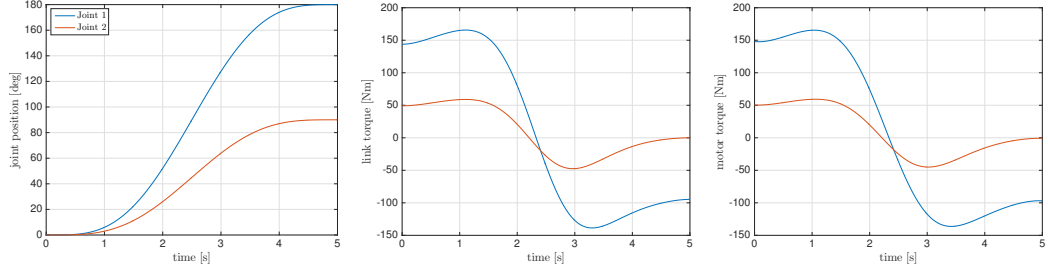


Figure 4.2 Joint and torque trajectory followed during validation

The results were compared with those obtained using EJNEA, both in its standard version and in that with the modified Denavit-Hartenberg convention. Figure 4.3 shows the difference between the motor torques τ obtained with the Lagrangian model and those given by EJNEA. For reference, also the difference in the link torque τ_e is shown.

It can be seen that the difference is practically zero for all cases, thus demonstrating the soundness of the proposed algorithm. In particular, it can be seen that the difference for τ is in the same order of magnitude as that for τ_e ; this is significant since it means that EJNEA is basically as reliable as regular RNEA in its results. Details on the physical parameters of the robot in Fig. 4.1 and of its Lagrangian model are reported in Appendix C.

4.1.2 Inverse dynamics of KUKA LWR4+

In this section, the results and performance of EJNEA over a more complex robot are reported. For the simulation, a realistic model of the 7R KUKA LWR4+ is employed. The kinematic and dynamic parameters used for the numerical evaluations are listed in Tab. 4.1 and 4.2, in standard Denavit-Hartenberg notation. For the dynamic parameters, the values of link masses and inertias and the position of the link CoMs were taken from [32], while the values for the drive inertia moments were taken from [49], and the joint stiffnesses have been chosen arbitrarily.

The desired link motion was specified by sinusoidal trajectories with different amplitudes and frequencies for each joint, lasting $T = 14.8$ s and having peak absolute velocities between $90^\circ/\text{s}$ and $200^\circ/\text{s}$. The obtained numerical results are shown in Figs. 4.4 to 4.6. During motion, the elastic joint deflections are as large as 3° for the second joint (Fig. 4.5). The maximum differences between the torques that would be needed in the rigid case and those needed in the elastic case range between 2% and 5% of the peak torques for the robot with elastic joints (Fig. 4.6), especially for the first 4 joints.

The algorithm was implemented in MATLAB, automatically converted to C code, and then run as a `mex` function on a standard personal computer.

CHAPTER 4. RESULTS

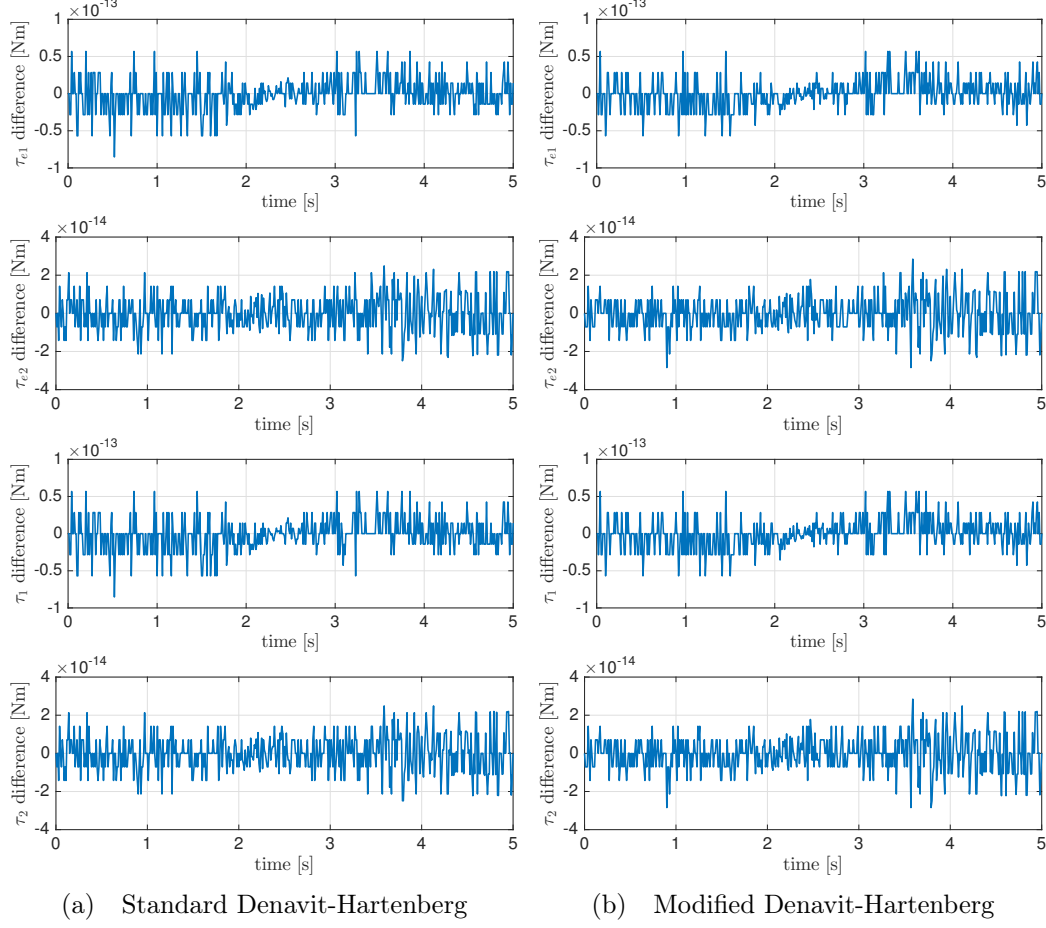


Figure 4.3 Difference between the inverse dynamic values obtained with the Lagrangian model and those given by EJNEA for the 2R robot, with different kinematic conventions.

		1	2	3	4	5	6	7
α_i	rad	$\pi/2$	$-\pi/2$	$-\pi/2$	$\pi/2$	$\pi/2$	$-\pi/2$	0
a_i	m	0	0	0	0	0	0	0
d_i	m	0	0	0.40	0	0.39	0	0
θ_i	rad	q_1	q_2	q_3	q_4	q_5	q_6	q_7
$ q _{\max i}$	deg	170	120	170	120	170	120	170
$ \dot{q} _{\max i}$	deg/s	110	110	128	128	204	184	184
$ \tau _{\max i}$	Nm	176	176	100	100	100	38	38



TABLE 4.1 The robot KUKA LWR4+ and its kinematic parameters.

PART I - COMPLIANT MANIPULATORS

p	value	p	value	p	value	p	value
B_1	3.20	c_{1x}	-0.0216387515	I_{1xx}	—	I_{4xz}	-0.0005187982
B_2	3.05	c_{1y}	—	I_{1yy}	0.0018932828	I_{4xy}	0.000000225
B_3	1.98	c_{1z}	-0.0376881829	I_{1zz}	—	I_{4yz}	-0.0005484476
B_4	2.06	c_{2x}	0.0003284751	I_{1xy}	—	I_{5xx}	0.006322648
B_5	0.801	c_{2y}	-0.0041132249	I_{1xz}	—	I_{5yy}	0.0012020203
B_6	0.48	c_{2z}	0.0823647642	I_{1yz}	—	I_{5zz}	0.0070806218
B_7	0.381	c_{3x}	0.0002593328	I_{2xx}	0.0474108647	I_{5xy}	-0.0002163196
K_1	1000	c_{3y}	0.1137431845	I_{2yy}	0.05	I_{5xz}	0.00000652
K_2	1000	c_{3z}	-0.000100257	I_{2zz}	0.001601901	I_{5yz}	-0.005
K_3	1000	c_{4x}	-0.0014648843	I_{2xy}	-0.00000621	I_{6xx}	0.0005278646
K_4	1000	c_{4y}	-0.0000461	I_{2xz}	0.0001166457	I_{6yy}	0
K_5	1000	c_{4z}	0.148580959	I_{2yz}	-0.0009141575	I_{6zz}	0.0034899625
K_6	1000	c_{5x}	-0.0003791484	I_{3xx}	0.0469510749	I_{6xy}	0.0000483
K_7	1000	c_{5y}	-0.0553526131	I_{3yy}	0.0008344566	I_{6xz}	-0.0000375
m_1	4.1948152162	c_{5z}	-0.0101255137	I_{3zz}	0.05	I_{6yz}	-0.0010605344
m_2	4.2996847737	c_{6x}	0.0020739022	I_{3xy}	-0.000271431	I_{7xx}	0
m_3	3.658530333	c_{6y}	0.0586184696	I_{3xz}	4.09E-008	I_{7yy}	0.0000323
m_4	2.3846673548	c_{6z}	-0.044799983	I_{3yz}	-0.000577228	I_{7zz}	0.0001187527
m_5	1.7035567183	c_{7x}	-0.0004601303	I_{4xx}	0.0124233226	I_{7xy}	-0.000000577
m_6	0.4000713156	c_{7y}	0.0014789221	I_{4yy}	0.0072708907	I_{7xz}	-0.000021407
m_7	0.6501439811	c_{7z}	0.0715608282	I_{4zz}	0.0099884782	I_{7yz}	0.000068807

TABLE 4.2 Dynamic parameters of KUKA LWR4+. All values are expressed in SI standard units. The values indicated with a dash do not influence the dynamics and can be set to zero.

The same simulation was run 100 times to measure the speed of the algorithm in this setting. Remarkably, 100 simulations of 14.8 s of inverse dynamics computations, with a sampling time $\Delta t = 0.01$ s, for a total of 148100 iterations, took only 1.3353 s of execution time, achieving an average of about 9 μ s per iteration and thus proving that the method is perfectly feasible for real-time control use.

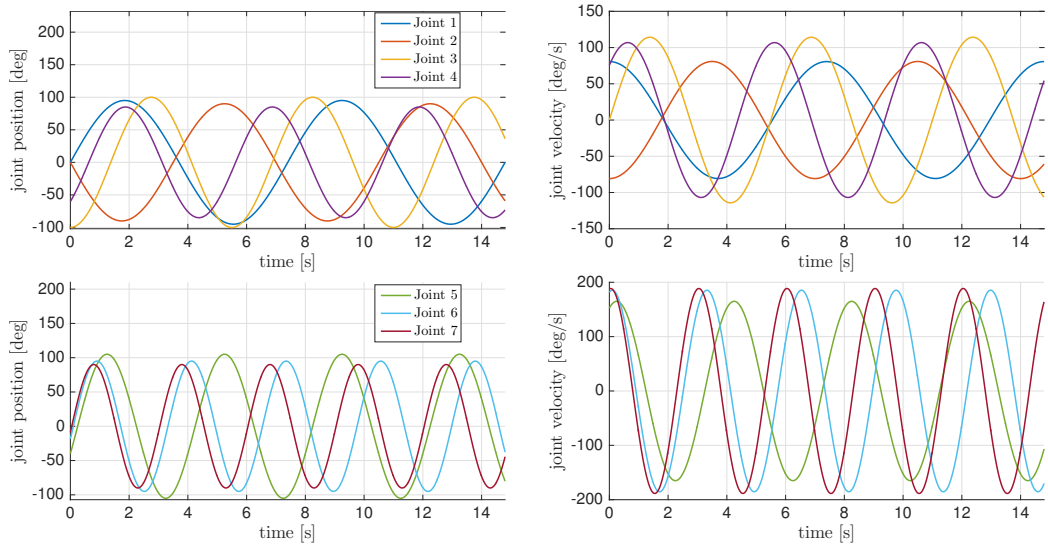


Figure 4.4 Joint positions $\mathbf{q}_d$ and velocities $\dot{\mathbf{q}}_d$ for the KUKA LWR4+.

CHAPTER 4. RESULTS

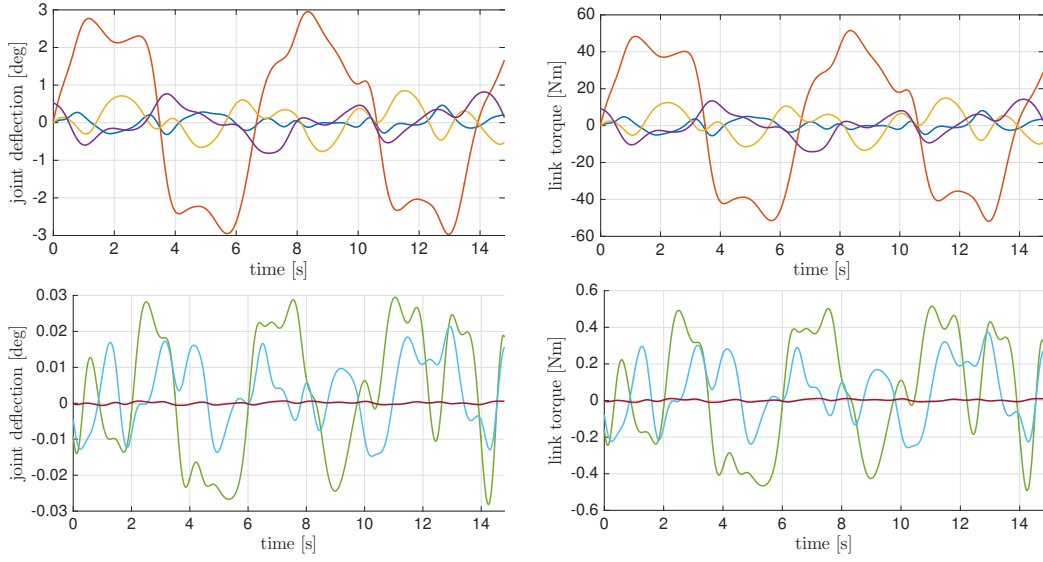


Figure 4.5 Joint deflections $\phi_d = \theta_{md} - q_d$ and elastic torques τ_{ed} for the KUKA LWR4+.

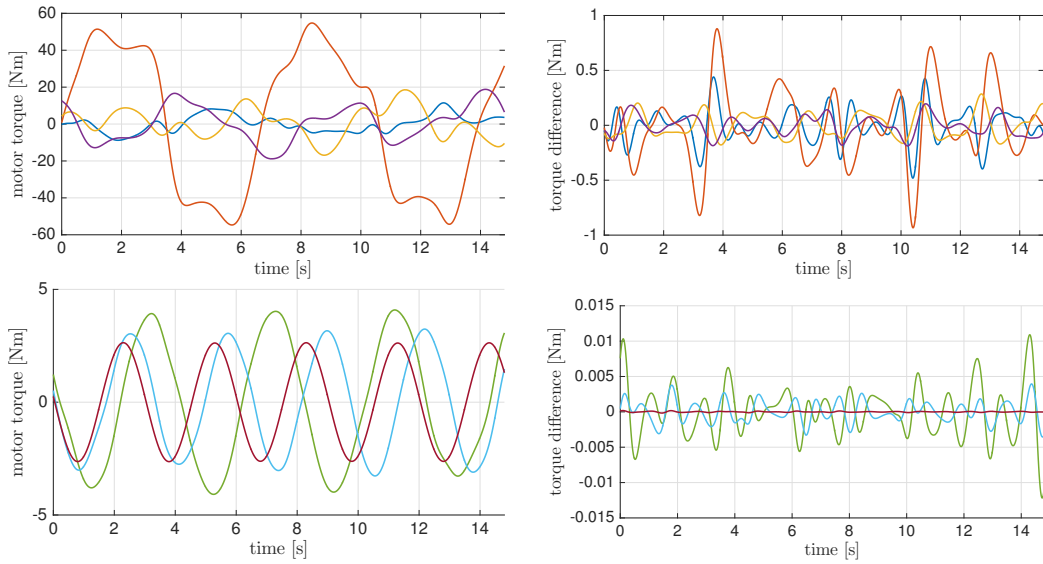


Figure 4.6 Motor torque τ_d and its difference with the desired motor torque $\tau_{r,d}$ in the rigid case, for the KUKA LWR4+.

4.2 VSA-NEA

4.2.1 Inverse dynamics

In order to validate the VSA-NEA, we consider a spatial 3-DoF robot in the presence of gravity (acting on links 2 and 3), with the three rotational joints driven by antagonistic VSA with cubic profiles of flexibility torques. For this simulation, viscous friction was included both at the link side and at the motor side. The robot and its physical parameters are shown in Tab. 4.3.

		1	2	3
α_i	rad	$\pi/2$	0	0
a_i	m	0	0.3	0.3
d_i	m	0	0	0
θ_i	rad	q_1	q_2	q_3
m_i	kg	9.0478	6.7858	5.0894
${}^i\mathbf{p}_{cx_i,i}$	m	0	-0.15	-0.15
${}^i\mathbf{p}_{cy_i,i}$	m	0	0	0
${}^i\mathbf{p}_{cz_i,i}$	m	0	0	0
${}^iI_{xx_i}$	kg m ²	0.1299	0.0139	0.0104
${}^iI_{yy_i}$	kg m ²	0.0185	0.0578	0.0434
${}^iI_{zz_i}$	kg m ²	0.1299	0.0578	0.0434
D_{li}	kg m ² /s	10^{-5}	10^{-5}	10^{-5}
$B_{k,i}$	kg m ²	3.20	3.05	1.98
$D_{mk,i}$	kg m ² /s	10^{-4}	10^{-4}	10^{-4}
K_i	Nm/rad	400	400	400
K_{ci}	Nm/rad ³	2000	2000	2000

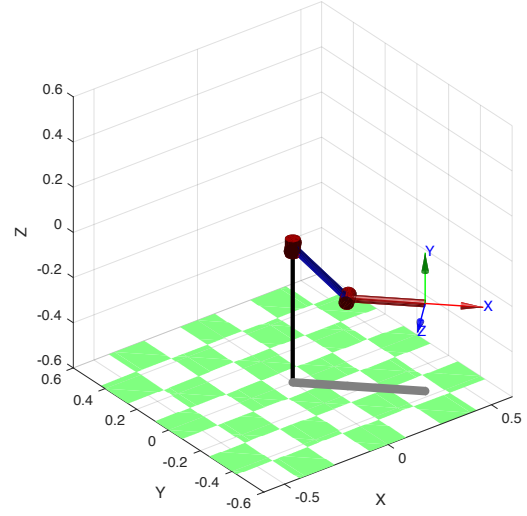


TABLE 4.3 The 3R VSA robot and its kinematic and dynamic parameters.

The link motion $\mathbf{q}_d$ was specified by a smooth rest-to-rest trajectory (polynomials of degree 7, with zero initial and final boundary conditions on velocity, acceleration, and jerk) lasting $T = 4$ s. In a first test, $\boldsymbol{\sigma}_d$ was also specified by a rest-to-rest polynomial trajectory of degree 3, identical for all joints, with zero boundary conditions on $\dot{\boldsymbol{\sigma}}$ at $t = 0$ and $t = T$. The stiffness of each joint goes from $\sigma_{min} = 850$ Nm/rad, which is close to the physical minimum possible stiffness, to $\sigma_{max} = 1275$ Nm/rad, achieving a 50% increase. Figure 4.7 shows the reference trajectories, while the obtained results are reported in Figs. 4.8–4.9.

We performed two other tests with the same joint trajectory $\mathbf{q}_d$ shown in Fig. 4.7, but one with σ_i identically equal to σ_{min} for all i , and a second with σ_i identically equal to σ_{max} . Since the joint movement is the same, the elastic torque $\boldsymbol{\tau}_e$ does not change with respect to Fig. 4.9. however, all the motor torques $\boldsymbol{\tau}_{d1}$ and $\boldsymbol{\tau}_{d2}$ have much larger absolute values in the second case. This shows how the motors spend considerable energy in order to keep a higher level of stiffness. In Fig. 4.10, we compare the desired nominal torques for the

CHAPTER 4. RESULTS

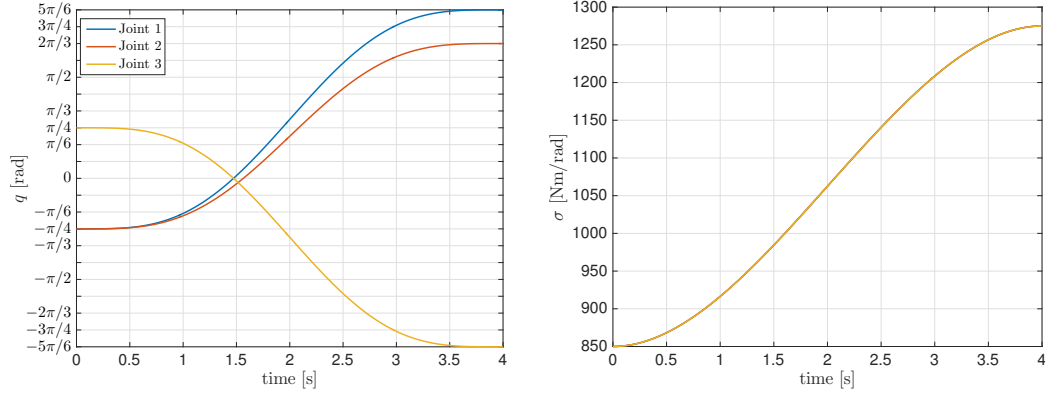


Figure 4.7 Desired joint position $\mathbf{q}_d$ and stiffness σ_d for the VSA robot (same stiffness profile for all joints).

