Technische Universität München
Fakultät für Elektrotechnik und Informationstechnik

Enhanced Robot Compliance, State Estimation and Identification using Distributed Tactile Feedback: Leveraging Redundancy and Multimodality

Quentin Leboutet, M.Sc.

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik
der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften (Dr. -Ing.)

genehmigten Dissertation.

Vorsitzender:

Prof. Dr. -Ing. Ulf Schlichtmann

Prüfer der Dissertation:

1.Prof. Gordon Cheng, Ph.D.

2.Prof. Gentiane Venture, Ph.D.

3.Prof. Ludovic Righetti, Ph.D.

Die Dissertation wurde am 12.01.2022 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 12.07.2022 angenommen.

To Mona.

Abstract

In a rapidly evolving society, the prospect of short-term, large-scale deployment of a wide variety of autonomous robots, and in particular humanoid robots, for use in everyday life applications raises multiple questions and challenges, both from a societal and engineering perspective. Obviously, such robots must be endowed with high levels of adaptability in order to safely and effectively operate within human environments. Much research suggests that advanced sensory capabilities are in this case essential to provide robots with adequate levels of situational awareness, allowing them to work side-by-side with humans or to execute advanced whole-body manipulation tasks. In this context, the implementation of an artificial sense of touch that can potentially equip any robot appears to be a promising research direction. Nevertheless, the fact entirely covering a robot with an artificial skin remains problematic in practice, for both hardware and algorithmic reasons. As a matter of fact, not only should the skin data be processed in real-time by robots with limited computational capabilities, but it should also be integrated within a more general control framework in order to generate suitable robot behaviors. Moreover, mounting an artificial skin on a robot is likely to significantly alter its dynamic properties, which may eventually lead to degraded control performance or even instability in the case of floating base mechanisms.

The work of this thesis is organized around the following research axes in order to provide a coherent answer to the above mentioned challenges. First, the modalities of use of tactile information in the context of human-robot interaction and distributed state estimation are explored, respectively through the development of a whole-body tactile compliance control strategy, and of a distributed sensor fusion framework. The ability to re-identify the robot's inertial parameters once it is covered with artificial skin is then explored via the development of a physically consistent identification strategy. This strategy has the potential to operate online by exploiting the skin's tactile feedback. For each of these research directions, particular emphasis is placed on the use of artificial skin data as a simplification agent for the considered problem.

Kurzfassung

In einer sich rasch entwickelnden Gesellschaft wirft die Aussicht auf den kurzfristigen, groß angelegten Einsatz zahlreicher autonomer Roboter, insbesondere humanoider Roboter, für Anwendungen im täglichen Leben zahlreiche Fragen und Herausforderungen auf, sowohl aus gesellschaftlicher als auch aus technischer Sicht. Es liegt auf der Hand, dass solche Roboter in einem hohen Maß anpassungsfähig sein müssen, um in der menschlichen Umgebung sicher und effektiv arbeiten zu können. Viele Forschungsarbeiten deuten darauf hin, dass fortschrittliche sensorische Fähigkeiten in diesem Fall unerlässlich sind, um den Robotern ein angemessenes Maß an Situationsbewusstsein zu verleihen, damit sie Seite an Seite mit dem Menschen arbeiten oder fortgeschrittene Manipulationsaufgaben mit dem ganzen Körper ausführen können. In diesem Zusammenhang scheint die Implementierung eines künstlichen Tastsinns, mit dem potenziell jeder Roboter ausgestattet werden kann, eine vielversprechende Forschungsrichtung zu sein. Dennoch bleibt die vollständige Bedeckung eines Roboters mit einer künstlichen Haut in der Praxis problematisch, sowohl aus Hardware- als auch aus algorithmischen Gründen. Die Daten der Haut sollten nicht nur in Echtzeit von Robotern mit begrenzten Rechenkapazitäten verarbeitet werden, sondern auch mit vorhandenen Steuerungs- und Regelungs-Algorithmen integriert werden, um geeignete Verhaltensweisen des Roboters zu erzeugen. Darüber hinaus verändert die Anbringung einer künstlichen Haut am Roboter dessen dynamische Eigenschaften erheblich. Dies beeinflusst das Regelverhalten und kann zu einem instabilen Bewegungsablauf des Gehens führen.