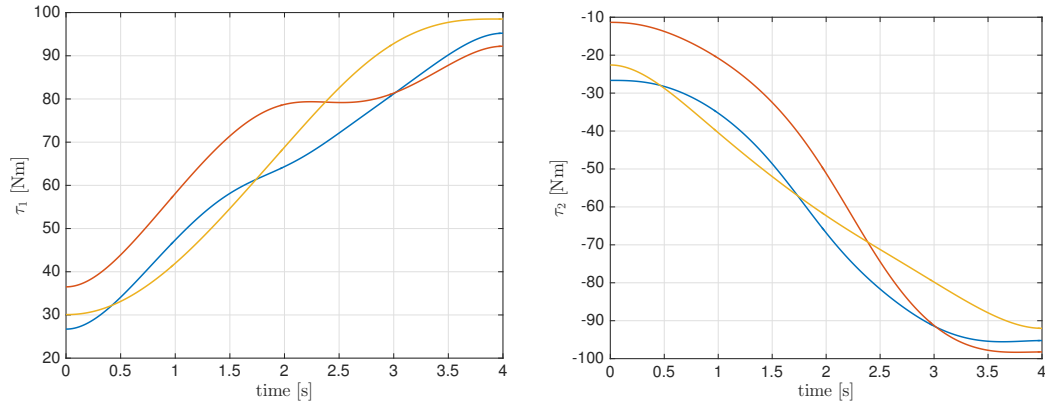


Figure 4.8 Nominal motor torques τ_{d1} and τ_{d2} for the VSA robot.

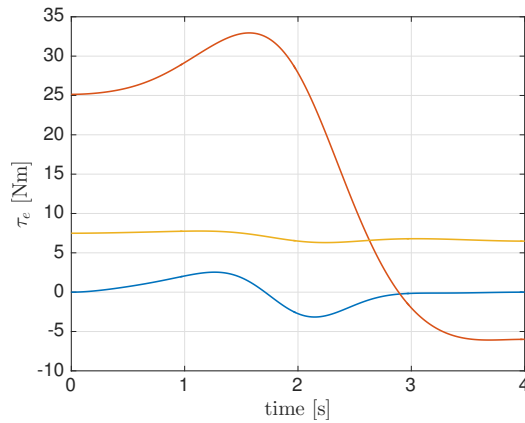


Figure 4.9 Total elastic torque $\tau_e = \tau_{e1} + \tau_{e2}$ for the VSA robot.

PART I - COMPLIANT MANIPULATORS

two cases. It can be seen how the initial values of τ_{d1} and τ_{d2} for $\sigma_{di} = \sigma_{min}$ ($i = 1, \dots, 7$) coincide with the initial values in Fig. 4.8, while their final values for $\sigma_{di} = \sigma_{max}$ coincide the final values in the same figures.

The algorithm was implemented in MATLAB, automatically converted to C code, and then run as a `mex` function on a standard personal computer. The execution times were always in the order of $10^{-5} \div 10^{-4}$ s per iteration.

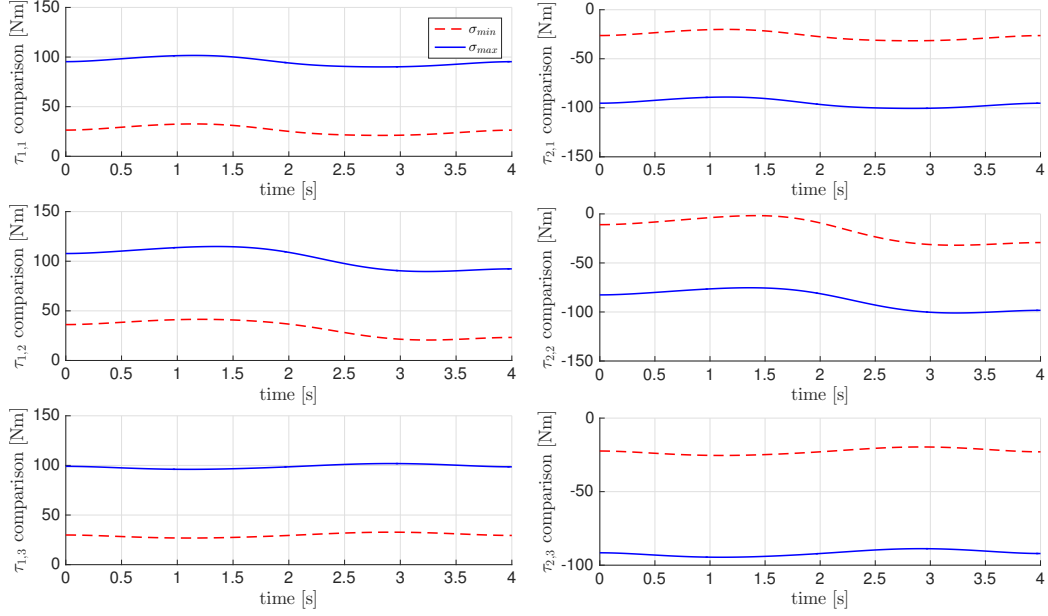


Figure 4.10 Comparison of the nominal motor torques exerted by the VSA robot for $\sigma_{di} = \sigma_{min} = 850$ Nm/rad and $\sigma_{di} = \sigma_{max} = 1275$ Nm/rad for all joints.

4.2.2 Control

In order to demonstrate the performance of the proposed algorithm in a control setting, the dynamic behavior of the robot in the previous section was simulated together with the feedback linearization controller in Sec. (3.2.4). The linearized control signal includes an integral action, with all eigenvalues being set to -30 . The same joint-stiffness trajectory (4.7) was imposed as a desired trajectory to the controller. In order to make the simulations more realistic, a small random error was added to both the initial values of $\mathbf{q}$ and $\dot{\mathbf{q}}$ and to the dynamic parameters employed by the controller. Further, the motor torques were saturated at $|\tau|_{\max k,i} = 150$ [Nm]. The simulation is made last 1 [s] more than the desired trajectory, in order to show the controller's behavior in static conditions. The final output trajectory is shown in Fig. 4.11, with the tracking errors given in Fig. (4.12) and the control torques in Fig. (4.13).

CHAPTER 4. RESULTS

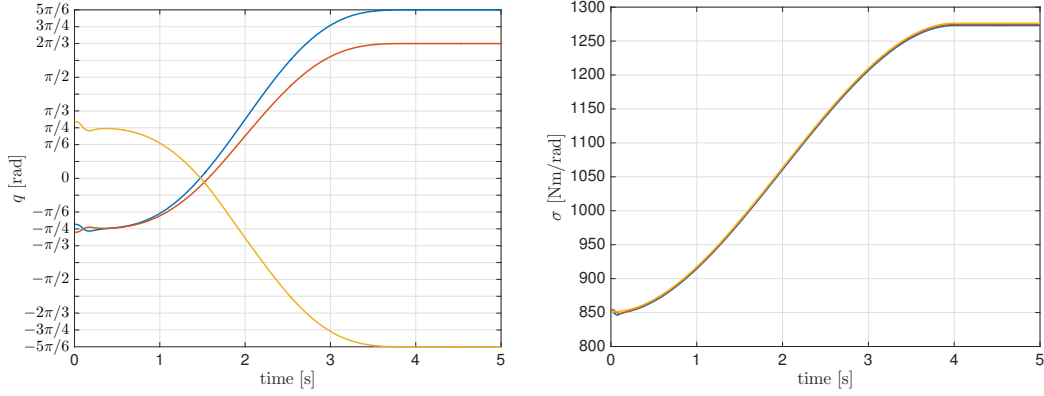


Figure 4.11 Actual joint position q and stiffness σ for the controlled VSA robot.

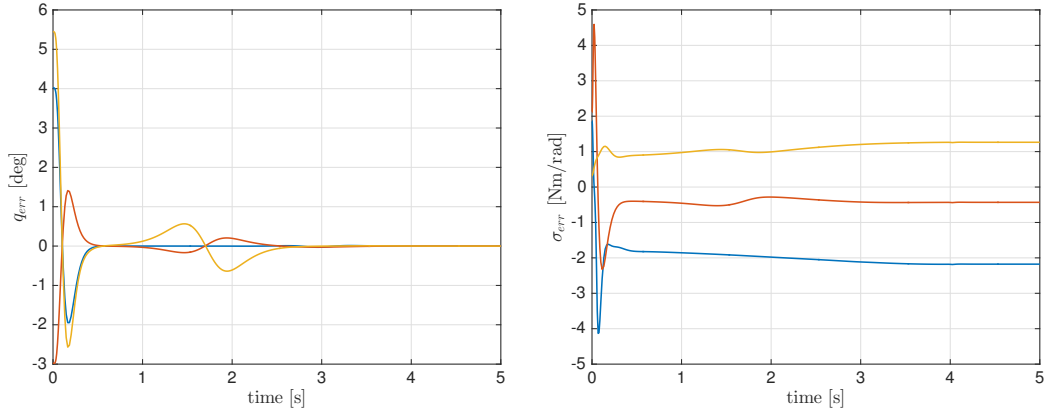


Figure 4.12 Joint position error q_{err} and stiffness error σ_{err} for the controlled VSA robot.

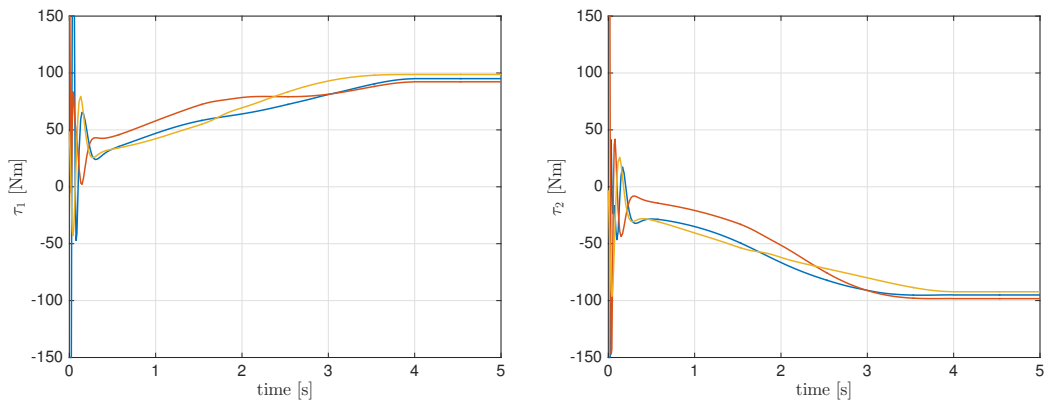


Figure 4.13 Actual motor torques τ_1 and τ_2 for the controlled VSA robot.

It can be seen that the initial joint position error is quickly recovered, while the stiffness error never exceeds 5 [Nm/rad] in absolute value. The

PART I - COMPLIANT MANIPULATORS

tracking error is never brought completely to zero, due to the modeling errors. In particular, this is true for the stiffness σ , which is not measured directly by the controller, but internally computed from the deflections ϕ_1 and ϕ_2 using the incorrect parameters; this explains why the system reaches a stationary input value with non-zero stiffness error, despite the presence of the integrator. Also, it can be seen that the motor torques are subject to strong oscillations in the initial phase and often saturate. Such a behavior may not be actually viable in a real robot; however, this problem is almost immediately overcome and the torques rapidly become very smooth.

Part II

OPTIMIZATION-BASED ACTUATION DESIGN FOR COMPLIANT HUMANOIDS

PART II - COMPLIANT HUMANOIDS

Chapter 5

Introduction

An optimization framework for the design and analysis of underactuated biped walkers is presented [10]. The robots can be characterized by passive or actuated joints with rigid or non-negligible elastic actuation/transmission elements. The framework is based on optimal control, dealing with geometric constraints and various dynamic objective functions, as well as boundary conditions, which helps in selecting optimal values both for the actuation and the transmission parameters. Solutions of the formulated problems are shown for different kinds of bipedal architectures, and comparisons drawn between traditional rigid robots and compliant ones show the energy-efficiency of compliant actuators in the context of locomotion.

5.1 Contribution

Human beings are energetically very efficient in the task of walking, when compared to current robots. An important characteristic of human walking patterns is that they exploit the human body's natural passive dynamics. The passive dynamics is the body dynamics when no actuation forces are present. In this case, the body is only subject to gravity, external forces, and to elastic or friction forces at joint level. The human body structure allows the main characteristics of dynamic walking to emerge naturally, thus requiring much lower energy and control frequency. Another characteristic of the human body is the intrinsic compliance of its muscle structure. The possibility to use it as a temporary energy storage is also an example of exploitation of the passive dynamics.

As seen in the Introduction and in Part I, the idea of introducing compliance was a direct inspiration for elastic actuators. The human passive walking dynamics is trying to be imitated by *passive walkers*, of which many



Figure 5.1 The passive humanoid robot Denise. Image edited from [88].

different prototypes exist today [89, 46]. In principle, this specific type of biped robots is able to maintain stability without any action from the motors, which can be entirely absent, and is able to walk along an inclined plane pushed by gravity only. Clearly, a real robot, in order to serve any practical purpose, cannot be entirely passive, but will need to possess some kind of actuation; thus, the guiding principle will be that of optimizing the design so that the actuator effort during *nominal* walking be minimal.

In view of the complexity of compliant passive robot design, there is a need to develop a software capable to automatize and speed up at least part of the design process, in particular the study of the different types of structures and actuators, the selection of the dynamic parameters of the robot itself, and the analysis of the relationship between these two choices. This is the basic idea of *co-design*. A first step in this direction was taken in [66]. In that work, the authors introduced a framework for simultaneous design and control of passivity-based walkers, and they applied it to different kinds of kinematic structures to draw quantitative comparisons between them. The framework presented there is based on mathematical optimization: the design problem is formulated as an optimal control problem (OCP), which is capable to simulate the robot behavior while walking and potentially optimize the state trajectory, the input trajectory and the robot parameters together in order to minimize a given cost function, subject to various path constraints. However, only the robot trajectory and some related parameters are optimized, while the dynamical parameters of the robot itself are left untouched.

In this work, the framework is extended to deal with more complex use cases. In particular, it is developed for humanoid robots equipped with elastic actuators. The actuator parameters are selected by the framework so to minimize the cost function. The simultaneous selection of the joint trajectory and the actuator parameters, by means of optimal control, allows to fully exploit the potential of elastic actuators. Among previous works featuring compliant humanoids simulation, [18, 55, 74, 75] are noted. Simulations results are presented, both on a simple model and on a more complex whole-body model, employing both rigid and elastic actuation, drawing comparisons between the different cases.

Different contributions include

- The presentation of an optimization framework allowing to design and simulate robots equipped with different kinds of elastic actuators. This is based on the experience of [66].
- The framework is shown to be capable of simulating whole-body humanoid robots equipped with elastic actuators.
- Different performance criteria are implemented. Simulation results include cost of transport (CoT) and torque minimization.

PART II - COMPLIANT HUMANOIDS

Chapter 6

Framework

6.1 Legged robots

Legged robots are characterized by the presence of a *continuous dynamics*, which is analogous to that of manipulators, and of an *impulsive dynamics*, which takes place when the robot sets its foot on the ground as a consequence of the resulting impact. For bipeds, the continuous dynamics is divided into single-support phase, when only one foot touches the ground, and double-support phase, when both feet do. During the single-support phase, the foot touching the ground is called stance foot, while the one that doesn't is called swing foot.

As for the continuous dynamics, there are three fundamental differences between a manipulator dynamics and that of a legged robot. The first one is that walkers are intrinsically *underactuated* systems, as they can only displace themselves by changing their configuration while making contact with the environment. This causes the number of motors m to be less than the robot degrees of freedom N . The contact also imposes some *constraints* on the robot motion. Contact with the ground can be modeled as either compliant or rigid. However, in the compliant case, the transient phase of the contact model is usually very short; further, the resulting dynamical system can be very stiff and difficult

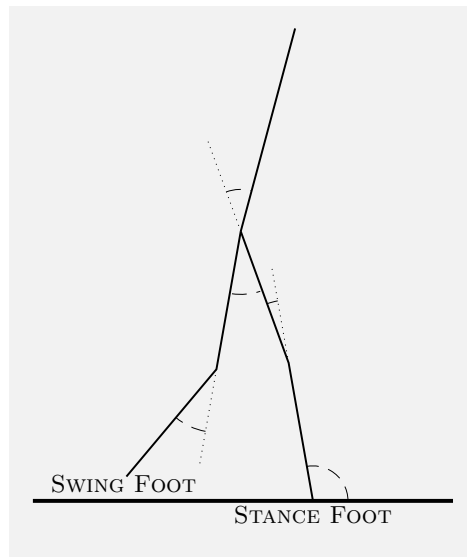


Figure 6.1 Schematic representation of a bipedal walker.

PART II - COMPLIANT HUMANOIDS

to analyze, and a realistic dynamic model is often difficult to obtain. For this reason, in this work, contact is always assumed to be rigid, as it is often done [87]. The second difference in the dynamics of manipulators and legged robots is that, in the latter case, part of the robot configuration is not represented by joint positions, but by the pose of a body in space. This is best represented with the Special Euclidean space $SE(3)$. Accordingly, its evolution is not represented through simple time derivatives of the configuration, but through their *tangent space*. For rigid robots, the full continuous dynamic model is given as

$$\mathbf{M}(\mathbf{q})\dot{\mathbf{v}} + \mathbf{C}(\mathbf{q}, \mathbf{v})\mathbf{v} + \mathbf{g}(\mathbf{q}) = \mathbf{S}^T \boldsymbol{\tau} + \mathbf{J}^T(\mathbf{q})\mathbf{F} \quad (6.1a)$$

$$\mathbf{J}(\mathbf{q})\dot{\mathbf{v}} + \dot{\mathbf{J}}(\mathbf{q})\mathbf{v} = \mathbf{0} \quad (6.1b)$$

where (6.1a) describes the dynamic evolution of the system and (6.1b) the nonholonomic dynamic constraints. Here, $\mathbf{J} \in \mathbb{R}^{k \times N}$ is the *constraint Jacobian* and $\mathbf{F} \in \mathbb{R}^k$ corresponds to the *constraint forces*, where k is the number of dynamic constraints. Also $\mathbf{q} \in SE(3) \times \mathbb{R}^m$ is the robot configuration and $\mathbf{v} \in \mathbb{R}^N$ is the robot configuration tangent space, while $\boldsymbol{\tau} \in \mathbb{R}^m$ is the vector of motor torques and $\mathbf{S} \in \mathbb{R}^{m \times N}$ is a selection matrix encoding the underactuation, where $N = 6 + m$ (or more generally $m < N$). The inverse dynamics is then expressed as

$$\mathbf{F} = -\boldsymbol{\Lambda}^{-1} \left(\mathbf{J}\mathbf{M}^{-1}(\mathbf{S}^T \boldsymbol{\tau} - \mathbf{n}) + \dot{\mathbf{J}}\mathbf{v} \right) \quad (6.2a)$$

$$\dot{\mathbf{v}} = \mathbf{M}^{-1}(\mathbf{S}^T \boldsymbol{\tau} - \mathbf{n} + \mathbf{J}^T \mathbf{F}) \quad (6.2b)$$

where dependencies have been omitted and $\boldsymbol{\Lambda}$ is the so-called operational-space inverse inertia matrix, given as

$$\boldsymbol{\Lambda}(\mathbf{q}) = \mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T. \quad (6.3)$$

The impulse dynamics causes an instantaneous change of velocity. Defining $\mathbf{v}^-$ as the robot configuration tangent velocity before the impact and $\mathbf{v}^+$ as the one after, the impulsive dynamics is usually modeled as

$$\begin{aligned} \mathbf{v}^+ &= \arg \min \frac{1}{2} \|\mathbf{v}^+ - \mathbf{v}^-\|_{\mathbf{M}(\mathbf{q})}^2 \\ &\text{s.t.} \end{aligned} \quad (6.4)$$

$$\mathbf{J}\mathbf{v}^+ = -\varepsilon \mathbf{J}\mathbf{v}^-$$

where $\|\cdot\|_{\mathbf{M}(\mathbf{q})}$ is the weighted norm with respect to $\mathbf{M}(\mathbf{q})$ and ε is the restitution coefficient. The resulting transition is

$$\boldsymbol{\lambda} = (1 + \varepsilon)\boldsymbol{\Lambda}^{-1}\mathbf{J}\mathbf{v}^- \quad (6.5a)$$

$$\mathbf{v}^+ = \mathbf{v}^- - \mathbf{M}^{-1}\mathbf{J}^T\boldsymbol{\lambda} \quad (6.5b)$$



Figure 6.2 The iCub, seen here standing on a single foot, can be modeled as a fixed-base kinematic tree. Still frame from video associated to [59].

where λ is the impulse resulting from the impact [9]. Other impact models (e.g. elastic) could also be introduced without loss of generality. This impact model is frequently used in the literature [69], even though it is contested for its physical consistency [15].

6.2 Dynamic modeling

In this work, only the single-support phase is considered. The double-support phase is seen as instantaneous and can therefore be skipped. Another very common simplifying assumption is that feet can be reduced to points [87]. Since the point-footed model is simpler than a more complete representation, the design, analysis and control of walking robots results facilitated. Therefore, in this work, punctual rigid contact between the robot stance foot and the ground is assumed. Under these working hypotheses, the continuous dynamics of a robotic walker is adequately modeled as that of a fixed-base kinematic tree, provided non-slippage constraints are respected. See Fig. 6.1 for a schematic illustration.

In the model, the robot stance ankle will be replaced by a single unactuated joint directly connected to the ground; this will preserve the inherent underactuation of a biped robot. If only the sagittal dynamics is concerned, this will be a revolute joint whose axis is perpendicular to the sagittal plane;

PART II - COMPLIANT HUMANOIDS

if the lateral dynamics is also taken into account, it will be a 3-DoF (3 Degrees of Freedom) spherical joint. Impact dynamics will be left out for the purpose of this paper.

In light of the above simplifications, the Lagrangian model of rigid robots becomes again more familiar

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \mathbf{S}^T \boldsymbol{\tau} \quad (6.6)$$

where again $\mathbf{q} \in \mathbb{R}^N$, $\boldsymbol{\tau} \in \mathbb{R}^m$ and $\mathbf{S} \in \mathbb{R}^{m \times N}$. The dynamic state of the robot is given by the link variables and their time derivatives, as in the manipulator case, and it is denoted as

$$\mathbf{x} = (\mathbf{q}^T \ \dot{\mathbf{q}}^T)^T \in \mathbb{R}^n, \quad n = 2N. \quad (6.7)$$

For series elastic actuators, employing the reduced model

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) = \mathbf{S}^T \boldsymbol{\Sigma}(\boldsymbol{\theta}_m - \mathbf{S}\mathbf{q}) \quad (6.8a)$$

$$\mathbf{B}\ddot{\boldsymbol{\theta}}_m + \boldsymbol{\Sigma}(\boldsymbol{\theta}_m - \mathbf{S}\mathbf{q}) = \boldsymbol{\tau} \quad (6.8b)$$

where the joint deflections and the dynamic state are defined as

$$\boldsymbol{\phi} = \boldsymbol{\theta}_m - \mathbf{S}\mathbf{q} \quad (6.9)$$

$$\mathbf{x} = (\mathbf{q}^T \ \dot{\mathbf{q}}^T \ \boldsymbol{\theta}_m^T \ \dot{\boldsymbol{\theta}}_m^T)^T \in \mathbb{R}^n, \quad n = 2(N + m). \quad (6.10)$$

In this we also test parallel elastic actuators. In this actuation scheme, motors directly drive the respective links through rigid transmission elements, but in addition the link is also connected to a fixed point through a spring. A schematic comparison of the series and parallel actuation model is given in Fig. 6.3. For parallel elastic actuators, the Lagrangian model is

$$\begin{aligned} \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) &= \mathbf{S}^T (\boldsymbol{\tau} + \boldsymbol{\Sigma}(\bar{\mathbf{q}} - \mathbf{q})) \\ \mathbf{x} &= (\mathbf{q}^T \ \dot{\mathbf{q}}^T)^T \in \mathbb{R}^n, \quad n = 2N \end{aligned} \quad (6.11)$$

where $\bar{\mathbf{q}}$ is the *constant* equilibrium point of the springs, and it can be seen that $\mathbf{x}$ is as in (6.7). In order not to compromise the mobility of the robot, the joint stiffness will be typically very low.

6.3 The framework

In this section, we setup the co-design problem using the tools of optimal control. The act of walking is studied in its steady state, i.e. the robot

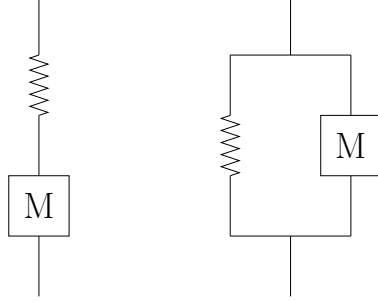


Figure 6.3 Comparison of the series and parallel actuation schemes.

does not start from zero velocity but instead is considered to have already performed a full step which is identical to the computed one. Therefore, the control problem can be designed over a single step, ensuring that the final solution can be repeated periodically with joint-space continuity; in particular, the initial and final configuration of the robot need to be symmetrical with respect to the sagittal plane.

In general, the goal of the optimal control problem is to minimize the integral of a cost function $c(\mathbf{x}, \boldsymbol{\tau})$ over the whole step duration, depending on the state trajectory, the control trajectory, and some parameters $\mathbf{p}$, subject to some constraints. These constraints are divided into *decoupled constraints* $\mathbf{f}_d$, which have to be valid at each trajectory point, and *coupled constraints* $\mathbf{f}_c$, which relate the initial and final trajectory point to ensure periodicity of the solution and other high-level trajectory properties.

6.3.1 Rigid robots

For rigid robots, the problem is set up as

$$\min_{\underline{\mathbf{x}}, \underline{\boldsymbol{\tau}}, \mathbf{p}} \int_0^T c(\mathbf{x}, \boldsymbol{\tau}, \mathbf{p}) dt \quad (6.12a)$$

s.t

$$\ddot{\mathbf{q}} = \mathbf{M}^{-1}(\mathbf{q})(\mathbf{S}^T \boldsymbol{\tau} - \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{g}(\mathbf{q})) \quad (6.12b)$$

$$\mathbf{f}_d(\mathbf{x}, \boldsymbol{\tau}) \geq 0 \quad (6.12c)$$

$$\mathbf{f}_c(\mathbf{x}(0), \boldsymbol{\tau}(0), \mathbf{x}(T), \boldsymbol{\tau}(T), \mathbf{p}) = 0 \quad (6.12d)$$

with $\underline{\mathbf{x}}$ and $\underline{\boldsymbol{\tau}}$ denoting the state and control trajectories respectively. The parameters $\mathbf{p}$ mostly include trajectory and environment properties, e.g. the step length and the inclination of the slope along which the robot walks.

The decoupled constraints $\mathbf{f}_d$ include bounds on all variables. Also, they ensure that the swing foot touches the ground both at the beginning and at

PART II - COMPLIANT HUMANOIDS

the end of the trajectory, by setting its altitude with respect to the slope to zero.

The coupled constraints $\mathbf{f}_c$ enforce mostly consist in forcing each joint's position to be specular that of its counterpart on the other side of the robot, e.g. the right hip position at the beginning of the trajectory needs to be equal to the left hip position at the end, and *vice versa*. Instead, the ankle joint initial configuration needs to be symmetrical to the final one with respect to the sagittal plane. The identical consideration needs to be done for link velocities. Also, the coupled constraints are used to ensure that the step length is equal to the value contained in $\mathbf{p}$. This is necessary if it is desired to set the step length to a fixed pre-defined value, or to constrain it to respect certain bounds.

The choice of the “best” cost function is still an open problem in robotics. In this paper, we explore two different alternatives.

The first possibility is the so-called *cost of transport* (CoT). This is equal to the ratio between the energy consumed by the system and the weight multiplied by the travel distance. The consumed energy is equal to the difference in potential energy plus the integral of the power input. If the slope is fixed, this is equivalent to minimizing

$$c(\mathbf{x}, \boldsymbol{\tau}) = c_{\text{CoT}} = \frac{|\boldsymbol{\tau}|^T \mathbf{S} |\dot{\mathbf{q}}|}{L} \quad (6.13)$$

where L is the step length, and $|\cdot|$ is the component-wise absolute value. Here, we are assuming energy *cannot* be recovered, which is why we included the absolute values. This may resemble human dynamics more closely. Notice that, on some modern hardware, such as the Cheetah robot [70], some energy can be recovered from braking. However, this is always at a huge loss, most of the energy going to be lost anyways. This was the cost function of choice in [66], as well as in classical works such as [68].

Another possibility is the *squared norm of the torque*, normalized by the step length:

$$c(\mathbf{x}, \boldsymbol{\tau}) = c_{\boldsymbol{\tau}} = \frac{\|\boldsymbol{\tau}\|^2}{L}. \quad (6.14)$$

This choice of function is useful to keep the torques exerted by the motors limited, and may in general lead to better-conditioned problems. It has already been observed that penalizing the squared norm of the torque variable instead of the mechanical work produces behaviors which tend to appear more natural [69].

6.3.2 Series elastic actuators

The optimal control problem for SEAs is set up similarly, taking into account the different dynamics (6.8). Care should be taken in ensuring the periodicity of $\boldsymbol{\theta}_m$ and $\dot{\boldsymbol{\theta}}_m$ too. Additionally, we optimize the parameters for $\boldsymbol{\Sigma}$ and $\mathbf{B}$. It should be noticed that we will want the parameters of some joints to coincide with each other. For instance, the joints at both hips should be characterized by the same stiffness and drive inertia moment, and the same can be said for possible knees, shoulders and elbows. Rather than enforcing this as a constraint, we will use a single parameter for the stiffness of each couple of symmetric joints, and one for the drive inertia moment, thus reducing the dimensionality of the problem and enhancing precision.

The mathematical expression of the CoT is slightly changed, becoming

$$c_{\text{CoT}} = \frac{|\boldsymbol{\tau}|^T |\dot{\boldsymbol{\theta}}_m|}{L}. \quad (6.15)$$

Notice that, since it is assumed that the final pose is exactly specular to the initial one, no contribution to the cost function is assumed from the potential elastic energy. Also, it should be ensured that the spring deflections do not exceed certain limits:

$$|\boldsymbol{\theta}_m - \mathbf{S}\mathbf{q}| < \phi_{\max}. \quad (6.16)$$

6.3.3 Parallel elastic actuators

For parallel elastic actuators, we optimize $\boldsymbol{\Sigma}$ and $\bar{\mathbf{q}}$, still ensuring symmetry of the parameters. In order not to constrain the mobility of the robot, we do not put bounds on the deflections, relying on the low stiffness. The CoT is still expressed as in (6.13).

6.3.4 Optimization techniques

In order to solve optimal control problems, two main techniques exist. The first one consists in *indirect methods*, exploiting the necessary conditions for optimality. However, such methods are currently unable to track path constraints. The second technique consists in *direct methods*. Direct methods first discretize the original problem into a finite dimensional nonlinear programming problem, which is then solved. Among direct methods, three main strategies are usually employed, listed below. Further details can be found in [26].

Single shooting The control and constraints are discretized according to a temporal grid. The state trajectory is recovered by integration of

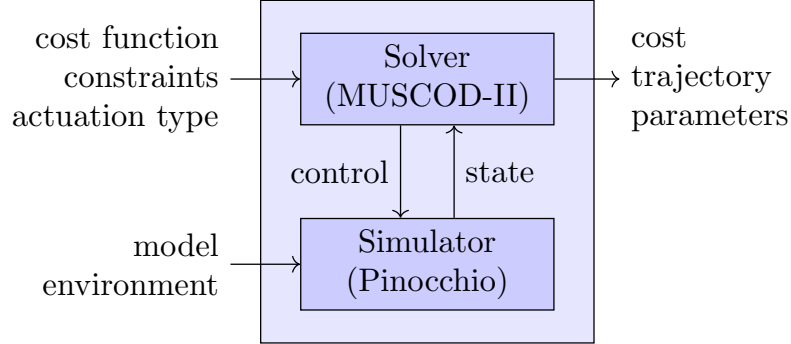


Figure 6.4 A schematic representation of the framework.

the discrete control trajectory along this grid. As only the control trajectory is searched, the optimization problem is of low dimension. However, the solver is hard to initialize if only an initial guess on the state trajectory is available or it may not converge at all in the context of unstable systems.

Collocation Both the control and the state trajectories are discretized. Additional constraints on the state trajectory are introduced so that the dynamics equation is respected. The problem can then be easily initialized from a given state trajectory and collocation handles well unstable dynamics. However, a very fine grid is required to make the state trajectory closer to the true dynamics of the system.

Multiple shooting Ideas from both previous methods are employed. The integration interval is divided on a coarser time grid into a fixed number of multiple-shooting intervals, separated by nodes. Each subinterval is additionally discretized and integrated independently, ensuring continuity at the nodes by imposing suitable equality constraints. In this way, the optimization problem remains low-dimensional and can be easily warm-started with an initial guess on the state trajectory. Furthermore, multiple shooting is really suited for multi-phase dynamics, as each phase is set independent to the others. The convergence properties of multiple shooting and its faithful representation of the system dynamics make it preferable to single shooting and collocation. Therefore, this is the method of choice in this work.

6.3.5 Implementation

As optimization library, the framework employs MUSCOD-II [52], a general-purpose solver based on multiple shooting, developed by the Optimization

and Simulation group at the University of Heidelberg. An alternative open-source implementation is ACADO [45]. Notice that the current version of MUSCOD-II is not able to deal with the subtle precaution of using $SE(3)$ and its corresponding tangent space to represent orientations and their time evolution. This being the case, the trick of using the fictitious ankle joint coordinates and their time derivatives solves the problem. Otherwise, local approximating coordinates of $SE(3)$, such as Euler angles, can be employed.

To compute the evolution of the system, the C++ dynamics library Pinocchio [13] is employed. This is an efficient open source implementation of the dynamics of general polyarticulated systems, developed at LAAS-CNRS. This library, based on the screw theory formulation of the dynamic algorithms [29, 30], employs the Eigen linear algebra library [39] and template meta-programming [58] for greater efficiency.

PART II - COMPLIANT HUMANOIDS

Chapter 7

Results

In the following, we demonstrate the effectiveness of our proposed framework over two different robot structures. The first one is a simple planar compass model, with only two joints, one per hip. For this model, 9 multiple-shooting nodes were employed. The second one corresponds to a full humanoid in 3D, fitted with knees, hips, neck, shoulders and elbows, with a total of 12 DoFs (including 3 at the stance ankle). Here, we used 10 multiple-shooting nodes. For the two structures, we test rigid joints, series-elastic actuators and parallel-elastic actuators. We optimize both the step length and the time duration, as well as the intrinsic parameters of the actuators in the case of compliantly-actuated models. The step length has a lower bound of 0.6m. Concerning the inertial parameters, they follow a standard human distribution.

The simulations are divided into two series: in the first one, we employ the squared norm cost function (6.14), while in the second one we employ the CoT (6.13) or (6.15) depending on the type of actuators. For all scenarios, we use 10 multiple-shooting nodes; the maximum computation time to solve the OCP is around 1 min for the more complex walker. We refer the reader to the accompanying video¹ to fully appreciate the resulting gaits.

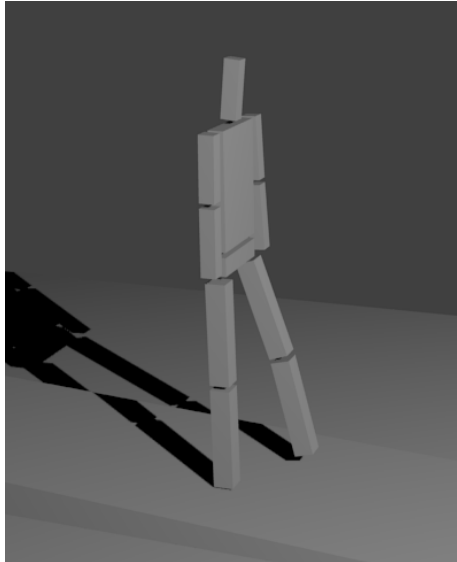
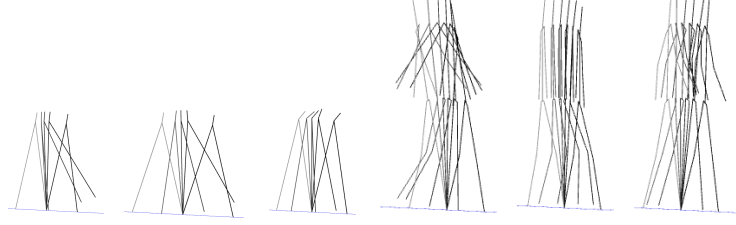


Figure 7.1 The humanoid model employed in the simulations.

¹<http://ieeexplore.ieee.org/document/8202228/media>



Type	rigid	SEA	parallel	rigid	SEA	parallel
Cost [m·N ²]	0.09563	0.00004	0.00060	0.62844	0.12800	0.36654
Step length [m]	0.60000	0.92726	0.65017	0.60000	0.60000	0.60000
Time [s]	0.74361	0.90204	0.73318	0.64572	0.81010	0.51664
Speed [m/s]	0.80687	1.02795	0.88679	0.92919	0.74065	1.16135

TABLE 7.1 Output of the simulations employing the squared torque as objective function.

7.1 Squared torque cost function

	Series elastic actuators		Parallel elastic actuators	
	Σ [Nm/rad]	B [kg m ²]	Σ [Nm/rad]	$\bar{q}$ [rad]
compass	1254.678	0.724	11.128	0.484
knees	1642.951	0.500	51.829	-0.032
hips	1564.161	0.434	17.847	-0.121
neck	1434.195	0.566	61.210	0.040
shoulders	1388.581	0.573	0.000	0.021
elbows	1480.827	0.426	30.969	-0.189

TABLE 7.2 Dynamic parameters of the compliant actuators, optimized w.r.t the squared torque cost function. On the first line, the values for the compass walker are shown, equal or symmetrical for both hips. In the rest of the table, the values for the joints of the complete walker are shown.

In this section, we study the influence of the squared torque cost function for the two proposed models with the three types of actuators. The output of the optimizer is displayed in Tab. 7.1 and the optimized parameters for the compliant models are shown in Tab. 7.2.

7.1.1 Compass walker

From Fig. 7.2, it appears that the torques exerted by the motors in the context of elastic models are much smaller than the ones exerted in the rigid model. Consequently, the values of the cost function for elastic models are quite similar and much smaller than those of the rigid example (see Tab. 7.1,

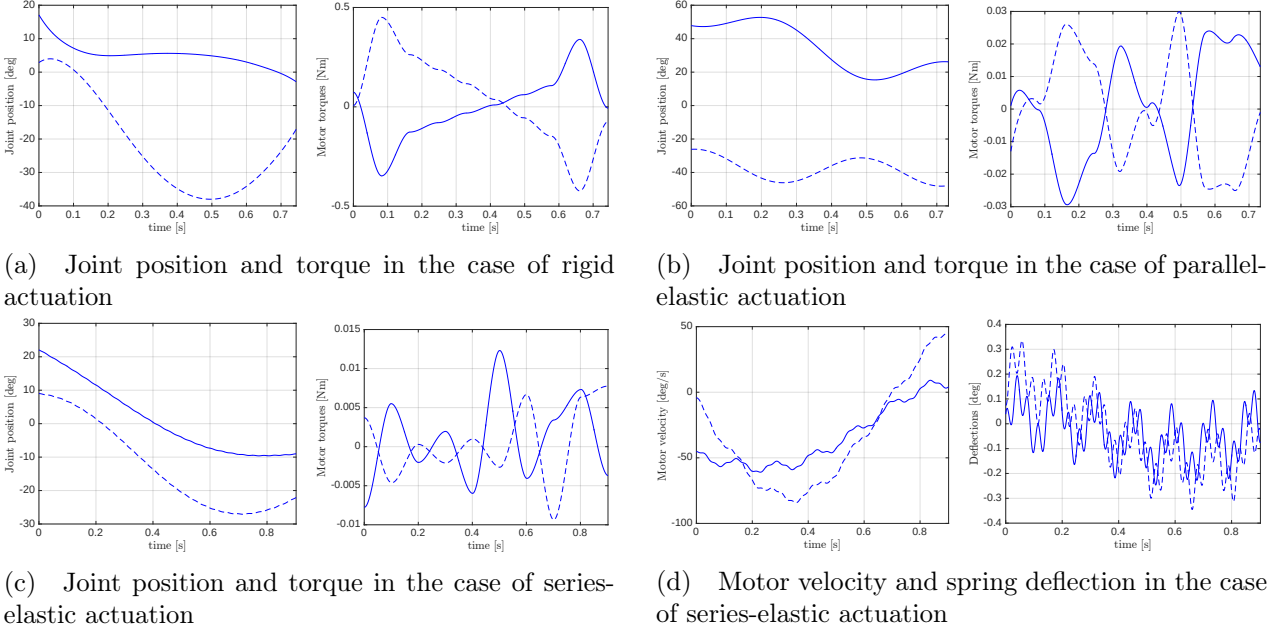


Figure 7.2 Plots of the joint positions and torques trajectories for the simple walker with the squared norm of the torque as objective function. Legend: solid=stance hip, dashed=free hip.

first row). For all cases, the joint trajectory is reasonably smooth. In the context of serial actuation, the joint deflection is kept within a small range, with high frequency oscillations appearing in the joint trajectory. On the other hand, the motor velocity is subject to strong oscillations. This is not surprising, since the optimal solution need not be smooth, and oscillatory behavior is typical of SEAs.

7.1.2 Full walker

We now focus on the rigidly-actuated complete walker. From Tab. 7.1, we see that the motion appears to be very natural, also exploiting the arms to obtain more thrust, while the elbows are barely moving (see the accompanying video). From Fig. 7.3a, we see that the exerted torques are very low, the largest one being at the free knee, reaching a peak of about ± 1 [Nm]. The nominal performance again improves using SEAs. From Fig. 7.3c, we notice some visible, small oscillations in the joint configuration, particularly in the shoulders and the neck. Also, notice that the joint deflections are below one degree.

We can see (also from Tab. 7.1) that the swing motion of the arms is reduced. The performance of the parallel-elastic actuated robot is also good.