Die Dissertation ist um die folgenden Forschungsachsen organisiert, um eine kohärente Antwort auf die oben genannten Herausforderungen zu geben: Erstens werden die Modalitäten der Nutzung taktiler Informationen im Kontext der Mensch-Roboter-Interaktion und der verteilten Zustandsschätzung erforscht, und zwar durch die Entwicklung einer taktilen Steuerungs-und Regelungsstrategie des ganzen Körpers und eines verteilten Sensor-Fusionssystems. Die Fähigkeit, die Trägheitsparameter des Roboters wiederzuerkennen, sobald er mit künstlicher Haut bedeckt ist, wird dann durch die Entwicklung einer physikalisch konsistenten Identifizierungsstrategie erforscht. Diese Strategie hat das Potenzial, online zu funktionieren, indem sie das taktile Feedback der Haut ausnutzt. Bei jeder dieser Forschungsrichtungen wird besonderer Wert auf die Verwendung von Daten über die künstliche Haut gelegt, um das betreffende Problem zu vereinfachen.

Acknowledgments

First and foremost, I would like to express my gratitude to Prof. Dr. Gordon Cheng for giving me the opportunity to conduct my doctoral research within his institute. I am grateful for his continuous supervision and support.

I would also like to express my gratitude to Dr. Alexandre Janot, who mentored this thesis and provided highly valuable scientific feedback.

To my love, Mona Y. Hussain, thank you for being by my side, thank you for your help during tough days, for your kindness, for your patience and for your encouragements. Thank you for being who you are. This would not have been possible without you.

I would like to thank my sister Caroline, my brother-in-law Enzo, as well as my parents and grand-parents for their unconditional support. To my dear friends Romain Duflot, Leopold D'Auvergne, Raphael Garcia-Melgares, Mathieu Gross and Germain Pradines: thank you for being such wonderful persons.

My gratitude also goes to the whole ICS team for its continued support throughout these years of effort, hard work but also joy and laughter. I would especially like to thank Dr. Julio Rogelio Guadarrama-Olvera, Simon Armleder, Dr. Florian Bergner, Dr. Stefan Ehrlich, Dr. Pablo Lanillos Pradas, Nicolas Berberich, Dr. Shunki Itadera and Dr. Jasmin Kajopoulos for the interesting and constructive scientific conversations. I would also like to thank Constantin Uhde, Dr. Erhard Wieser, Natalia Paredes and Ilya Dianov for their joint efforts and help with teaching as well as Katarina Stadler and Sebastian Stenner for their assistance with manufacturing and 3D printing. A special thanks also goes to Ilona Nar-Witte, Wibke Borngesser, and Brigitte Rosenlehner for their availability, kindness and support in administrative formalities.

I would finally like to thank my former students Adon Yazigi, Ayca Öner, Julien Roux, Lucie Galland, Hanwen Shao, Po-Chuan Chan, Louis Rossier-Tillard, XiaoTian Guo, Primoz Kocevar and Stephan Luming-Li for their dedication and the serious of their work.

“The mystery of life isn’t a problem to solve, but a reality to experience.”

— Frank Herbert, (*Dune*)

“Try not to become a man of success, but rather try to become a man of value.”

— Albert Einstein

Table of Contents

List of Publications

Parts of this thesis have already been published in:

Journals

J1: **Leboutet, Q**. Roux, J. Janot, A. Guadarrama-Olvera, J. R. Cheng, G. “Inertial Parameter Identification in Robotics: A Survey”. In: MDPI Applied Science

J2: Cheng, G. Dean-Leon, E. Bergner, F. Guadarrama-Olvera, J. R. **Leboutet, Q**. Mittendorfer, P. “A comprehensive realisation of Robot Skin: Sensors, Sensing, Control and Applications”. In: Proceedings of the IEEE 107.10 (2019), pp. 2034–2051

J3: **Leboutet, Q**. Dean-Leon, E. Bergner, F. Cheng, G. “Tactile-based whole-body compliance with force propagation for mobile manipulators”. In: IEEE Transactions on Robotics, 35.2 (2019), pp.330-342.

J4: **Leboutet, Q**. Janot, A. Armleder, S. Guadarrama-Olvera, J. R. Cheng, G. “Adaptive Stabilization of Floating-Base Robots using the Distributed Tactile Feedback of an Artificial Skin”. In: IEEE Transactions on Robotics – in submission.

Peer-Reviewed Conference Papers

C1: **Leboutet, Q**. Bergner, F. Cheng, G. “Online Configuration Selection for Redundant Arrays of Inertial Sensors: Application to Robotic Systems Covered with a Multimodal Artificial Skin”. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2020

C2: **Leboutet, Q**. Guadarrama-Olvera, J. R. Cheng, G. “Second-order Kinematics for Floating-base Robots using the Redundant Acceleration Feedback of an Artificial Sensory Skin”. In: IEEE International Conference on Robotics and Automation (ICRA). 2020

C3: **Leboutet, Q**. Dean-Leon, F. Cheng, G. “Tactile-based compliance with hierarchical force propagation for omnidirectional mobile manipulators”. In: IEEE-RAS 16th International Conference on Humanoid Robots (2016), pp. 926-931.