PART II - COMPLIANT HUMANOIDS

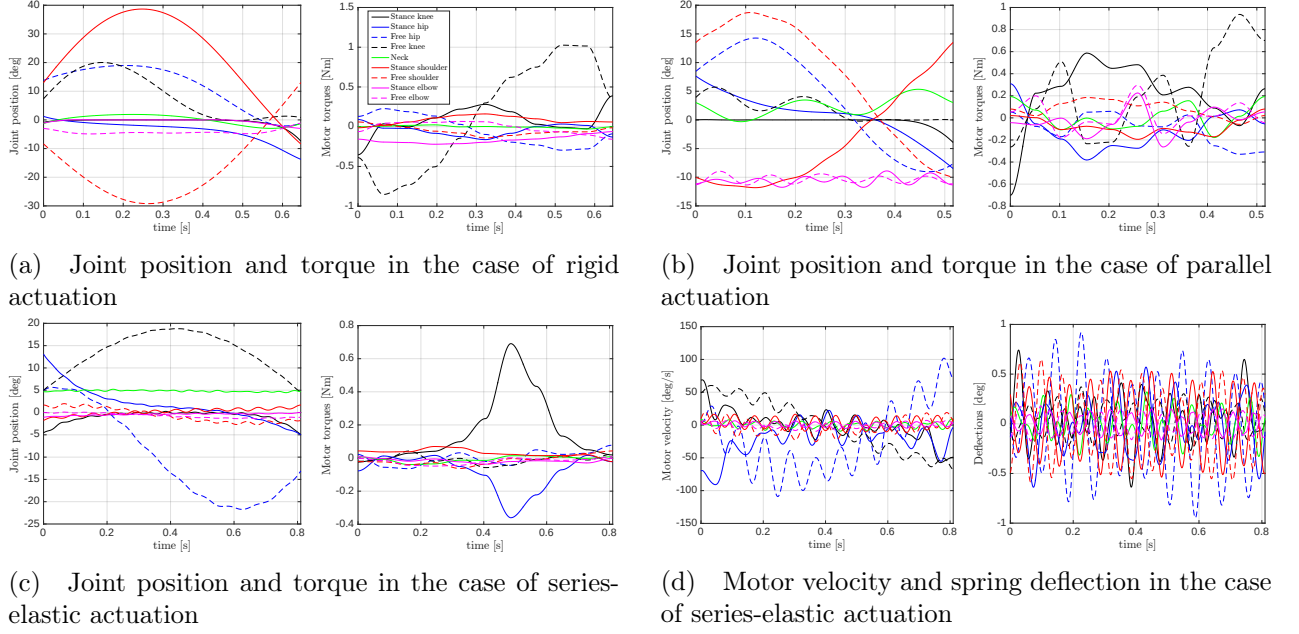


Figure 7.3 Plots of the joint positions and torque trajectories for the full walker with the squared norm of the torque as objective function. Legend: solid/dashed = stance/swing, black = knees, blue = hips, green = neck, red = shoulders, magenta = elbows.

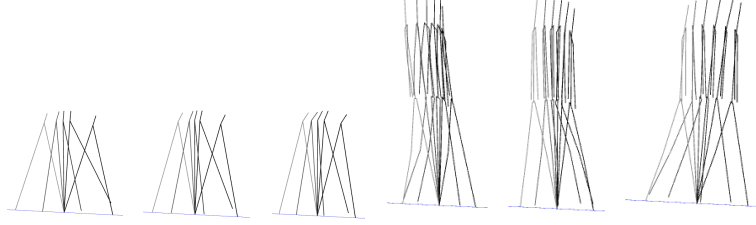
Again, we can notice the role of the arms during motion. It is worth to notice that in Tab. 7.2 the spring stiffness value reaches zero for the shoulder parallel-elastic actuator. It is clear that this means that in this case the parallel elastic actuator has become a rigid one.

7.2 Cost of transport objective function

In this section, we study the influence of the cost of transport as objective function on the various case studies. Tables 7.3 and 7.4, as well as Figs. 7.4 and 7.5 show the output of this second series of simulations.

7.2.1 Compass walker

For the compass walker with CoT as objective function, higher frequency oscillations appear on the SEAs. In the meantime, all the torque trajectories have similar amplitude. At this stage, it is worth to mention that during a non-negligible amount of time the torque trajectories are identically equal to zero for both hips, although the peak values are higher. This means that, during this period, the OCP solver manages to find out a strategy to exploit



Type	rigid	SEA	parallel	rigid	SEA	parallel
Cost [W/m]	0.45729	0.04651	0.03863	0.51831	0.00844	0.04792
Step length [m]	0.91416	0.78682	0.69466	0.60000	0.60000	0.85526
Time [s]	0.80094	0.87833	0.61705	0.59480	0.82184	0.47053
Speed [m/s]	1.14136	0.89581	1.12577	1.00875	0.73007	1.81764

TABLE 7.3 Output of the simulations employing the CoT as objective function.

	Series elastic actuators		Parallel elastic actuators	
	Σ [Nm/rad]	B [kg m ²]	Σ [Nm/rad]	$\bar{q}$ [rad]
compass	1231.374	0.308	37.648	0.485
knees	1588.078	0.758	129.195	-0.017
hips	1378.095	0.280	44.888	-0.069
neck	1415.037	0.479	60.235	0.085
shoulders	1485.496	0.638	131.771	-0.055
elbows	1341.142	0.479	37.859	-0.011

TABLE 7.4 Dynamic parameters of the compliant actuators, optimized w.r.t the CoT cost function.

the walker's intrinsic dynamics only, which was not the case with the previous cost function.

7.2.2 Full walker

The same patterns as in the previous series apply here, with some exceptions. In general, we see that the arms are not moving, but lie rigidly next to the body; the parallel-elastic actuated walker is holding them slightly to its front. Many joints are left almost unactuated (practically all of them except the knees and one elbow for the SEA case). It should be noted that minimizing the CoT required in general much more computation time than minimizing the squared torque.

PART II - COMPLIANT HUMANOIDS

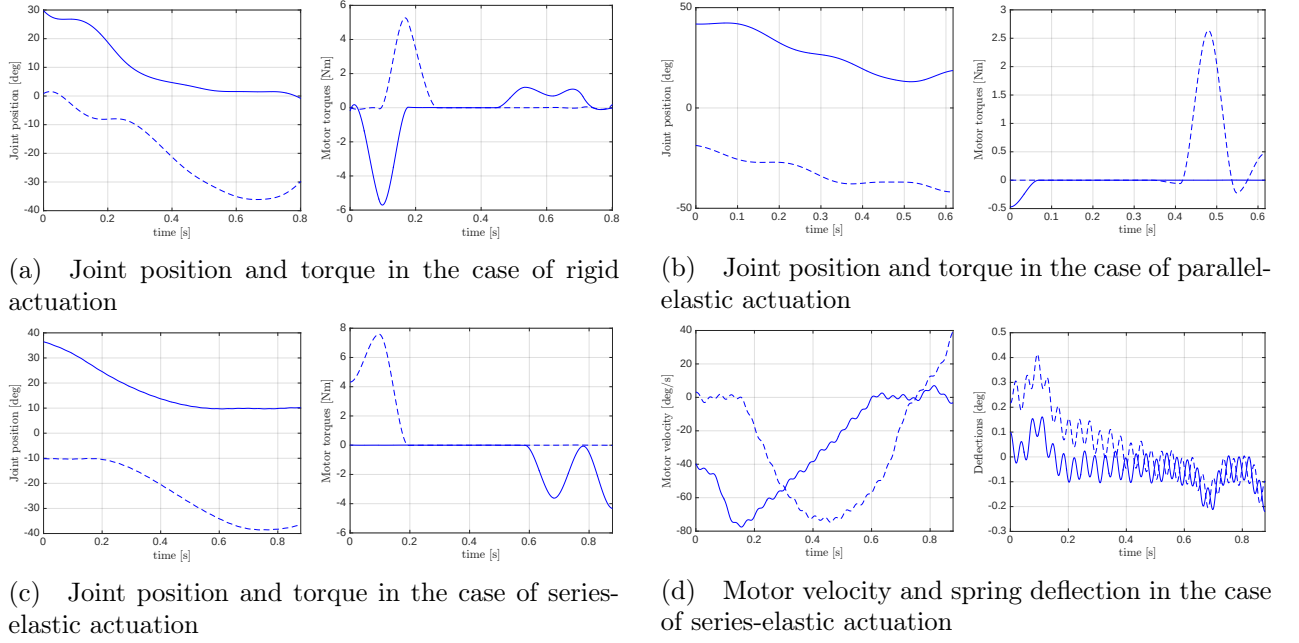


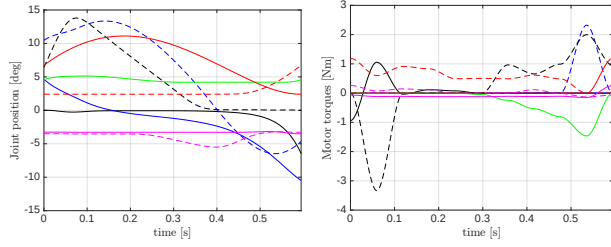
Figure 7.4 Plots of the joint positions and torques trajectories for the simple walker with the cost of transport as objective function.

7.3 Serial vs. parallel actuations

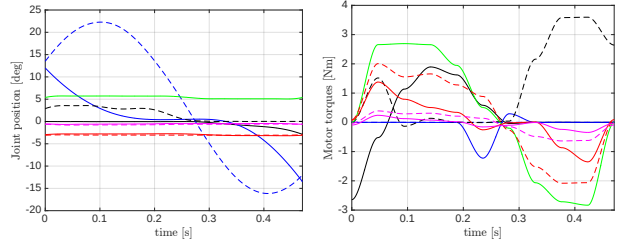
For both cost functions, it appears SEAs produce high oscillations of the motor shafts. In other words, with such kind of actuation both the electronics and the motor are highly stressed by the spring element and must be designed accordingly. In addition, comparing stiffness values in Tabs. 7.2 and 7.4, it appears that SEAs need higher stiffness (bigger by one order of magnitude). Then springs occupy a larger volume in the robot in the serial case.

For all simulations, it is worth to notice that parallel elastic actuators unlike SEAs tend to make the mean motor torque value close to zero. In other words, with serial elastic actuators, motors have to support a greater burden. This also means that optimal control allows to correctly exploit the spring as storage entity in the context of parallel actuators and minimally stress the motors.

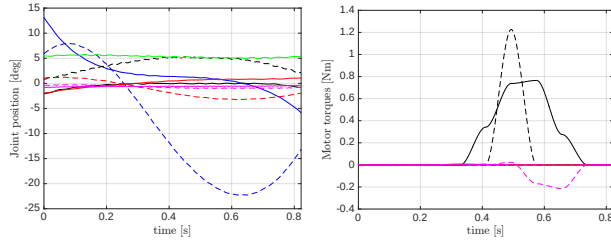
CHAPTER 7. RESULTS



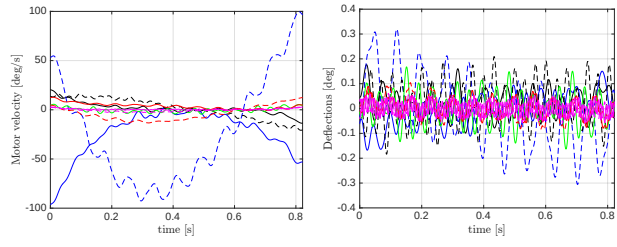
(a) Joint position and torque in the case of rigid actuation



(b) Joint position and torque in the case of parallel actuation



(c) Joint position and torque in the case of series-elastic actuation



(d) Motor velocity and spring deflection in the case of series-elastic actuation

Figure 7.5 Plots of the joint positions and torques trajectories for the full walker with the cost of transport as objective function.

CONCLUSIONS

CONCLUSIONS

Conclusions

In this work, the dynamics of compliant-joint robots was analyzed in detail and solutions were developed to aid the design and the control of such kind of robots. Many different categories of robots were considered, including robots equipped with Series Elastic Actuators (SEA), Variable Stiffness Actuators (VSA), fixed-base manipulators, and humanoids.

On the control side, an efficient numerical solution has been presented for computing the inverse dynamics of robot manipulators with elastic joints, based on a differential extension of the recursive Newton-Euler algorithm for rigid robots. The solution is general and has a computational complexity that grows linearly with the number of joints, as in the rigid case. To achieve this result, recursive equations are defined up to the fourth differential order for link motion variables, and up to the second time derivative for forces and torques. The recursive numerical algorithm avoids even in the joint elastic case the effort of deriving symbolic models and customizing the dynamics for efficiency when the number of joints is large. Many extensions and variants of the algorithm have been introduced, including different kinematic conventions, the case of nonlinear spring, friction, and, most importantly, Variable Stiffness Actuators and dedicated versions for feedback linearization control.

On the design side, a framework capable to design and analyze the actuation system of complex humanoids was presented. It has been tested it over different robot structures, with various cost functions, showing that in general our framework is able to properly exploit the elastic potential energy



Figure 7 The VSA-actuated humanoid torso David. Image edited from [93].

CONCLUSIONS

along the complete trajectory.

The complexity of a Lagrangian-based feedback linearization design has barred so far the use of advanced model-based control laws for all but very simple robotic structures with elastic joints. Using instead the extended Newton-Euler algorithm, the issue of real-time implementation of such complex laws, such as feedback linearization control, was addressed for robots with joint elasticity for the first time. The resulting procedures have a very general nature and can be readily applied to any kind of robot whose dynamic parameters are known. For VSA, some analytical study of the spring model is still necessary for the pure inverse dynamics: the general procedure is conceptually divided in two distinct parts, the first one consisting in the fourth-order Newton-Euler recursive equations, as in robots with elastic joints; the core of the second part, instead, requires solving for each joint a two-by-two nonlinear system in terms of the desired joint motion-stiffness trajectory. This system depends on the specific technology of the VSA device and its solution, which is in general obtained numerically by a root finding method, has been illustrated here for some representative cases. The variant which was proposed for feedback linearization control, instead, using full state measurements, does not require a numerical root finding method and can be used directly, once the spring model is known.

Similarly, traditional design schemes may prove limited when the overall dynamics of the whole robot is to be considered optimized for a given goal. The proposed framework, devised for compliant humanoids, provides a first answer to the need of developing of a useful tool for the design of energy-efficient complex humanoid structures.

As all model-based approaches, the effectiveness of the proposed solutions is strongly depended on the precision of the available model, either in its use, as in the case of control, or in its physical realization, in the design case. In the control case, this is not only true for the proposed numerical algorithms, but also for their symbolic counterparts: the quality of the results will depend on the accuracy of the kinematic and dynamic parameters, in particular for what concerns the spring model. A sensitivity analysis of the algorithms to perturbed dynamic parameters could be beneficial. For the design framework, the inclusion of robustness measures in the optimization might be required.

The developed control algorithms may be further extended by considering different classes of robots. For instance, floating-base robots can be studied; arguably, this would transform the problem into a different one, namely, the computation of the the hybrid dynamics. Additionally, nonholonomic dy-

CONCLUSIONS

dynamic constraints can be considered. Often, the two topics are related, as in the case of quadrotor-based manipulators [92, 78]. As a conceptually simpler extension, VSA-NEA may be modified to deal with actuators in serial mode, or further extended to more general kinds of actuation, such as Variable Impedance Actuators, which also exhibit variable damping. Further, the same methodology used to obtain EJNEA could be employed to derive higher-order versions of the articulated body algorithm (ABA), to be used in particular in the context of feedback linearization. A thorough computational analysis could establish the conditions under which its use may result convenient over $O(N^3)$ methods.

The design framework can be extended in different ways. In the first place, the selection of the type of actuator should be automatic for each joint; the most straightforward implementation of this concept would be setting up a mixed-integer optimal control problem (MIOCP) which will choose the optimal actuator combination minimizing the objective [65, 51]. Actuators individually combining serial and parallel elastic connections could also be interesting. MIOCP would also allow to work with discrete parameter values, enabling for instance the selection of specific actuators within a collection of commercially available ones. Or, taking an opposite approach, the actuator design could be considered in greater detail, by taking into consideration physical constraints which may make not all combinations of dynamic parameters actually realizable; for instance, the increase of spring stiffness should also correspond to a greater mass. Ideally, the whole mass distribution of the humanoid should be considered. More sophisticated contact models, such as round feet or elastic contacts, may be analyzed. Different tasks such as jumping or running can be studied; it would be interesting to see how the parameters change in these cases. Another extension could be the implementation of a strategy to avoid local minima. A Montecarlo method on the initial values seems the most promising approach. A stochastic solution may also be employed to achieve robustness, not only for what the dynamic parameters are concerned, but also in consideration of uneven terrain.

Still concerning the design framework, the most open and promising aspect consists in a deep study of not only the mechanics of the robot, but also its electromechanics. This might be required in order to account for low-level limitations in, e.g., current, tension and tracking controllers. Also, the design of a cost function that faithfully reflects the actual energy consumption of the robot might involve modeling the electromechanics, for instance to include the contribution of motors and regenerative system as in Cheetah robot [70]. Indeed, the influence of the gearbox and motor on the energy consumption can be very important in practice [5, 84, 85], and should therefore be considered to obtain efficient behaviors. Depending on how much the

CONCLUSIONS

energy consumption model of the robot is similar to that of human beings, the resulting gaits may or may not be human-like. One might wonder if it is even desirable, then, that a robot exclusively pursue energy efficiency without regard for the biological inspirations. Most likely, the answer will depend on the application.

The hope and vision of most people currently working in robotics is to one day see robots as a widespread presence in our everyday lives. To this end, it is fundamental that robots be safe and energetically efficient. Thus, the future development of robotics and its dissemination will not be able to transcend from the adoption of soft technologies.

APPENDICES

APPENDICES

Appendix A

Derivation of EJNEA

Here, we show how the algorithm EJNEA can be derived.

Notice equations (2.25), (2.26), (2.29), (2.30), (2.35), (2.38), (2.41), (2.44), and (2.47) are simply the standard RNEA, as applied to the links of an elastic joint robot; therefore, their derivation will not be shown. Here, we will only focus on the derivation of the additional formulas, which constitute the novelty of the algorithm.

Third-order dynamic equations, i.e. (2.27), (2.31), (2.32), (2.36), (2.39), (2.42), (2.45), and (2.48), have first appeared in [38], together with their derivation. Nonetheless, they will reported here for completeness.

Derivation of EJNEA is realized starting from regular RNEA, referred to base coordinates. Here, we also include the recursive formulation of link velocity $\mathbf{v}_i$, as it will be employed during the derivation.

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} + \dot{\theta}_i \hat{\mathbf{z}}_{i-1} \quad (\text{A.1})$$

$$\boldsymbol{\gamma}_i = \boldsymbol{\gamma}_{i-1} + \ddot{\theta}_i \hat{\mathbf{z}}_{i-1} + \boldsymbol{\omega}_{i-1} \times \dot{\theta}_i \hat{\mathbf{z}}_{i-1} \quad (\text{A.2})$$

$$\mathbf{v}_i = \mathbf{v}_{i-1} + \boldsymbol{\omega}_i \times \mathbf{p}_{i,i-1} + \dot{d}_i \hat{\mathbf{z}}_{i-1} \quad (\text{A.3})$$

$$\begin{aligned} \mathbf{a}_i = & \mathbf{a}_{i-1} + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{i,i-1}) \\ & + \boldsymbol{\gamma}_i \times \mathbf{p}_{i,i-1} + \ddot{d}_i \hat{\mathbf{z}}_{i-1} + 2\dot{d}_i (\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i-1}) \end{aligned} \quad (\text{A.4})$$

$$\mathbf{a}_{c_i} = \mathbf{a}_i + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{c_i,i}) + \boldsymbol{\gamma}_i \times \mathbf{p}_{c_i,i} \quad (\text{A.5})$$

$$\mathbf{F}_i = m_i \mathbf{a}_{c_i} \quad (\text{A.6})$$

$$\mathbf{N}_i = \mathbf{I}_i \boldsymbol{\gamma}_i + \boldsymbol{\omega}_i \times (\mathbf{I}_i \boldsymbol{\omega}_i) \quad (\text{A.7})$$

$$\mathbf{f}_i = \mathbf{f}_{i+1} + \mathbf{F}_i \quad (\text{A.8})$$

$$\mathbf{n}_i = \mathbf{n}_{i+1} + \mathbf{p}_{c_i,i} \times \mathbf{F}_i + \mathbf{p}_{i,i-1} \times \mathbf{f}_i + \mathbf{N}_i \quad (\text{A.9})$$

APPENDICES

$$\tau_{ei} = \begin{cases} \mathbf{n}_i^T \hat{\mathbf{z}}_{i-1} & \text{if joint } i \text{ is revolute} \\ \mathbf{f}_i^T \hat{\mathbf{z}}_{i-1} & \text{if joint } i \text{ is prismatic} \end{cases} \quad (\text{A.10})$$

The equations of EJNEA are obtained by symbolically differentiating (A.1)–(A.10) with respect to time, and then referring both sides of each equation to link coordinates.

A.1 Forward recursion

A.1.1 Angular jerk

The expression for the angular jerk $\boldsymbol{\iota}_i = \dot{\boldsymbol{\gamma}}_i$ can be obtained by differentiating (A.2):

$$\boldsymbol{\iota}_i = \boldsymbol{\iota}_{i-1} + \ddot{\theta}_i \hat{\mathbf{z}}_{i-1} + \ddot{\theta}_i \dot{\hat{\mathbf{z}}}_{i-1} + \boldsymbol{\gamma}_{i-1} \times \dot{\hat{\mathbf{z}}}_{i-1} + \boldsymbol{\omega}_{i-1} \times \ddot{\hat{\mathbf{z}}}_{i-1} + \boldsymbol{\omega}_{i-1} \times \dot{\theta}_i \dot{\hat{\mathbf{z}}}_{i-1}. \quad (\text{A.11})$$

Remembering

$$\hat{\mathbf{z}}_{i-1} = \mathbf{R}_{i-1}^{i-1} \hat{\mathbf{z}}_{i-1} = \mathbf{R}_{i-1} \hat{\mathbf{z}}_0, \quad (\text{A.12})$$

we recall the well-known result

$$\dot{\mathbf{R}}_i = \mathbf{S}(\boldsymbol{\omega}_i) \mathbf{R}_i, \quad (\text{A.13})$$

where $\mathbf{S}(\boldsymbol{\omega}_i)$ is the skew-symmetric matrix computed from $\boldsymbol{\omega}_i$ that performs the vector product operation, i.e., such that $\mathbf{S}(\boldsymbol{\omega}_i) \mathbf{v} = \boldsymbol{\omega}_i \times \mathbf{v}$, for every vector $\mathbf{v}$. Therefore

$$\dot{\hat{\mathbf{z}}}_{i-1} = \dot{\mathbf{R}}_{i-1} \hat{\mathbf{z}}_0 = \mathbf{S}(\boldsymbol{\omega}_{i-1}) \mathbf{R}_{i-1} \hat{\mathbf{z}}_0 = \mathbf{S}(\boldsymbol{\omega}_{i-1}) \hat{\mathbf{z}}_{i-1} = \boldsymbol{\omega}_{i-1} \times \hat{\mathbf{z}}_{i-1}. \quad (\text{A.14})$$

With (A.14), we can rewrite (A.11) as

$$\begin{aligned} \boldsymbol{\iota}_i = \boldsymbol{\iota}_{i-1} + \ddot{\theta}_i \hat{\mathbf{z}}_{i-1} + \ddot{\theta}_i (\boldsymbol{\omega}_{i-1} \times \hat{\mathbf{z}}_{i-1}) + \boldsymbol{\gamma}_{i-1} \times \dot{\hat{\mathbf{z}}}_{i-1} \\ + \boldsymbol{\omega}_{i-1} \times \ddot{\hat{\mathbf{z}}}_{i-1} + \boldsymbol{\omega}_{i-1} \times \dot{\theta}_i (\boldsymbol{\omega}_{i-1} \times \hat{\mathbf{z}}_{i-1}) \end{aligned} \quad (\text{A.15})$$

which is easily rearranged as

$$\begin{aligned} \boldsymbol{\iota}_i = \boldsymbol{\iota}_{i-1} + \ddot{\theta}_i \hat{\mathbf{z}}_{i-1} + 2\ddot{\theta}_i (\boldsymbol{\omega}_{i-1} \times \hat{\mathbf{z}}_{i-1}) \\ + \dot{\theta}_i \boldsymbol{\gamma}_{i-1} \times \hat{\mathbf{z}}_{i-1} + \dot{\theta}_i \boldsymbol{\omega}_{i-1} \times (\boldsymbol{\omega}_{i-1} \times \hat{\mathbf{z}}_{i-1}) \end{aligned} \quad (\text{A.16})$$

Finally, we refer everything to frame i . This is done by pre-multiplying both sides by the rotation matrix ${}^i \mathbf{R}_0 = \mathbf{R}_i^T$. Noticing

$${}^i \mathbf{R}_{i-1}^{i-1} \mathbf{R}_0 \quad (\text{A.17})$$

and remembering (A.12), we obtain (2.27), after some rearrangements.

A.1.2 Angular snap

Derivation of (2.28) can start from (A.16). Differentiating it with respect to time, we obtain:

$$\begin{aligned}
 \mathbf{s}_i &= \mathbf{s}_{i-1} + \ddot{\theta}_i \hat{\mathbf{z}}_{i-1} + \ddot{\theta}_i \dot{\hat{\mathbf{z}}}_{i-1} + 2\ddot{\theta}_i (\boldsymbol{\omega}_{i-1} \times \hat{\mathbf{z}}_{i-1}) \\
 &\quad + 2\ddot{\theta}_i (\boldsymbol{\gamma}_{i-1} \times \hat{\mathbf{z}}_{i-1}) + 2\ddot{\theta}_i (\boldsymbol{\omega}_{i-1} \times \dot{\hat{\mathbf{z}}}_{i-1}) \\
 &\quad + \ddot{\theta}_i (\boldsymbol{\gamma}_{i-1} \times \dot{\hat{\mathbf{z}}}_{i-1}) + \dot{\theta}_i (\boldsymbol{\iota}_{i-1} \times \hat{\mathbf{z}}_{i-1}) + \dot{\theta}_i (\boldsymbol{\gamma}_{i-1} \times \dot{\hat{\mathbf{z}}}_{i-1}) \\
 &\quad + \dot{\theta}_i \boldsymbol{\omega}_{i-1} \times (\boldsymbol{\omega}_{i-1} \times \hat{\mathbf{z}}_{i-1}) + \dot{\theta}_i \boldsymbol{\gamma}_{i-1} \times (\boldsymbol{\omega}_{i-1} \times \hat{\mathbf{z}}_{i-1}) \\
 &\quad + \dot{\theta}_i \boldsymbol{\omega}_{i-1} \times (\boldsymbol{\gamma}_{i-1} \times \hat{\mathbf{z}}_{i-1} + \boldsymbol{\omega}_{i-1} \times \dot{\hat{\mathbf{z}}}_{i-1})
 \end{aligned} \tag{A.18}$$

Remembering (A.14), we can rewrite the previous equation in the following form

$$\begin{aligned}
 \mathbf{s}_i &= \mathbf{s}_{i-1} + \ddot{\theta}_i \hat{\mathbf{z}}_{i-1} + 3\ddot{\theta}_i \boldsymbol{\omega}_{i-1} \times \hat{\mathbf{z}}_{i-1} + 3\ddot{\theta}_i \boldsymbol{\gamma}_{i-1} \times \hat{\mathbf{z}}_{i-1} \\
 &\quad + 3\ddot{\theta}_i \boldsymbol{\omega}_{i-1} \times (\boldsymbol{\omega}_{i-1} \times \hat{\mathbf{z}}_{i-1}) + \dot{\theta}_i (\boldsymbol{\iota}_{i-1} \times \hat{\mathbf{z}}_{i-1}) \\
 &\quad + \dot{\theta}_i \boldsymbol{\omega}_{i-1} \times [\boldsymbol{\gamma}_{i-1} \times \hat{\mathbf{z}}_{i-1} + \boldsymbol{\omega}_{i-1} \times (\boldsymbol{\omega}_{i-1} \times \hat{\mathbf{z}}_{i-1})] \\
 &\quad + 2\dot{\theta}_i \boldsymbol{\gamma}_{i-1} \times (\boldsymbol{\omega}_{i-1} \times \hat{\mathbf{z}}_{i-1})
 \end{aligned} \tag{A.19}$$

Describing the preceding equation with respect to the frame i and rearranging a little bit, we obtain (2.28).

A.1.3 Linear jerk

In the same way as done for $\boldsymbol{\iota}_i$, derivation of (2.31) can start by differentiating the base frame form of the acceleration $\mathbf{a}_i$, given by (A.4):

$$\begin{aligned}
 \mathbf{j}_i &= \mathbf{j}_{i-1} + \boldsymbol{\gamma}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{i,i-1}) + \boldsymbol{\omega}_i \times (\boldsymbol{\gamma}_i \times \mathbf{p}_{i,i-1}) + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \dot{\mathbf{p}}_{i,i-1}) \\
 &\quad + \boldsymbol{\iota}_i \times \mathbf{p}_{i,i-1} + \boldsymbol{\gamma}_i \times \dot{\mathbf{p}}_{i,i-1} + \ddot{d}_i \hat{\mathbf{z}}_{i-1} + \dot{d}_i \dot{\hat{\mathbf{z}}}_{i-1} \\
 &\quad + 2\ddot{d}_i (\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i-1}) + 2\dot{d}_i (\boldsymbol{\gamma}_i \times \hat{\mathbf{z}}_{i-1}) + 2\dot{d}_i (\boldsymbol{\omega}_i \times \dot{\hat{\mathbf{z}}}_{i-1})
 \end{aligned} \tag{A.20}$$

Knowing $\mathbf{p}_{i,i-1} = \mathbf{p}_i - \mathbf{p}_{i-1}$, it is possible to define $\dot{\mathbf{p}}_{i,i-1} = \dot{\mathbf{p}}_i - \dot{\mathbf{p}}_{i-1} = \mathbf{v}_i - \mathbf{v}_{i-1}$, which leads, by means of (A.3), to

$$\dot{\mathbf{p}}_{i,i-1} = \boldsymbol{\omega}_i \times \mathbf{p}_{i,i-1} + \dot{d}_i \hat{\mathbf{z}}_{i-1}. \tag{A.21}$$

Moreover, from (A.14) and (A.1), it is possible to rewrite $\dot{\hat{\mathbf{z}}}_{i-1}$ as

$$\dot{\hat{\mathbf{z}}}_{i-1} = \boldsymbol{\omega}_{i-1} \times \hat{\mathbf{z}}_{i-1} = (\boldsymbol{\omega}_i - \dot{\theta}_i \hat{\mathbf{z}}_{i-1}) \times \hat{\mathbf{z}}_{i-1} = \boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i-1} \tag{A.22}$$

APPENDICES

since $\hat{\mathbf{z}}_{i-1} \times \hat{\mathbf{z}}_{i-1} = \mathbf{0}$. This allows us to rewrite the jerk as

$$\begin{aligned} \mathbf{j}_i &= \mathbf{j}_{i-1} + \boldsymbol{\iota}_i \times \mathbf{p}_{i,i-1} + 2\boldsymbol{\gamma}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{i,i-1}) \\ &\quad + \boldsymbol{\omega}_i \times [\boldsymbol{\gamma}_i \times \mathbf{p}_{i,i-1} + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{i,i-1})] + \ddot{\mathbf{d}}_i \hat{\mathbf{z}}_{i-1} \\ &\quad + 3\ddot{\mathbf{d}}_i \boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i-1} + 3\dot{\mathbf{d}}_i \boldsymbol{\gamma}_i \times \hat{\mathbf{z}}_{i-1} + 3\dot{\mathbf{d}}_i \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i-1}) \end{aligned} \quad (\text{A.23})$$

Finally, referring everything to frame i and rearranging, we obtain (2.31).

The linear jerk of the CoM, i.e. $\mathbf{j}_{c_i}$, can be computed from $\mathbf{j}_i$. Consider the frame with origin located at the center of mass of link i , parallel to frame i . Equation (2.32) can be obtained directly from (2.31), replacing frame i with frame c_i and frame $i-1$ with frame i . Since frame i and frame c_i are attached to the same link, then $\dot{\mathbf{d}}_{c_i}$, $\ddot{\mathbf{d}}_{c_i}$, and $\ddot{\mathbf{d}}_{c_i}$ are all equal to zero. We obtain

$$\begin{aligned} {}^{c_i}\mathbf{j}_{c_i} &= {}^{c_i}\mathbf{R}_i {}^i\mathbf{j}_i + {}^{c_i}\boldsymbol{\iota}_{c_i} \times {}^{c_i}\mathbf{p}_{c_i,i} + 2 {}^{c_i}\boldsymbol{\gamma}_{c_i} \times ({}^{c_i}\boldsymbol{\omega}_{c_i} \times {}^{c_i}\mathbf{p}_{c_i,i}) \\ &\quad + {}^{c_i}\boldsymbol{\omega}_{c_i} \times [{}^{c_i}\boldsymbol{\gamma}_{c_i} \times {}^{c_i}\mathbf{p}_{c_i,i} + {}^{c_i}\boldsymbol{\omega}_{c_i} \times ({}^{c_i}\boldsymbol{\omega}_{c_i} \times {}^{c_i}\mathbf{p}_{c_i,i})]. \end{aligned} \quad (\text{A.24})$$

Since frame i and c_i are parallel, then ${}^{c_i}\mathbf{R}_i$ is equal to the identity, while ${}^{c_i}\boldsymbol{\omega}_{c_i} = {}^i\boldsymbol{\omega}_i$, ${}^{c_i}\boldsymbol{\gamma}_{c_i} = {}^i\boldsymbol{\gamma}_i$, ${}^{c_i}\boldsymbol{\iota}_{c_i} = {}^i\boldsymbol{\iota}_i$, ${}^{c_i}\mathbf{p}_{c_i,i} = {}^i\mathbf{p}_{c_i,i}$ and ${}^{c_i}\mathbf{j}_{c_i} = {}^i\mathbf{j}_{c_i}$.

A.1.4 Linear snap

We can obtain the snap $\mathbf{s}_i$ starting from (A.23). Differentiating:

$$\begin{aligned} \mathbf{s}_i &= \mathbf{s}_{i-1} + \boldsymbol{\varsigma}_i \times \mathbf{p}_{i,i-1} + \boldsymbol{\iota}_i \times \dot{\mathbf{p}}_{i,i-1} + 2\boldsymbol{\iota}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{i,i-1}) \\ &\quad + 2\boldsymbol{\gamma}_i \times (\boldsymbol{\gamma}_i \times \mathbf{p}_{i,i-1}) + 2\boldsymbol{\gamma}_i \times (\boldsymbol{\omega}_i \times \dot{\mathbf{p}}_{i,i-1}) \\ &\quad + \boldsymbol{\gamma}_i \times [\boldsymbol{\gamma}_i \times \mathbf{p}_{i,i-1} + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{i,i-1})] \\ &\quad + \boldsymbol{\omega}_i \times [\boldsymbol{\iota}_i \times \mathbf{p}_{i,i-1} + \boldsymbol{\gamma}_i \times \dot{\mathbf{p}}_{i,i-1} + \boldsymbol{\gamma}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{i,i-1})] \\ &\quad + \boldsymbol{\omega}_i \times (\boldsymbol{\gamma}_i \times \mathbf{p}_{i,i-1} + \boldsymbol{\omega}_i \times \dot{\mathbf{p}}_{i,i-1}) + \ddot{\mathbf{d}}_i \hat{\mathbf{z}}_{i-1} \\ &\quad + \ddot{\mathbf{d}}_i \dot{\hat{\mathbf{z}}}_{i-1} + 3\ddot{\mathbf{d}}_i \boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i-1} + 6\ddot{\mathbf{d}}_i \boldsymbol{\gamma}_i \times \hat{\mathbf{z}}_{i-1} \\ &\quad + 3\ddot{\mathbf{d}}_i \boldsymbol{\omega}_i \times \dot{\hat{\mathbf{z}}}_{i-1} + 3\dot{\mathbf{d}}_i \boldsymbol{\iota}_i \times \hat{\mathbf{z}}_{i-1} + 3\dot{\mathbf{d}}_i \boldsymbol{\gamma}_i \times \dot{\hat{\mathbf{z}}}_{i-1} \\ &\quad + 3\dot{\mathbf{d}}_i \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i-1}) + 3\dot{\mathbf{d}}_i \boldsymbol{\gamma}_i \times (\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i-1}) \\ &\quad + 3\dot{\mathbf{d}}_i \boldsymbol{\omega}_i \times (\boldsymbol{\gamma}_i \times \hat{\mathbf{z}}_{i-1} + \boldsymbol{\omega}_i \times \dot{\hat{\mathbf{z}}}_{i-1}) \end{aligned} \quad (\text{A.25})$$

APPENDIX A. DERIVATION OF EJNEA

Then, applying the substitutions in (A.22) and (A.21), after some manipulations:

$$\begin{aligned}
\mathbf{s}_i = & \mathbf{s}_{i-1} + \boldsymbol{\varsigma}_i \times \mathbf{p}_{i,i-1} + \boldsymbol{\iota}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{i,i-1} + \dot{d}_i \hat{\mathbf{z}}_{i-1}) \\
& + 2\boldsymbol{\iota}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{i,i-1}) + 3\boldsymbol{\gamma}_i \times (\boldsymbol{\gamma}_i \times \mathbf{p}_{i,i-1}) \\
& + 2\boldsymbol{\gamma}_i \times [\boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{i,i-1} + \dot{d}_i \hat{\mathbf{z}}_{i-1})] \\
& + \boldsymbol{\gamma}_i \times [\boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{i,i-1})] + \boldsymbol{\omega}_i \times (\boldsymbol{\iota}_i \times \mathbf{p}_{i,i-1}) \\
& + \boldsymbol{\omega}_i \times [\boldsymbol{\gamma}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{i,i-1} + \dot{d}_i \hat{\mathbf{z}}_{i-1})] \\
& + \boldsymbol{\omega}_i \times [\boldsymbol{\gamma}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{i,i-1})] + \boldsymbol{\omega}_i \times \{\boldsymbol{\omega}_i \times [\boldsymbol{\gamma}_i \times \mathbf{p}_{i,i-1} \\
& + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{i,i-1} + \dot{d}_i \hat{\mathbf{z}}_{i-1})]\} + \ddot{d}_i \hat{\mathbf{z}}_{i-1} \\
& + 4\ddot{d}_i \boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i-1} + 6\ddot{d}_i [\boldsymbol{\gamma}_i \times \hat{\mathbf{z}}_{i-1} + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i-1})] \\
& + 3\dot{d}_i \boldsymbol{\iota}_i \times \hat{\mathbf{z}}_{i-1} + 6\dot{d}_i \boldsymbol{\gamma}_i \times (\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i-1}) \\
& + 3\dot{d}_i \boldsymbol{\omega}_i \times (\boldsymbol{\gamma}_i \times \hat{\mathbf{z}}_{i-1}) + 3\dot{d}_i \boldsymbol{\omega}_i \times [\boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i-1})]
\end{aligned} \tag{A.26}$$

Finally, we can rearrange the terms, collect some similar expressions, and describe $\mathbf{s}$ with respect to the base frame, obtaining (2.33).

As done previously for the linear jerk, the linear snap $\mathbf{s}_{c_i}$ can be computed from $\mathbf{s}_i$. Equation (2.34) is obtained from (2.33), by simply letting i instead of $i-1$, c_i instead of i , and considering that frames i and c_i are fixed to the same link and parallel to each other, implying $\dot{d}_{c_i}$, $\ddot{d}_{c_i}$, $\ddot{d}_{c_i}$ and $\ddot{d}_{c_i}$ are all equal to zero.

A.2 Backward recursion

A.2.1 Forces and moments acting on the centers of mass

Equations (2.36) and (2.37) are trivially obtained by differentiating (A.6) as

$$\dot{\mathbf{F}}_i = m_i \dot{\mathbf{j}}_{c_i} \tag{A.27}$$

$$\ddot{\mathbf{F}}_i = m_i \mathbf{s}_{c_i} \tag{A.28}$$

and referring the resulting equations to frame i .

The derivation of the moment derivatives requires more attention. We begin from Euler's law of motion

$$\mathbf{N}_i = \frac{d}{dt}(\mathbf{I}_i \boldsymbol{\omega}_i). \tag{A.29}$$

Here, the inertia tensor is not constant. Since

$$\mathbf{I}_i = {}^0\mathbf{R}_i {}^i\mathbf{I}_i {}^i\mathbf{R}_0 \tag{A.30}$$

APPENDICES

its time derivative, remembering (A.13), can be obtained as

$$\begin{aligned}
\dot{\mathbf{I}}_i &= {}^0\dot{\mathbf{R}}_i {}^i\mathbf{I}_i {}^i\mathbf{R}_0 + {}^0\mathbf{R}_i {}^i\dot{\mathbf{I}}_i {}^i\mathbf{R}_0 + {}^0\mathbf{R}_i {}^i\mathbf{I}_i {}^i\dot{\mathbf{R}}_0 \\
&= \dot{\mathbf{R}}_i {}^i\mathbf{I}_i {}^i\mathbf{R}_0 + \mathbf{R}_i {}^i\mathbf{I}_i \dot{\mathbf{R}}_i^T \\
&= \mathbf{S}(\boldsymbol{\omega}_i) \mathbf{R}_i {}^i\mathbf{I}_i {}^i\mathbf{R}_0 + \mathbf{R}_i {}^i\mathbf{I}_i \mathbf{R}_i^T \mathbf{S}^T(\boldsymbol{\omega}_i) \\
&= \mathbf{S}(\boldsymbol{\omega}_i) \mathbf{I}_i - \mathbf{I}_i \mathbf{S}(\boldsymbol{\omega}_i).
\end{aligned} \tag{A.31}$$

It can be verified that, from (A.29) and (A.31), it is possible to derive (A.7). Differentiating this last equation, we obtain

$$\dot{\mathbf{N}}_i = \dot{\mathbf{I}}_i \boldsymbol{\gamma}_i + \mathbf{I}_i \boldsymbol{\iota}_i + \boldsymbol{\gamma}_i \times \mathbf{I}_i \boldsymbol{\omega}_i + \boldsymbol{\omega}_i \times \frac{d}{dt}(\mathbf{I}_i \boldsymbol{\omega}_i). \tag{A.32}$$

Taking again (A.29) and (A.31) into account, we can rewrite (A.32) as

$$\begin{aligned}
\dot{\mathbf{N}}_i &= \mathbf{S}(\boldsymbol{\omega}_i) \mathbf{I}_i \boldsymbol{\gamma}_i - \mathbf{I}_i \mathbf{S}(\boldsymbol{\omega}_i) \boldsymbol{\gamma}_i + \mathbf{I}_i \boldsymbol{\iota}_i + \boldsymbol{\gamma}_i \times \mathbf{I}_i \boldsymbol{\omega}_i + \boldsymbol{\omega}_i \times \mathbf{N}_i \\
&= \boldsymbol{\omega}_i \times \mathbf{I}_i \boldsymbol{\gamma}_i + \mathbf{I}_i (\boldsymbol{\gamma}_i \times \boldsymbol{\omega}_i) + \mathbf{I}_i \boldsymbol{\iota}_i + \boldsymbol{\gamma}_i \times \mathbf{I}_i \boldsymbol{\omega}_i + \boldsymbol{\omega}_i \times \mathbf{N}_i.
\end{aligned} \tag{A.33}$$

Representing $\dot{\mathbf{N}}_i$ with respect to the i -th frame and grouping some similar terms, we obtain (2.38).