List of Figures

List of Tables

List of Algorithms

List of Acronyms

A

AdaNN - Adaline Neural Network... x, 1, 22, 93, 96, 98, 103, 107, 110, 111, 115, 123, 158

AO-IMU - Accelerometer-Only Inertial Measurement Unit...................................... 1, 70

B

BIRDy - Benchmark for Identification of Robot Dynamics.. xi, xii, xviii, 1, 5, 93, 96–100, 104, 106, 109, 116, 123, 125, 127, 142–144, 155, 158–160, 163

C

CAD - Computer-Aided Design... 1, 103, 111

CDKF - Central Difference Kalman Filter..... xiii, xxi, 1, 22, 98, 106, 107, 123, 158, 172, 175, 176

CLIE - Closed Loop Input Error. xvii, 1, 19, 20, 93, 95, 96, 98, 101, 103, 106, 110–112, 115, 116, 123, 158

CLOE - Closed Loop Output Error........... 1, 19, 20, 93, 95, 96, 98, 101, 103, 106, 110–112, 115–117, 123, 158

CoM - Center of Mass........... xviii, 1, 32, 65, 117, 126, 134, 135, 138, 139, 141, 151, 153

CoP - Center of Pressure.. 1, 42, 47

CPU - Central Processing Unit.. 1, 56, 58

D

DCM - Divergent Component of Motion.. 1

DDM - Direct Dynamic Model.................... 1, 8, 20, 21, 99, 101, 103, 105, 110–112, 116

DIDIM - Direct and Inverse Dynamics Identification Model. xviii, 1, 19, 20, 27, 93, 95, 96, 98, 101, 103, 105, 107, 109–112, 114, 115, 123, 158

DoF - Degrees of Freedom.................................... 1, 30, 34, 56, 82, 101, 149, 158, 163

E

EKF - Extended Kalman Filter..... xiii, xxi, 1, 22, 25, 77, 94–96, 98, 106, 107, 112, 123, 158, 171

F

FoM - Figure of Merit.. 1, 118–122, 144, 147

FPGA - Field-Programmable Gate Array... 1, 2

G

GF-IMU - Gyroscope-Free Inertial Measurement Unit.. 1, 70

GTLS - Generalized Total Least-Square.. 1, 14

H

HTRNN - Hopfield-Tank Recurrent Neural Network..... 1, 25, 93, 96, 98, 103, 107, 110, 111, 115, 123, 158

I

IDIM - Inverse Dynamics Identification Model xii, xvii, xviii, 1, 12–20, 22–25, 27–29, 93–96, 98, 100, 103–105, 107, 109–116, 118–123, 127, 130, 132, 133, 143–146, 148–155, 158

IDM - Inverse Dynamic Model 1, 7, 8, 14, 30, 102, 110, 111

IFT - Iterative Feedback Tuning 1

IMU - Inertial Measurement Unit 1, 3, 69–71, 73, 76, 77, 79, 82, 83, 92, 142, 148, 158

IRLS - Iteratively Reweighted Least Squares 1, 12, 13, 27, 93, 96, 98, 104, 107, 110, 111, 123, 133, 143, 145, 158

IV - Instrumental Variables xviii, 1, 20, 27, 93, 95, 96, 98, 103, 105, 107, 109–112, 114–116, 123, 158

K

KKT - Karush-Kuhn-Tucker 1

L

LIPM - Linear Inverted Pendulum Model 1

LMI - Linear Matrix Inequality 1, 5, 25, 27, 28, 127, 131, 159

M

MCS - Monte Carlo Simulation xii, 1, 10, 88, 108, 109, 115, 119–122, 134, 143, 145, 147, 155

MEMS - Micro Electro-Mechanical Systems 1, 70, 79, 84

ML - Maximum Likelihood 1, 16, 93, 94, 96, 98, 103, 104, 109–112, 115, 116, 123, 158

N

NKF - Nonlinear Kalman Filter xvii, 1, 21, 93, 101, 103, 106, 111, 112, 115, 116

O

OEM - Optimal Exciting Motions 1, 32

OLS - Ordinary Least Squares 1, 12, 16, 18–20, 22–25, 27–29, 93–96, 98, 100, 103, 104, 107, 109–112, 115, 116, 123, 143, 145, 158