Differentiating (A.33), using (A.29) and (A.31) again and referring everything to the i -th frame, we obtain

$$\begin{aligned}
{}^i\ddot{\mathbf{N}}_i &= {}^i\boldsymbol{\gamma}_i \times {}^i\mathbf{I}_i {}^i\boldsymbol{\gamma}_i + {}^i\boldsymbol{\omega}_i \times [\mathbf{S}({}^i\boldsymbol{\omega}_i) {}^i\mathbf{I}_i - {}^i\mathbf{I}_i \mathbf{S}({}^i\boldsymbol{\omega}_i)] {}^i\boldsymbol{\gamma}_i \\
&\quad + {}^i\boldsymbol{\omega}_i \times {}^i\mathbf{I}_i {}^i\boldsymbol{\iota}_i + [\mathbf{S}({}^i\boldsymbol{\omega}_i) {}^i\mathbf{I}_i] ({}^i\boldsymbol{\gamma}_i \times {}^i\boldsymbol{\omega}_i) \\
&\quad - [{}^i\mathbf{I}_i \mathbf{S}({}^i\boldsymbol{\omega}_i)] ({}^i\boldsymbol{\gamma}_i \times {}^i\boldsymbol{\omega}_i) + {}^i\mathbf{I}_i ({}^i\boldsymbol{\iota}_i \times {}^i\boldsymbol{\omega}_i) \\
&\quad + [\mathbf{S}({}^i\boldsymbol{\omega}_i) {}^i\mathbf{I}_i - {}^i\mathbf{I}_i \mathbf{S}({}^i\boldsymbol{\omega}_i)] {}^i\boldsymbol{\iota}_i + {}^i\mathbf{I}_i {}^i\boldsymbol{\varsigma}_i \\
&\quad + {}^i\boldsymbol{\iota}_i \times {}^i\mathbf{I}_i {}^i\boldsymbol{\omega}_i + 2({}^i\boldsymbol{\gamma}_i \times {}^i\mathbf{N}_i) + {}^i\boldsymbol{\omega}_i \times {}^i\dot{\mathbf{N}}_i \\
&= {}^i\boldsymbol{\gamma}_i \times {}^i\mathbf{I}_i {}^i\boldsymbol{\gamma}_i + {}^i\boldsymbol{\omega}_i \times ({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{I}_i {}^i\boldsymbol{\gamma}_i) \\
&\quad - {}^i\boldsymbol{\omega}_i \times {}^i\mathbf{I}_i ({}^i\boldsymbol{\omega}_i \times {}^i\boldsymbol{\gamma}_i) + {}^i\boldsymbol{\omega}_i \times {}^i\mathbf{I}_i {}^i\boldsymbol{\iota}_i \\
&\quad + {}^i\boldsymbol{\omega}_i \times {}^i\mathbf{I}_i ({}^i\boldsymbol{\gamma}_i \times {}^i\boldsymbol{\omega}_i) - {}^i\mathbf{I}_i [{}^i\boldsymbol{\omega}_i \times ({}^i\boldsymbol{\gamma}_i \times {}^i\boldsymbol{\omega}_i)] \\
&\quad + {}^i\mathbf{I}_i ({}^i\boldsymbol{\iota}_i \times {}^i\boldsymbol{\omega}_i) + {}^i\boldsymbol{\omega}_i \times {}^i\mathbf{I}_i {}^i\boldsymbol{\iota}_i - {}^i\mathbf{I}_i ({}^i\boldsymbol{\omega}_i \times {}^i\boldsymbol{\iota}_i) \\
&\quad + {}^i\mathbf{I}_i {}^i\boldsymbol{\varsigma}_i + {}^i\boldsymbol{\iota}_i \times {}^i\mathbf{I}_i {}^i\boldsymbol{\omega}_i + 2({}^i\boldsymbol{\gamma}_i \times {}^i\mathbf{N}_i) + {}^i\boldsymbol{\omega}_i \times {}^i\dot{\mathbf{N}}_i
\end{aligned} \tag{A.34}$$

The final expression is obtained by collecting similar terms.

A.2.2 Transmitted forces and moments

Again, equations (2.42) and (2.43) are obtained trivially by differentiating (A.8)

$$\dot{\mathbf{f}}_i = \dot{\mathbf{f}}_{i+1} + \dot{\mathbf{F}}_i \quad (\text{A.35})$$

$$\ddot{\mathbf{f}}_i = \ddot{\mathbf{f}}_{i+1} + \ddot{\mathbf{F}}_i \quad (\text{A.36})$$

and referring everything to frame i .

We now study the moment time derivatives. Differentiating (A.9):

$$\dot{\mathbf{n}}_i = \dot{\mathbf{n}}_{i+1} + \dot{\mathbf{p}}_{c_i,i} \times \mathbf{F}_i + \mathbf{p}_{c_i,i} \times \dot{\mathbf{F}}_i + \dot{\mathbf{p}}_{i,i-1} \times \mathbf{f}_i + \mathbf{p}_{i,i-1} \times \dot{\mathbf{f}}_i + \dot{\mathbf{N}}_i. \quad (\text{A.37})$$

Now, since $\mathbf{p}_{c_i,i} = \mathbf{R}_i^i \mathbf{p}_{c_i,i}$, with ${}^i\mathbf{p}_{c_i,i}$ constant, we can write

$$\dot{\mathbf{p}}_{c_i,i} = \dot{\mathbf{R}}_i^i \mathbf{p}_{c_i,i} = \mathbf{S}(\boldsymbol{\omega}_i) \mathbf{R}_i^i \mathbf{p}_{c_i,i} = \boldsymbol{\omega}_i \times \mathbf{p}_{c_i,i} \quad (\text{A.38})$$

therefore, using (A.21) and (A.38), we can rewrite (A.37) as

$$\begin{aligned} \dot{\mathbf{n}}_i = & \dot{\mathbf{n}}_{i+1} + (\boldsymbol{\omega}_i \times \mathbf{p}_{c_i,i}) \times \mathbf{F}_i + \mathbf{p}_{c_i,i} \times \dot{\mathbf{F}}_i \\ & + (\boldsymbol{\omega}_i \times \mathbf{p}_{i,i-1} + \dot{d}_i \hat{\mathbf{z}}_{i-1}) \times \mathbf{f}_i + \mathbf{p}_{i,i-1} \times \dot{\mathbf{f}}_i + \dot{\mathbf{N}}_i \end{aligned} \quad (\text{A.39})$$

whence we obtain (2.45) by referring everything to frame i .

Differentiating (A.39) with the help of (A.21) and (A.38):

$$\begin{aligned} \ddot{\mathbf{n}}_i = & \ddot{\mathbf{n}}_{i+1} + [\boldsymbol{\gamma}_i \times \mathbf{p}_{c_i,i} + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{c_i,i})] \times \mathbf{F}_i \\ & + (\boldsymbol{\omega}_i \times \mathbf{p}_{c_i,i}) \times \dot{\mathbf{F}}_i + (\boldsymbol{\omega}_i \times \mathbf{p}_{c_i,i}) \times \ddot{\mathbf{F}}_i + \mathbf{p}_{c_i,i} \times \ddot{\mathbf{F}}_i \\ & + [\boldsymbol{\gamma}_i \times \mathbf{p}_{i,i-1} + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{i,i-1} + \dot{d}_i \hat{\mathbf{z}}_{i-1}) + \ddot{d}_i \hat{\mathbf{z}}_{i-1} \\ & + \dot{d}_i \boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i-1}] \times \mathbf{f}_i + (\boldsymbol{\omega}_i \times \mathbf{p}_{i,i-1} + \dot{d}_i \hat{\mathbf{z}}_{i-1}) \times \dot{\mathbf{f}}_i \\ & + (\boldsymbol{\omega}_i \times \mathbf{p}_{i,i-1} + \dot{d}_i \hat{\mathbf{z}}_{i-1}) \times \ddot{\mathbf{f}}_i + \mathbf{p}_{i,i-1} \times \ddot{\mathbf{f}}_i + \ddot{\mathbf{N}}_i \end{aligned} \quad (\text{A.40})$$

from which it is relatively easy to obtain (2.46).

A.2.3 Elastic torques and motor torques

To obtain the time derivative of the elastic torque, we start from the case of a revolute joint

$$\boldsymbol{\tau}_{ei} = \mathbf{n}_i^T \hat{\mathbf{z}}_{i-1} \quad (\text{A.41})$$

Differentiating (A.41) we obtain

$$\dot{\boldsymbol{\tau}}_{ei} = \dot{\mathbf{n}}_i^T \hat{\mathbf{z}}_{i-1} + \mathbf{n}_i^T \dot{\hat{\mathbf{z}}}_{i-1} = \dot{\mathbf{n}}_i^T \hat{\mathbf{z}}_{i-1} + \mathbf{n}_i^T (\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i-1}). \quad (\text{A.42})$$

APPENDICES

where (A.22) was used. Now, we can use the well-known property of the scalar triple product

$$\mathbf{a}^T(\mathbf{b} \times \mathbf{c}) = (\mathbf{a} \times \mathbf{b})^T \mathbf{c} \quad \forall \mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbb{R}^3 \quad (\text{A.43})$$

in order to rewrite (A.42) as

$$\dot{\tau}_{ei} = \dot{\mathbf{n}}_i^T \hat{\mathbf{z}}_{i-1} + (\mathbf{n}_i \times \boldsymbol{\omega}_i)^T \hat{\mathbf{z}}_{i-1} = (\dot{\mathbf{n}}_i + \mathbf{n}_i \times \boldsymbol{\omega}_i)^T \hat{\mathbf{z}}_{i-1} \quad (\text{A.44})$$

where expressing all quantities into frame i will return the expression for (2.48), in the case of revolute joints.

Finally, differentiating (A.44) and using (A.22) we obtain

$$\begin{aligned} \ddot{\tau}_{ei} &= (\ddot{\mathbf{n}}_i + \dot{\mathbf{n}}_i \times \boldsymbol{\omega}_i + \mathbf{n}_i \times \boldsymbol{\gamma}_i)^T \hat{\mathbf{z}}_{i-1} \\ &\quad + (\dot{\mathbf{n}}_i + \mathbf{n}_i \times \boldsymbol{\omega}_i)^T (\boldsymbol{\omega}_i \times \hat{\mathbf{z}}_{i-1}) \\ &= (\ddot{\mathbf{n}}_i + \dot{\mathbf{n}}_i \times \boldsymbol{\omega}_i + \mathbf{n}_i \times \boldsymbol{\gamma}_i)^T \hat{\mathbf{z}}_{i-1} \\ &\quad + [\dot{\mathbf{n}}_i \times \boldsymbol{\omega}_i + (\mathbf{n}_i \times \boldsymbol{\omega}_i) \times \boldsymbol{\omega}_i]^T \hat{\mathbf{z}}_{i-1} \end{aligned} \quad (\text{A.45})$$

Collecting for $\hat{\mathbf{z}}_{i-1}$ and expressing all quantities into frame i will return (2.49) for revolute joints.

For a prismatic joint, we start from

$$\tau_{ei} = \mathbf{f}_i^T \hat{\mathbf{z}}_{i-1}. \quad (\text{A.46})$$

The derivation of the higher-order formulas will then be exactly on the lines as that for the case of a revolute joint.

The final expressions for the motor velocities, accelerations, and torques, i.e. equations (2.50), (2.51), and (2.52), are obvious from (2.15), (2.16) and (2.17), since $\boldsymbol{\Sigma}$ and $\mathbf{B}$ are diagonal.

A.3 Modified Denavit-Hartenberg notation

The derivation of EJNEA for Craig's modified Denavit-Hartenberg notation starts from the equations

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} + \dot{\theta}_i \hat{\mathbf{z}}_i \quad (\text{A.47})$$

$$\boldsymbol{\gamma}_i = \boldsymbol{\gamma}_{i-1} + \ddot{\theta}_i \hat{\mathbf{z}}_i + \boldsymbol{\omega}_{i-1} \times \dot{\theta}_i \hat{\mathbf{z}}_i \quad (\text{A.48})$$

$$\mathbf{v}_i = \mathbf{v}_{i-1} + \boldsymbol{\omega}_{i-1} \times \mathbf{p}_{i,i-1} + \dot{d}_i \hat{\mathbf{z}}_i \quad (\text{A.49})$$

$$\begin{aligned} \mathbf{a}_i &= \mathbf{a}_{i-1} + \boldsymbol{\omega}_{i-1} \times (\boldsymbol{\omega}_{i-1} \times \mathbf{p}_{i,i-1}) \\ &\quad + \boldsymbol{\gamma}_{i-1} \times \mathbf{p}_{i,i-1} + \ddot{d}_i \hat{\mathbf{z}}_i + 2\dot{d}_i (\boldsymbol{\omega}_{i-1} \times \hat{\mathbf{z}}_i) \end{aligned} \quad (\text{A.50})$$

APPENDIX A. DERIVATION OF EJNEA

$$\mathbf{a}_{c_i} = \mathbf{a}_i + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \mathbf{p}_{c_i,i}) + \boldsymbol{\gamma}_i \times \mathbf{p}_{c_i,i} \quad (\text{A.51})$$

$$\mathbf{F}_i = m_i \mathbf{a}_{c_i} \quad (\text{A.52})$$

$$\mathbf{N}_i = \mathbf{I}_i \boldsymbol{\gamma}_i + \boldsymbol{\omega}_i \times (\mathbf{I}_i \boldsymbol{\omega}_i) \quad (\text{A.53})$$

$$\mathbf{f}_i = \mathbf{f}_{i+1} + \mathbf{F}_i \quad (\text{A.54})$$

$$\mathbf{n}_i = \mathbf{n}_{i+1} + \mathbf{p}_{c_i,i} \times \mathbf{F}_i + \mathbf{p}_{i+1,i} \times \mathbf{f}_{i+1} + \mathbf{N}_i \quad (\text{A.55})$$

$$\tau_{ei} = \begin{cases} \mathbf{n}_i^T \hat{\mathbf{z}}_i & \text{if joint } i \text{ is revolute} \\ \mathbf{f}_i^T \hat{\mathbf{z}}_i & \text{if joint } i \text{ is prismatic} \end{cases} \quad (\text{A.56})$$

The derivation is along the lines of that for standard EJNEA. A proof of the third-order dynamic equations can be found in [37], where these formulas first appeared. Notice equations (A.51)–(A.54) are identical to (A.5)–(A.8), and therefore their time differentiation follows exactly the same steps as in the standard case.

APPENDICES

Appendix B

Nonlinear Spring Solutions

B.1 Cubic spring

In the following, we derive necessary and sufficient conditions for solving

$$(\cos \xi_i + \sin \xi_i) + \frac{\sigma_i - 2K_i}{3K_i}(\cos^3 \xi_i + \sin^3 \xi_i) = \frac{\tau_{ei}}{K_i R_i} \quad (\text{B.1})$$

where $\sigma_i \geq 2K_i$, $K_i \geq 0$ and

$$R_i = \sqrt{\frac{\sigma_i - 2K_i}{3K_i}} > 0. \quad (\text{B.2})$$

For convenience, we define

$$a_i = \frac{\sigma_i - 2K_i}{3K_i} > 0 \quad b_i = \frac{\tau_{ei}}{K_i R_i} \quad (\text{B.3})$$

and we drop subscript i . Then, the object of the study will be

$$f(\xi) = (\cos \xi + \sin \xi) + a(\cos^3 \xi + \sin^3 \xi) = b. \quad (\text{B.4})$$

First of all, we notice f is a bounded, continuous function of ξ . Therefore, it is possible to find a global minimum f_m and a global maximum f_M . By the intermediate value theorem, if $f_m \leq b \leq f_M$, then it is possible to find a value of ξ such that $f(\xi) = b$. The converse is also true. Therefore, a solution exists if and only if b falls within these bounds. Inspection of f reveals that, for any $c \in \mathbb{R}$, if $f(\xi) = c$, then $f(\xi + \pi) = -c$. From this, it is easy to understand that $f_m = -f_M$. Therefore, the bounds on b can be re-expressed as $|b| \leq f_M$, with positive f_M . Now, it is only matter of finding f_M . Since f is periodic with period 2π , the search can be restricted to $\xi \in (-\pi, \pi]$. The bounds on b will then be translated as upper bounds to $|\tau_e|$.

APPENDICES

The search for f_M can start by finding the stationary points. Differentiating (B.4) with respect to ξ we obtain

$$\begin{aligned} f'(\xi) &= (-\sin \xi + \cos \xi) + a(-3 \cos^2 \xi \sin \xi + 3 \sin^2 \xi \cos \xi) \\ &= (\cos \xi - \sin \xi)(1 - 3a \sin \xi \cos \xi) \\ &= (\cos \xi - \sin \xi)(1 - \frac{3}{2}a \sin(2\xi)) = 0 \end{aligned} \quad (\text{B.5})$$

There are two possible kinds of solutions. The first one is obtained by solving

$$(\cos \xi - \sin \xi) = 0. \quad (\text{B.6})$$

This equation has two solutions. Only one of them leads to a positive value of $f(\xi)$, which is

$$\xi_1 = \frac{\pi}{4}. \quad (\text{B.7})$$

Plugging it into (B.4), we obtain

$$f(\xi_1) = \sqrt{2} + a\frac{\sqrt{2}}{2} = \sqrt{2} \left(\frac{2}{3} + \frac{\sigma}{6K} \right) \quad (\text{B.8})$$

The second kind of solutions is found by solving

$$1 - \frac{3}{2}a \sin(2\xi) = 0 \quad (\text{B.9})$$

$$\sin(2\xi) = \frac{2}{3a}. \quad (\text{B.10})$$

This equation is only solvable if $a \geq \frac{2}{3}$, i.e. $\sigma \geq 4K$. Thus, we see that, if $\sigma < 4K$, then ξ_1 is the only positive stationary point; therefore, in this case, it must be the global maximum point.

If instead $\sigma \geq 4K$, there are two possible solutions

$$\xi_2 = \frac{1}{2} \arcsin \left(\frac{2}{3a} \right) \quad (\text{B.11})$$

$$\xi_3 = \frac{\pi}{2} - \frac{1}{2} \arcsin \left(\frac{2}{3a} \right) = \frac{\pi}{2} - \xi_2. \quad (\text{B.12})$$

It is straightforward to see that

$$\sigma = 4K \implies a = \frac{2}{3} \implies \xi_2 = \xi_3 = \frac{\pi}{4} = \xi_1 \quad (\text{B.13})$$

and also that, for all values of $\sigma \geq 4K$

$$f(\xi_3) = f(\xi_2). \quad (\text{B.14})$$

Therefore, we will concentrate on ξ_2 only to compute the value of $f(\xi)$ at both ξ_2 and ξ_3 . Let us rewrite f as

$$\begin{aligned} f(\xi) &= (\cos \xi + \sin \xi) + a(\cos \xi + \sin \xi)(\cos^2 \xi - \cos \xi \sin \xi + \sin^2 \xi) \\ &= (\cos \xi + \sin \xi) \left(1 + a \left(1 - \frac{1}{2} \sin(2\xi) \right) \right). \end{aligned} \quad (\text{B.15})$$

APPENDIX B. NONLINEAR SPRING SOLUTIONS

Now, from (B.10)

$$\begin{aligned}
 f(\xi_2) &= (\cos \xi_2 + \sin \xi_2) \left(1 + a \left(1 - \frac{1}{3a} \right) \right) \\
 &= (\cos \xi_2 + \sin \xi_2) \left(a + \frac{2}{3} \right) \\
 &= (\cos \xi_2 + \sin \xi_2) \frac{\sigma}{3K} \\
 &= \sqrt{2} \cos \left(\xi_2 - \frac{\pi}{4} \right) \frac{\sigma}{3K}
 \end{aligned} \tag{B.16}$$

Now

$$\cos \left(\xi_2 - \frac{\pi}{4} \right) = \cos \left(\frac{1}{2} \left(a \sin \left(\frac{2}{3a} \right) - \frac{\pi}{2} \right) \right) = \pm \sqrt{\frac{1 + \cos \left(a \sin \left(\frac{2}{3a} \right) - \frac{\pi}{2} \right)}{2}} \tag{B.17}$$

which becomes (discarding the negative solution)

$$\cos \left(\xi_2 - \frac{\pi}{4} \right) = \sqrt{\frac{1 + \sin \left(a \sin \left(\frac{2}{3a} \right) \right)}{2}} = \sqrt{\frac{1 + \frac{2}{3a}}{2}} = \sqrt{\frac{\sigma}{2(\sigma - 2K)}} \tag{B.18}$$

therefore

$$f(\xi_2) = \sqrt{2} \sqrt{\frac{\sigma}{2(\sigma - 2K)}} \frac{\sigma}{3K} = \frac{\sigma}{3K} \sqrt{\frac{\sigma}{\sigma - 2K}} \tag{B.19}$$

Important notice: even though (B.19) can be computed for all $\sigma > 2K$, in reality, for this to be an extremum, it must be $\sigma \geq 4K$, otherwise its value is meaningless.

Now, we will compare $f(\xi_1)$ and $f(\xi_2)$ by finding the conditions such that $f(\xi_1) \leq f(\xi_2)$.

$$\sqrt{2} \left(\frac{2}{3} + \frac{\sigma}{6K} \right) \leq \sqrt{2} \sqrt{\frac{\sigma}{2(\sigma - 2K)}} \frac{\sigma}{3K} \tag{B.20}$$

$$\frac{2}{3} + \frac{\sigma}{6K} \leq \sqrt{\frac{\sigma}{2(\sigma - 2K)}} \frac{\sigma}{3K} \tag{B.21}$$

$$\left(\frac{1}{2} + \frac{2K}{\sigma} \right) \frac{\sigma}{3K} \leq \sqrt{\frac{1}{2 \left(1 - \frac{2K}{\sigma} \right)}} \frac{\sigma}{3K} \tag{B.22}$$

$$\frac{1}{2} + \frac{2K}{\sigma} \leq \sqrt{\frac{1}{2 \left(1 - \frac{2K}{\sigma} \right)}} \tag{B.23}$$

where the last simplification is possible since $\frac{\sigma}{3K} \geq 0$.

Now, we define the auxiliary variable

$$x = \frac{2K}{\sigma} \tag{B.24}$$

APPENDICES

leading to

$$\frac{1}{2} + x \leq \sqrt{\frac{1}{2(1-x)}} \quad (\text{B.25})$$

$$1 + 2x \leq \sqrt{\frac{2}{1-x}} \quad (\text{B.26})$$

Since $\sigma \geq 4K$, then

$$0 < x \leq \frac{1}{2} \quad (\text{B.27})$$

This implies both sides of (B.26) are positive, so that they can be squared maintaining the inequality. Further, $(1 - x)$ is positive too. Therefore

$$1 + 4x + 4x^2 \leq \frac{2}{1-x} \quad (\text{B.28})$$

$$(1 + 4x + 4x^2)(1 - x) \leq 2 \quad (\text{B.29})$$

$$4x^3 - 3x + 1 \geq 0 \quad (\text{B.30})$$

$$(x + 1)(2x - 1)^2 \geq 0 \quad (\text{B.31})$$

The solution of this equation is

$$x \geq -1 \quad (\text{B.32})$$

Therefore, all admissible values of x satisfy the inequality. This implies that, for $\sigma \geq 4K$, it is always $f(\xi_2) \geq f(\xi_1)$. The inequality holds strictly for all values of σ , except for $\sigma = 4K$, where we have already seen that $\xi_1 = \xi_2 = \xi_3$.

We have found that, for $\sigma > 4K$, $f(\xi_1)$ is a local extremum, while $f(\xi_2) = f(\xi_3) > f(\xi_1)$ are also local extrema. There are no other positive extrema. There is only one possible explanation to this: ξ_2 and ξ_3 are two global maximum points, while ξ_1 is a local minimum point lying in between.

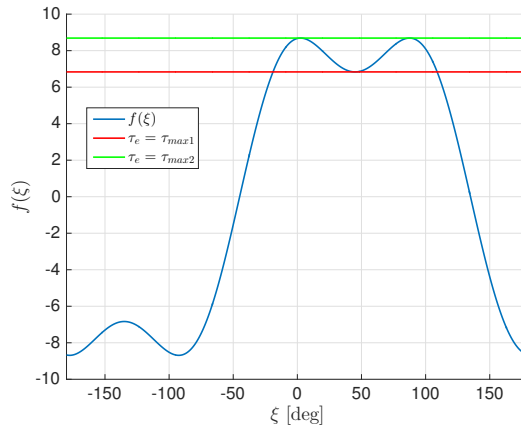


Figure B.1 The values of $f(\xi)$ for $K = 400$, $K_c = 2000$, and $\sigma = 10000 > 4K$, with the two extrema in evidence.

APPENDIX B. NONLINEAR SPRING SOLUTIONS

To summarize, for $\sigma \leq 4K$, then $f(\xi_1)$ is a global maximum. For $\sigma > 4K$, $f(\xi_1)$ is a local minimum, while $f(\xi_2) = f(\xi_3)$ is a global maximum.

Finally, for $k = 1, 2$, the significant extrema for $|\tau_e|$ are defined as

$$\frac{|\tau_e|_{maxk}}{KR} = f(\xi_k) \quad (\text{B.33})$$

so that

$$|\tau_e|_{max1} = \sqrt{2}KR \left(\frac{2}{3} + \frac{\sigma}{6K} \right) = \sqrt{2}KR \frac{4K+\sigma}{6K} = \frac{\sqrt{2}}{6}R(4K + \sigma) \quad (\text{B.34})$$

while

$$|\tau_e|_{max2} = KR \frac{\sigma}{3K} \sqrt{\frac{\sigma}{\sigma-2K}} = \frac{\sigma}{3} \sqrt{\frac{\sigma-2K}{3K} \frac{\sigma}{\sigma-2K}} = \frac{\sigma}{3} \sqrt{\frac{\sigma}{3K}} = \sqrt{\frac{1}{K} \left(\frac{\sigma}{3} \right)^3} \quad (\text{B.35})$$

which concludes our demonstration. To provide some visual intuition on the solutions, Fig. B.1 depicts the values of $f(\xi)$ when $\sigma > 4K$.

B.2 Quadratic spring

In the following, we seek solutions to the following system of equations

$$\tau_{ei} = K_i(\phi_{1,i} + \phi_{2,i}) + K_{qi}(\text{sign}(\phi_{1,i})\phi_{1,i}^2 + \text{sign}(\phi_{2,i})\phi_{2,i}^2) \quad (\text{B.36a})$$

$$\sigma_i = 2K_i + 2K_{qi}(|\phi_{1,i}| + |\phi_{2,i}|). \quad (\text{B.36b})$$

with $\sigma_i > 2K_i$, $K_i > 0$, $K_{qi} > 0$. To solve (B.36), we will first make some assumptions about the signs of the solutions, and we will later check whether the obtained values agree with the given assumption. From now on, we drop subscript i , for notational simplicity.

In the first place, let's assume the two deflections have opposite sign. This transforms (B.36) to

$$\tau_e = K(\phi_1 + \phi_2) \pm K_q(\phi_2^2 - \phi_1^2) \quad (\text{B.37a})$$

$$\sigma = 2K \pm 2K_q(\phi_2 - \phi_1). \quad (\text{B.37b})$$

From (B.37b), we can write

$$\phi_2 - \phi_1 = \pm \frac{\sigma - 2K}{2K_q}. \quad (\text{B.38})$$

On the other hand, (B.37a) can be rewritten as

$$\tau_e = K(\phi_1 + \phi_2) \pm K_q(\phi_1 + \phi_2)(\phi_2 - \phi_1) \quad (\text{B.39})$$

APPENDICES

therefore, substituting (B.38) in (B.39)

$$\tau_e = K(\phi_1 + \phi_2) + (\phi_1 + \phi_2) \frac{\sigma - 2K}{2} = \frac{\sigma}{2}(\phi_1 + \phi_2). \quad (\text{B.40})$$

For $\sigma = 0$, then the left-hand side of (B.40) is zero, and therefore there are no solutions unless $\tau_e = 0$, in which case there is an infinite number of them. Excluding these cases, we can write

$$\phi_2 = \frac{2\tau_e}{\sigma} - \phi_1 \quad (\text{B.41})$$

which can be replaced in (B.38)

$$\frac{2\tau_e}{\sigma} - 2\phi_1 = \pm \frac{\sigma - 2K}{2K_q} \quad (\text{B.42})$$

which is easily solved as

$$\phi_1 = \frac{\tau_e}{\sigma} \mp \frac{\sigma - 2K}{4K_q}. \quad (\text{B.43})$$

Finally, the obtained solution can be replaced in (B.41) to obtain

$$\phi_2 = \frac{\tau_e}{\sigma} \pm \frac{\sigma - 2K}{4K_q}. \quad (\text{B.44})$$

For these solutions to be valid, it must be either $\phi_1 \geq 0$ and $\phi_2 \leq 0$ or $\phi_1 \leq 0$ and $\phi_2 \geq 0$. By solving (B.43) and (B.44) for these inequalities and assuming $\sigma \geq 2K_i \geq 0$, it can be verified that these conditions only hold when

$$|\tau_e| \leq \frac{\sigma(\sigma - 2K)}{4K_q} \quad (\text{B.45})$$

and that at least one between ϕ_1 and ϕ_2 is equal to zero when the upper bound for $|\tau_e|$ is reached.

Now, let's assume the two deflections have the same sign. System (B.36) can then be written as

$$\tau_e = K(\phi_1 + \phi_2) \pm K_q(\phi_1^2 + \phi_2^2) \quad (\text{B.46a})$$

$$\sigma = 2K \pm 2K_q(\phi_1 + \phi_2) \quad (\text{B.46b})$$

where the plus sign is selected when both deflections are positive, and the minus when both are negative. From (B.46b) we can write

$$\phi_1 + \phi_2 = \pm \frac{\sigma - 2K}{2K_q} = \pm a \quad (\text{B.47})$$

APPENDIX B. NONLINEAR SPRING SOLUTIONS

which is replaced in (B.46a)

$$\tau_e = \pm (Ka + K_q(\phi_1^2 + \phi_2^2)) \quad (\text{B.48})$$

yielding

$$\phi_1^2 + \phi_2^2 = \frac{\pm \tau_e - Ka}{K_q}. \quad (\text{B.49})$$

Since this quantity must be positive, this imposes the constraint

$$\pm \tau_e \geq Ka. \quad (\text{B.50})$$

Further, we can solve (B.47) for ϕ_2

$$\phi_2 = \pm a - \phi_1 \quad (\text{B.51})$$

and replace this value in (B.49) to obtain the quadratic equation

$$2\phi_1^2 \mp 2a\phi_1 + \left(a^2 + \frac{Ka}{K_q} \mp \frac{\tau_e}{K_q}\right) = 0. \quad (\text{B.52})$$

For this equation to have a solution, the determinant

$$\Delta = -a^2 - 2\frac{Ka}{K_q} \pm 2\frac{\tau_e}{K_q} = -\frac{\sigma^2 - 4K^2}{4K_q^2} \pm 2\frac{\tau_e}{K_q} \quad (\text{B.53})$$

must not be negative. This translates to

$$\pm \tau_e \geq \frac{K_q}{2}a^2 + Ka = \frac{\sigma^2 - 4K^2}{8K_q}. \quad (\text{B.54})$$

This constraint is more stringent than (B.50), since

$$\frac{\sigma^2 - 4K^2}{8K_q} = \frac{\sigma + 2K}{4} \frac{\sigma - 2K}{2K_q} \geq Ka \quad (\text{B.55})$$

since $\sigma \geq 2K$. Therefore, we can forget about (B.50) and only refer to (B.54). Solving (B.52) and using (B.51), one pair of solutions is obtained as

$$\phi_1 = \pm \frac{a}{2} + \sqrt{\frac{\Delta}{4}} \quad (\text{B.56a})$$

$$\phi_2 = \pm \frac{a}{2} - \sqrt{\frac{\Delta}{4}} \quad (\text{B.56b})$$

the other pair consisting in the same values being swapped.

APPENDICES

Now, we need to check when these two values have the same sign. Both for positive and for negative deflections, the condition turns out to be the same

$$a^2 \geq \Delta \quad (\text{B.57})$$

which becomes

$$\pm \tau_e \leq \frac{\sigma(\sigma - 2K)}{4K_q} \quad (\text{B.58})$$

which is the same as (B.45), yielding a single condition for solutions to exist. Also, this constraint is always compatible with (B.54), since

$$\frac{\sigma^2 - 4K^2}{8K_q} = \frac{\sigma + 2K}{2} \frac{\sigma - 2K}{4K_q} \quad (\text{B.59})$$

and $\sigma + 2K \leq 2\sigma$ for $\sigma \geq 2K$. Summing up, when

$$\frac{\sigma_i^2 - 4K_i^2}{8K_{qi}} \leq \tau_{ei} \leq \frac{\sigma_i(\sigma_i - 2K_i)}{4K_{qi}} \quad (\text{B.60})$$

then two positive pairs of solutions are available

$$\phi_{1,i} = \frac{1}{4K_{qi}} \left(\sigma_i - 2K_i \pm \sqrt{4K_i^2 - \sigma_i^2 + 8K_{qi}\tau_{ei}} \right) \quad (\text{B.61a})$$

$$\phi_{2,i} = \frac{1}{4K_{qi}} \left(\sigma_i - 2K_i \mp \sqrt{4K_i^2 - \sigma_i^2 + 8K_{qi}\tau_{ei}} \right) \quad (\text{B.61b})$$

while when

$$-\frac{\sigma_i(\sigma_i - 2K_i)}{4K_{qi}} \leq \tau_{ei} \leq -\frac{\sigma_i^2 - 4K_i^2}{8K_{qi}} \quad (\text{B.62})$$

then there are two extra negative pairs of solutions

$$\phi_{1,i} = \frac{1}{4K_{qi}} \left(2K_i - \sigma_i \pm \sqrt{4K_i^2 - \sigma_i^2 - 8K_{qi}\tau_{ei}} \right) \quad (\text{B.63a})$$

$$\phi_{2,i} = \frac{1}{4K_{qi}} \left(2K_i - \sigma_i \mp \sqrt{4K_i^2 - \sigma_i^2 - 8K_{qi}\tau_{ei}} \right). \quad (\text{B.63b})$$

Finally, it can be observed that, when the upper bound of (B.60) is reached, then the values of (B.61) are the same as (B.43) and (B.44), while when the upper bound of (B.62) is reached, then the same is true for the values of (B.63). Otherwise, the values are different.

Appendix C

Dynamic Model of SEA 2R robot

Details on the physical parameters of the robot in Sec. 4.1.1 and of its Lagrangian model. The robot and its physical parameters are shown in Fig. C.1.

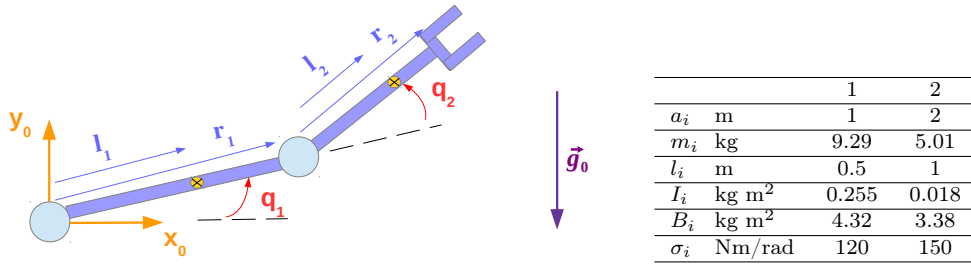


Figure C.1 Planar 2R robot used for validation

The values in Fig. C.1 are easily converted to Denavit-Hartenberg parameters, shown in Tab. C.1.

	1	2
α_i	0	0
a_i	r_1	r_2
d_i	0	0
θ_i	q_1	q_2
${}^i\mathbf{p}_{cx_i,i}$	$l_1 - r_1$	$l_2 - r_2$
${}^i\mathbf{p}_{cy_i,i}$	0	0
${}^i\mathbf{I}_{zz_i}$	I_1	I_2

(a) Standard

	1	2
α_{i-1}	0	0
a_{i-1}	0	r_1
d_i	0	0
θ_i	q_1	q_2
${}^i\mathbf{p}_{cx_i,i}$	l_1	l_2
${}^i\mathbf{p}_{cy_i,i}$	0	0
${}^i\mathbf{I}_{zz_i}$	I_1	I_2

(b) Modified

TABLE C.1 Physical parameters of the 2R robot according to the two different Denavit-Hartenberg kinematic conventions.

APPENDICES

In the following, the Lagrangian dynamic model is detailed by giving the symbolic expression for every term in (2.18). Remembering (1.4) and knowing

$$\mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} \quad (\text{C.1})$$

then (2.18), omitting the subscripts, can be rewritten as

$$\begin{aligned} \boldsymbol{\tau} = \mathbf{B}\boldsymbol{\Sigma}^{-1} & \left[\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + 2\dot{\mathbf{M}}(\mathbf{q})\dot{\mathbf{q}} + \ddot{\mathbf{M}}(\mathbf{q})\ddot{\mathbf{q}} \right. \\ & \left. + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\ddot{\mathbf{q}} + 2\dot{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \ddot{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \ddot{\mathbf{g}}(\mathbf{q}) \right] \\ & + [\mathbf{M}(\mathbf{q}) + \mathbf{B}]\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) \end{aligned} \quad (\text{C.2})$$

therefore, the dynamic model can be given as the symbolic expressions for each scalar term in

$$\mathbf{M}(\mathbf{q}) = \begin{pmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \end{pmatrix} \quad \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) = \begin{pmatrix} c_{11} & c_{12} \\ c_{12} & c_{22} \end{pmatrix} \quad \mathbf{g}(\mathbf{q}) = \begin{pmatrix} g_1 \\ g_2 \end{pmatrix} \quad (\text{C.3})$$

and of their first and second time derivatives. These are listed below.

$$m_{12} = I_2 + m_2(l_2^2 + r_1 l_2 \cos(q_2)) \quad (\text{C.4})$$

$$m_{11} = I_1 + m_1 l_1^2 + I_2 + m_2(r_1^2 + l_2^2 + 2r_1 l_2 \cos(q_2)) \quad (\text{C.5})$$

$$m_{22} = I_2 + m_2 l_2^2 \quad (\text{C.6})$$

$$c_{11} = -m_2 r_1 l_2 \sin(q_2) \dot{q}_2 \quad (\text{C.7})$$

$$c_{21} = m_2 r_1 l_2 \sin(q_2) \dot{q}_1 \quad (\text{C.8})$$

$$c_{12} = -m_2 r_1 l_2 \sin(q_2)(\dot{q}_1 + \dot{q}_2) \quad (\text{C.9})$$

$$c_{22} = 0 \quad (\text{C.10})$$

$$g_2 = g m_2 l_2 \cos(q_1 + q_2) \quad (\text{C.11})$$

$$g_1 = g_2 + g(m_1 l_1 + m_2 r_1) \cos(q_1) \quad (\text{C.12})$$

$$\dot{m}_{12} = -m_2 r_1 l_2 \sin(q_2) \dot{q}_2 \quad (\text{C.13})$$

$$\dot{m}_{11} = 2\dot{m}_{12} \quad (\text{C.14})$$

$$\dot{m}_{22} = 0 \quad (\text{C.15})$$

$$\dot{c}_{11} = -m_2 r_1 l_2 (\cos(q_2) \dot{q}_2^2 + \sin(q_2) \ddot{q}_2) \quad (\text{C.16})$$

$$\dot{c}_{21} = m_2 r_1 l_2 (\cos(q_2) \dot{q}_1 \dot{q}_2 + \sin(q_2) \ddot{q}_1) \quad (\text{C.17})$$

$$\dot{c}_{12} = \dot{c}_{11} - \dot{c}_{21} \quad (\text{C.18})$$

$$\dot{c}_{22} = 0 \quad (\text{C.19})$$

$$\dot{g}_2 = -g m_2 l_2 \sin(q_1 + q_2)(\dot{q}_1 + \dot{q}_2) \quad (\text{C.20})$$

APPENDIX C. DYNAMIC MODEL OF SEA 2R ROBOT

$$\dot{g}_1 = \dot{g}_2 - g(m_1 l_1 + m_2 r_1) \sin(q_1) \dot{q}_1 \quad (\text{C.21})$$

$$\ddot{m}_{12} = -m_2 r_1 l_2 (\cos(q_2) \dot{q}_2^2 + \sin(q_2) \ddot{q}_2) \quad (\text{C.22})$$

$$\ddot{m}_{11} = 2\ddot{m}_{12} \quad (\text{C.23})$$

$$\ddot{m}_{22} = 0 \quad (\text{C.24})$$

$$\ddot{c}_{11} = -m_2 r_1 l_2 (-\sin(q_2) \dot{q}_2^3 + 3 \cos(q_2) \dot{q}_2 \ddot{q}_2 + \sin(q_2) \ddot{q}_2^2) \quad (\text{C.25})$$

$$\ddot{c}_{21} = m_2 r_1 l_2 (-\sin(q_2) \dot{q}_1 \dot{q}_2^2 + \cos(q_2) (2\ddot{q}_1 \dot{q}_2 + \dot{q}_1 \ddot{q}_2) + \sin(q_2) \ddot{q}_1^2) \quad (\text{C.26})$$

$$\ddot{c}_{12} = \ddot{c}_{11} - \ddot{c}_{21} \quad (\text{C.27})$$

$$\ddot{c}_{22} = 0 \quad (\text{C.28})$$

$$\ddot{g}_2 = -g m_2 l_2 (\cos(q_1 + q_2) (\dot{q}_1 + \dot{q}_2)^2 + \sin(q_1 + q_2) (\ddot{q}_1 + \ddot{q}_2)) \quad (\text{C.29})$$

$$\ddot{g}_1 = \ddot{g}_2 - g(m_1 l_1 + m_2 r_1) (\cos(q_1) \dot{q}_1^2 + \sin(q_1) \ddot{q}_1) \quad (\text{C.30})$$

References

- [1] A. Albu-Schäffer, S. Haddadin, C. Ott, A. Stemmer, T. Wimböck, and G. Hirzinger, “The DLR lightweight robot – Lightweight design and soft robotics control concepts for robots in human environments,” *Industrial Robot*, vol. 34, no. 5, pp. 376–385, 2007.
- [2] A. Albu-Schäffer and G. Hirzinger, “A globally stable state feedback controller for flexible joint robots,” *Advanced Robotics*, vol. 15, no. 8, pp. 799–814, 2001.
- [3] R. M. Alexander, *Elastic Mechanisms in Animal Movement*. Cambridge University Press, 1988.
- [4] R. M. Alexander and H. C. Bennet-Clark, “Storage of elastic strain energy in muscle and other tissues,” *Nature*, vol. 265, pp. 114–117, 1977.
- [5] P. Beckerle, G. Mathijssen, and B. Vanderborght, “Series and parallel elastic actuation: Influence of operating positions on design and control,” *IEEE/ASME Transactions on Mechatronics*, vol. 22, no. 1, pp. 521–529, 2017.
- [6] P. A. Bhounsule, J. Cortell, A. Grewal, B. Hendriksen, J. D. Karssen, C. Paul, and A. Ruina, “Low-bandwidth reflex-based control for lower power walking: 65 km on a single battery charge,” *International Journal of Robotics Research*, vol. 33, no. 10, pp. 1305–1321, 2014.
- [7] R. Bischoff, J. Kurth, G. Schreiber, R. Köppe, A. Albu-Schäffer, A. Beyer, O. Eiberger, S. Haddadin, A. Stemmer, G. Grunwald, and G. Hirzinger, “The KUKA-DLR lightweight robot arm: A new reference platform for robotics research and manufacturing,” in *Proc. 41st International Symposium on Robotics*, 2010, pp. 1–10.
- [8] D. J. Braun, F. Petit, F. Huber, S. Haddadin, P. van der Smagt, A. Albu-Schäffer, and S. Vijayakumar, “Robots driven by compliant

REFERENCES

- actuators: Optimal control under actuation constraints,” *IEEE Transactions on Robotics*, vol. 29, no. 5, pp. 1085–1101, 2013.
- [9] B. Brogliato, *Nonsmooth mechanics*. Springer, 1999.
- [10] G. Buondonno, J. Carpentier, G. Saurel, N. Mansard, A. De Luca, and J.-P. Laumond, “Actuator design of compliant walkers via optimal control,” in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017, pp. 705–711.
- [11] G. Buondonno and A. De Luca, “A recursive Newton-Euler algorithm for robots with elastic joints and its application to control,” in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015, pp. 5526–5532.
- [12] G. Buondonno and A. De Luca, “Efficient computation of inverse dynamics and feedback linearization for VSA-based robots,” *IEEE Robotics and Automation Letters*, vol. 1, no. 3, pp. 908–916, Jul. 2016.
- [13] J. Carpentier, F. Valenza, N. Mansard *et al.*, “Pinocchio: Fast forward and inverse dynamics for poly-articulated systems,” <https://stack-of-tasks.github.io/pinocchio>, 2015–today.
- [14] M. Catalano, G. Grioli, E. Farnioli, A. Serio, C. Piazza, and A. Bicchi, “Adaptive synergies for the design and control of the Pisa/IIT Soft-Hand,” *International Journal of Robotics Research*, vol. 33, no. 5, pp. 768–782, 2014.
- [15] A. Chatterjee and A. Ruina, “A new algebraic rigid-body collision law based on impulse space considerations,” *Journal of Applied Mechanics*, vol. 65, no. 4, pp. 939–951, 1998.
- [16] S. Collins, A. Ruina, R. Tedrake, and M. Wisse, “Efficient bipedal robots based on passive-dynamic walkers,” *Science*, vol. 307, no. 5712, pp. 1082–1085, 2005.
- [17] J. J. Craig, *Introduction to Robotics: Mechanics and Control*, 2nd ed. Addison-Wesley Longman Publishing Co., Inc., 1989.
- [18] H. Dallali, “Modelling and dynamic stabilisation of a compliant humanoid robot, CoMan,” PhD thesis, University of Manchester, 2011.
- [19] A. De Luca, A. Albu-Schäffer, S. Haddadin, and G. Hirzinger, “Collision detection and safe reaction with the DLR-III lightweight robot arm,”

REFERENCES

- in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, pp. 1623–1630.
- [20] A. De Luca and W. Book, “Robots with flexible elements,” in *Handbook of Robotics*, 1st ed., B. Siciliano and O. Khatib, Eds. Springer, 2008, ch. 13, pp. 287–319.
 - [21] A. De Luca, R. Farina, and P. Lucibello, “On the control of robots with visco-elastic joints,” in *Proc. IEEE International Conference on Robotics and Automation*, 2005, pp. 4297–4302.
 - [22] A. De Luca, F. Flacco, A. Bicchi, and R. Schiavi, “Nonlinear decoupled motion-stiffness control and collision detection/reaction for the VSA-II variable stiffness device,” in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, pp. 5487–5494.
 - [23] A. De Luca and L. Lanari, “Robots with elastic joints are linearizable via dynamic feedback,” in *Proc. 34th IEEE Conference on Decision and Control*, 1995, pp. 3895–3897.
 - [24] A. De Luca and P. Lucibello, “A general algorithm for dynamic feedback linearization of robots with elastic joints,” in *Proc. IEEE International Conference on Robotics and Automation*, 1998, pp. 504–510.
 - [25] J. Denavit and R. Hartenberg, “A kinematic notation for lower pair mechanisms based on matrices,” *ASME Journal of Applied Mechanics*, vol. 22, pp. 215–221, 1955.
 - [26] M. Diehl, H. G. Bock, H. Diedam, and P.-B. Wieber, “Fast direct multiple shooting algorithms for optimal robot control,” in *Fast Motions in Biomechanics and Robotics: Optimization and Feedback Control*, M. Diehl and K. Mombaur, Eds. Springer Berlin Heidelberg, 2006, pp. 65–93.
 - [27] J. Engelsberger, 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. IEEE-RAS International Conference on Humanoid Robots*, 2014, pp. 916–923.
 - [28] R. Featherstone, “The calculation of robot dynamics using articulated-body inertias,” *International Journal of Robotics Research*, vol. 2, no. 1, pp. 13–30, 1983.

REFERENCES

- [29] R. Featherstone, *Robot Dynamics Algorithms*. Kluwer Academic Publishers, 1987.
- [30] R. Featherstone, *Rigid Body Dynamics Algorithms*. Springer, 2008.
- [31] F. Flacco, A. De Luca, I. Sardellitti, and N. Tsagarakis, “On-line estimation of variable stiffness in flexible robot joints,” *International Journal of Robotics Research*, vol. 31, no. 13, pp. 1556–1577, 2012.
- [32] C. Gaz, F. Flacco, and A. De Luca, “Identifying the dynamic model used by the KUKA LWR: A reverse engineering approach,” in *Proc. IEEE International Conference on Robotics and Automation*, 2014, pp. 1386–1392.
- [33] A. Giusti, J. Malzahn, N. G. Tsagarakis, and M. Althoff, “Combined inverse-dynamics/passivity-based control for robots with elastic joints,” in *Proc. IEEE International Conference on Robotics and Automation*, 2017, pp. 5281–5288.
- [34] M. Grebenstein, A. Albu-Schäffer, T. Bahls, M. Chalon, O. Eiberger, W. Friedl, R. Gruber, S. Haddadin, U. Hagn, R. Haslinger, H. Höppner, S. Jörg, M. Nickl, A. Nothhelfer, F. Petit, J. Reill, N. Seitz, T. Wimböck, S. Wolf, T. Wüsthoff, and G. Hirzinger, “The DLR Hand Arm System,” in *Proc. IEEE International Conference on Robotics and Automation*, 2011, pp. 3175–3182.
- [35] M. Grimmer, M. Eslamy, S. Gliech, and A. Seyfarth, “A comparison of parallel- and series elastic elements in an actuator for mimicking human ankle joint in walking and running,” in *Proc. IEEE International Conference on Robotics and Automation*, 2012, pp. 2463–2470.
- [36] G. Grioli, S. Wolf, M. Garabini, M. Catalano, E. Burdet, D. G. Caldwell, R. Carloni, W. Friedl, M. Grebenstein, M. Laffranchi, D. Lefeber, S. Stramigioli, N. Tsagarakis, M. Van Damme, B. Vanderborght, A. Albu-Schäffer, and A. Bicchi, “Variable stiffness actuators: The user’s point of view,” *International Journal of Robotics Research*, vol. 34, no. 6, pp. 727–743, 2015.
- [37] C. Guarino Lo Bianco, “Evaluation of generalized force derivatives by means of a recursive Newton-Euler approach,” *IEEE Transactions on Robotics*, vol. 25, no. 4, pp. 954–959, 2009.
- [38] C. Guarino Lo Bianco and E. Fantini, “A recursive Newton-Euler approach for the evaluation of generalized forces derivatives,” in *Proc.*

REFERENCES

- 12th IEEE International Conference on Methods and Models in Automation and Robotics*, 2006, pp. 739–744.
- [39] G. Guennebaud, B. Jacob *et al.*, “Eigen v3,” <http://eigen.tuxfamily.org>, 2010.
 - [40] S. Haddadin, “Optimal exploitation of soft-robot dynamics,” in *Soft Robotics*, A. Verl, A. Albu-Schäffer, O. Brock, and A. Raatz, Eds. Springer, 2015, pp. 92–99.
 - [41] S. Haddadin and E. Croft, “Physical human-robot interaction,” in *Handbook of Robotics*, 2nd ed., B. Siciliano and O. Khatib, Eds. Springer, 2016.
 - [42] S. Haddadin, K. Krieger, M. Kunze, and A. Albu-Schäffer, “Exploiting potential energy storage for cyclic manipulation: An analysis for elastic dribbling with an anthropomorphic robot,” in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 1789–1796.
 - [43] G. Hirzinger, A. Albu-Schäffer, M. Hähnle, I. Schaefer, and N. Sporer, “On a new generation of torque controlled light-weight robots,” in *Proc. IEEE International Conference on Robotics and Automation*, 2001, pp. 3356–3363.
 - [44] R. Höpler and M. Thümmel, “Symbolic computation of the inverse dynamics of elastic joint robots,” in *Proc. IEEE International Conference on Robotics and Automation*, 2004, pp. 4314–4319.
 - [45] B. Houska, H. J. Ferreau, and M. Diehl, “ACADO toolkit – An open-source framework for automatic control and dynamic optimization,” *Optimal Control Applications and Methods*, vol. 32, no. 3, pp. 298–312, 2011.
 - [46] Y. Ikemata, A. Sano, and H. Fujimoto, “A physical principle of gait generation and its stabilization derived from mechanism of fixed point,” in *Proc. IEEE International Conference on Robotics and Automation*, 2006, pp. 836–841.
 - [47] A. Jafari, N. Tsagarakis, B. Vanderborght, and D. Caldwell, “A novel actuator with adjustable stiffness (AwAS),” in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010, pp. 4201–4206.

REFERENCES

- [48] A. Jafari, N. Tsagarakis, B. Vanderborght, and D. Caldwell, “AwAS-II: A new actuator with adjustable stiffness based on the novel principle of adaptable pivot point and variable lever ratio,” in *Proc. IEEE International Conference on Robotics and Automation*, 2011, pp. 4638–4643.
- [49] A. Jubien, M. Gautier, and A. Janot, “Dynamic identification of the Kuka LWR robot using motor torques and joint torque sensors data,” in *Proc. 19th IFAC World Congress*, 2014, pp. 8391–8396.
- [50] N. Kashiri, M. Laffranchi, N. Tsagarakis, I. Sardellitti, and D. Caldwell, “Dynamic modeling and adaptable control of the CompAct arm,” in *Proc. IEEE International Conference on Mechatronics*, 2013, pp. 477–482.
- [51] C. Kirches, “Mixed-integer optimal control,” in *Fast Numerical Methods for Mixed-Integer Nonlinear Model-Predictive Control*. Springer, 2011, pp. 31–60.
- [52] D. B. Leineweber, I. Bauer, H. G. Bock, and J. P. Schlöder, “An efficient multiple shooting based reduced SQP strategy for large-scale dynamic process optimization. Part 1: Theoretical aspects,” *Computers & Chemical Engineering*, vol. 27, no. 2, pp. 157–166, 2003.
- [53] 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*, vol. 102, no. 2, pp. 69–76, 1980.
- [54] S. Migliore, E. Brown, and S. DeWeerth, “Biologically inspired joint stiffness control,” in *Proc. IEEE International Conference on Robotics and Automation*, 2005, pp. 4508–4513.
- [55] M. Mosadeghzad, N. G. Tsagarakis, G. A. Medrano-Cerda, and D. G. Caldwell, “Power efficient balancing control for humanoids based on approximate optimal ankle compliance regulation,” in *Proc. IEEE International Conference on Robotics and Automation*, 2014, pp. 5103–5108.
- [56] A. Müller, “Recursive second-order inverse dynamics for serial manipulators,” in *Proc. IEEE International Conference on Robotics and Automation*, 2017, pp. 2483–2489.
- [57] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.

REFERENCES

- [58] M. Naveau, J. Carpentier, S. Barthelemy, O. Stasse, and P. Souères, “METAPOD – Template META-programming applied to dynamics: CoP-CoM trajectories filtering,” in *Proc. IEEE-RAS International Conference on Humanoid Robots*, 2014, pp. 401–406.
- [59] F. Nori, S. Traversaro, J. Eljaik, F. Romano, A. Del Prete, and D. Pucci, “iCub whole-body control through force regulation on rigid non-coplanar contacts,” *Frontiers in Robotics and AI*, vol. 2, p. 6, 2015.
- [60] C. Ott, *Cartesian Impedance Control of Redundant and Flexible-Joint Robots*, ser. Springer Tracts in Advanced Robotics (STAR), vol. 49. Springer, 2008.
- [61] M. C. Özparpucu and S. Haddadin, “Optimal control for maximizing link velocity of visco-elastic joints,” in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 3035–3042.
- [62] G. Palli, C. Melchiorri, and A. De Luca, “On the feedback linearization of robots with variable joint stiffness,” in *Proc. IEEE International Conference on Robotics and Automation*, 2008, pp. 1753–1759.
- [63] G. Palli, C. Melchiorri, T. Wimböck, M. Grebenstein, and G. Hirzinger, “Feedback linearization and simultaneous stiffness-position control of robots with antagonistic actuated joints,” in *Proc. IEEE International Conference on Robotics and Automation*, 2007, pp. 4367–4372.
- [64] G. Pratt and M. Williamson, “Series elastic actuators,” in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1995, pp. 399–406.
- [65] S. Sager, *Numerical methods for mixed-integer optimal control problems*. Der Andere Verlag, 2005.
- [66] G. Saurel, J. Carpentier, N. Mansard, and J.-P. Laumond, “A simulation framework for simultaneous design and control of passivity based walkers,” in *Proc. IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots*, 2016, pp. 105–110.
- [67] R. Schiavi, G. Grioli, S. Sen, and A. Bicchi, “VSA-II: A novel prototype of variable stiffness actuator for safe and performing robots interacting with humans,” in *Proc. IEEE International Conference on Robotics and Automation*, 2008, pp. 2171–2176.

REFERENCES

- [68] E. Schuitema, D. G. Hobbelen, P. P. Jonker, M. Wisse, and J. D. Karssen, “Using a controller based on reinforcement learning for a passive dynamic walking robot,” in *Proc. IEEE-RAS International Conference on Humanoid Robots*, 2005, pp. 232–237.
- [69] G. Schultz and K. Mombaur, “Modeling and optimal control of human-like running,” *IEEE/ASME Transactions on Mechatronics*, vol. 15, no. 5, pp. 783–792, 2010.
- [70] S. Seok, A. Wang, M. Y. Chuah, D. Otten, J. Lang, and S. Kim, “Design principles for highly efficient quadrupeds and implementation on the MIT Cheetah robot,” in *Proc. IEEE International Conference on Robotics and Automation*, 2013, pp. 3307–3312.
- [71] A. Seyfarth, H. Geyer, R. Blickhan, S. Lipfert, J. Rummel, Y. Minekawa, and F. Iida, “Running and walking with compliant legs,” in *Fast Motions in Biomechanics and Robotics: Optimization and Feedback Control*, M. Diehl and K. Mombaur, Eds. Springer Berlin Heidelberg, 2006.
- [72] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modeling, Planning and Control*, 3rd ed. London: Springer, 2008.
- [73] M. Spong, “Modeling and control of elastic joint robots,” *ASME Journal of Dynamic Systems, Measurement, and Control*, vol. 109, no. 4, pp. 310–319, 1987.
- [74] E. Spyrakos-Papastavridis, D. G. Caldwell, and N. G. Tsagarakis, “Balance and impedance optimization control for compliant humanoid stepping,” in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2016, pp. 1349–1355.
- [75] E. Spyrakos-Papastavridis, N. Kashiri, J. Lee, N. G. Tsagarakis, and D. G. Caldwell, “Online impedance parameter tuning for compliant biped balancing,” in *Proc. IEEE-RAS International Conference on Humanoid Robots*, 2015, pp. 210–216.
- [76] I. Thorson and D. G. Caldwell, “A nonlinear series elastic actuator for highly dynamic motions,” in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 390–394.
- [77] M. Thümmel, M. Otter, and J. Bals, “Control of robots with elastic joints based on automatic generation of inverse dynamics models,” in

REFERENCES

- Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2001, pp. 925–930.
- [78] M. Tognon, B. Yüksel, G. Buondonno, and A. Franchi, “Dynamic decentralized control for protocentric aerial manipulators,” in *Proc. IEEE International Conference on Robotics and Automation*, 2017, pp. 6375–6380.
 - [79] P. Tomei, “A simple PD controller for robots with elastic joints,” *IEEE Transactions on Automatic Control*, vol. 36, no. 10, pp. 1208–1213, 1991.
 - [80] G. Tonietti, R. Schiavi, and A. Bicchi, “Design and control of a variable stiffness actuator for safe and fast physical human/robot interaction,” in *Proc. IEEE International Conference on Robotics and Automation*, 2005, pp. 528–533.
 - [81] T. D. Tuttle and W. P. Seering, “A nonlinear model of a harmonic drive gear transmission,” *IEEE Transactions on Robotics and Automation*, vol. 12, no. 3, pp. 368–374, 1996.
 - [82] R. Van Ham, B. Vanderborght, M. Van Damme, B. Verrelst, and D. Lefeber, “MACCEPA, the mechanically adjustable compliance and controllable equilibrium position actuator: Design and implementation in a biped robot dynamic computer simulation of robotic mechanisms,” *Robotics and Autonomous Systems*, vol. 55, no. 10, pp. 761–768, 2007.
 - [83] B. Vanderborght, A. Albu-Schäffer, A. Bicchi, E. Burdet, D. Caldwell, R. Carloni, M. Catalano, O. Eiberger, W. Friedl, G. Ganesh, M. Garabini, M. Grebenstein, G. Grioli, S. Haddadin, H. Hoppner, A. Jafari, M. Laffranchi, D. Lefeber, F. Petit, S. Stramigioli, N. Tsagarakis, M. Van Damme, R. Van Ham, L. Visser, and S. Wolf, “Variable impedance actuators: A review,” *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1601–1614, 2013.
 - [84] T. Verstraten, P. Beckerle, R. Furnémont, G. Mathijssen, B. Vanderborght, and D. Lefeber, “Series and parallel elastic actuation: Impact of natural dynamics on power and energy consumption,” *Mechanism and Machine Theory*, vol. 102, no. Supplement C, pp. 232–246, 2016.
 - [85] T. Verstraten, G. Mathijssen, J. Geeroms, L. Flynn, B. Vanderborght, and D. Lefeber, “On the importance of a motor model for the optimization of SEA-driven prosthetic ankles,” in *Wearable Robotics: Challenges and Trends: Proceedings of the 2nd International Symposium on*

REFERENCES

- Wearable Robotics, WeRob2016, October 18-21, 2016, Segovia, Spain*, J. González-Vargas, J. Ibáñez, J. L. Contreras-Vidal, H. van der Kooij, and J. L. Pons, Eds. Springer International Publishing, 2017, pp. 403–407.
- [86] M. W. Walker and D. E. Orin, “Efficient dynamic computer simulation of robotic mechanisms,” *ASME Journal of Dynamic Systems, Measurement, and Control*, vol. 104, no. 3, pp. 205–211, 1982.
 - [87] E. R. Westervelt, J. W. Grizzle, C. Chevallereau, J. Choi, and B. Morris, *Feedback Control of Dynamic Bipedal Robot Locomotion*. CRC Press, 2007.
 - [88] M. Wisse, G. Feliksdal, J. Van Frankenhuyzen, and B. Moyer, “Passive-based walking robot,” *Robotics and Automation Magazine*, pp. 52–62, Jun. 2007.
 - [89] M. Wisse and R. Q. Van der Linde, *Delft pneumatic bipeds*. Springer Science & Business Media, 2007, vol. 34.
 - [90] S. Wolf, O. Eiberger, and G. Hirzinger, “The DLR FSJ: Energy based design of a variable stiffness joint,” in *Proc. IEEE International Conference on Robotics and Automation*, 2011, pp. 5082–5089.
 - [91] S. Wolf and G. Hirzinger, “A new variable stiffness design: Matching requirements of the next robot generation,” in *Proc. IEEE International Conference on Robotics and Automation*, 2008, pp. 1741–1746.
 - [92] B. Yüksel, G. Buondonno, and A. Franchi, “Differential flatness and control of protocentric aerial manipulators with any number of arms and mixed rigid-/elastic-joints,” in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2016, pp. 561–566.
 - [93] [Online]. Available: <http://www.dlr.de/rmc/rm/en>
 - [94] [Online]. Available: <http://www.dlr.de/rmc/rm/en/desktopdefault.aspx/tabid-11666/#gallery/28234>
 - [95] [Online]. Available: <http://harmonicdrive.net/>
 - [96] [Online]. Available: <https://www.kuka.com/>
 - [97] [Online]. Available: <http://www.qbrobotics.com/>
 - [98] [Online]. Available: <http://www.rethinkrobotics.com/baxter/>

REFERENCES

- [99] [Online]. Available: <http://www.softhands.eu/>
- [100] [Online]. Available: <http://www.viactors.org/>