P

PC - Physically Consistent xii, xviii, 1, 27, 28, 93, 96, 107, 112, 113, 123, 127, 132, 133, 143, 145, 146, 149, 158

PCB - Printed Circuit Board 1, 2

PF - Particle Filter 1, 22, 106

Q

QP - Quadratic Programming 1, 5, 28, 33, 36, 38, 40, 47, 132, 133, 135, 148, 154, 159

R

RBDL - Rigid Body Dynamic Library 1

RK - Runge-Kutta........ 1, 99, 103, 105

RNE - Recursive Newton Euler........ 1, 21

ROS - Robot Operating System........ 1, 101, 148, 160

S

SDP - Semi-Definite Programming........ 1, 5, 25, 27, 130, 131, 145, 159

SMU - Set Membership Uncertainty........ 1, 94

SNR - Signal-to-Noise Ratio........ 1, 13, 158

SoT - Stack-of-Task........ 1

SPKF - Sigma-Point Kalman Filter........ xiii, 1, 22, 77, 96, 172

SQP - Sequential Quadratic Programming........ xii, xviii, 1, 25, 125, 127, 133, 143–146, 148–155

SRCDKF - Square Root Central Difference Kalman Filter........ 1, 98, 106, 107, 123, 158

SREKF - Square Root Extended Kalman Filter........ 1, 106, 123, 158

SRI - Singularity Robust Inverse........ 1, 34

SRUKF - Square Root Unscented Kalman Filter........ 1, 98, 106, 107, 123, 158

STP-BV - Sphere-Torus-Patches Bounding Volume........ 1, 43

T

TLS - Total Least Squares........ 1, 12–14, 93, 94, 96, 98, 104, 110, 115, 123, 158

TSID - Task Space Inverse Dynamics........ 1

TSU - Tactile Section Unit........ 1, 2, 58, 59

U

UKF - Unscented Kalman Filter........ xiii, xxi, 1, 22, 78, 85, 86, 98, 106, 107, 112, 123, 158, 172–175

URDF - Unified Robot Description Format........ 1, 116, 160

W

WLS - Weighted Least Squares.. xviii, 1, 12, 16, 27, 28, 93–96, 98, 104, 107, 109–111, 113, 115, 116, 123, 133, 143–146, 149–154, 158

Z

ZMP - Zero Moment Point........ 1, 49, 65, 126

Chapter 1

Introduction

1.1. Motivation and Problem Description

Originally developed for purely industrial or scientific purposes, robots were not intended to physically interact with humans and were therefore supposed to remain isolated for safety and operability reasons. This situation is now rapidly evolving, as robots are currently about to be introduced into multiple sectors of society, including home automation, medicine, education and entertainment, where they will be required to share their workspace with humans on a daily basis [1, 2, 3]. To be seamlessly integrated into such complex environments, it is essential that robots possess human-like physical and sensory capabilities. In essence, robots must not only operate autonomously and reliably, but they must also be safe, intuitive and robust enough to cooperate and even physically interact with the different components of their environment and its actors [4]. One way to achieve this specific purpose is to endow robots with a human-like sense of touch, for example by supplementing their proprioceptive sensors with an artificial skin capable of perceiving physical interaction with the environment at any point on their body and in a wide range of modalities. Just as humans use skin to interact with the world, preliminary results [5, 6] suggests that robots would greatly benefit from such a capability. Although initially limited by the prohibitive cost of sensors, the use of large numbers of highly redundant distributed multimodal sensing elements, covering the entire surface of a robot – thereby acting like an *artificial skin* – is now receiving sustained attention due to the large-scale democratization of low-cost sensors, referred to as micro-electro-mechanical systems (MEMS) [7]. It appears that the use of distributed tactile feedback is relevant in multiple areas of robotics because, unlike proprioceptive feedback, it is robust to nullspace interactions and provides a direct, context-dependant assessment of the physical interaction at the contact point level. Such features are of paramount importance for the generation of robust compliant motions and more generally adaptive context-dependent control strategies. Although the continuous increase in embedded computing power as well as the emergence of new design paradigms and the development of high-performance control architectures have enabled the advent of a new family of more agile and versatile walking robots, with improved dynamic capabilities and sharp reflexes (e.g. Atlas-2, MIT Cheetah 3, MIT Mini-Cheetah, Spot, etc.), the critical issues of human-robot physical interaction, of its intuitiveness and of the robustness of control strategies remain unsolved in practice. It is worth emphasizing that the first two issues are intrinsically correlated to the notion of *whole-body compliance*: in practice, the different forces resulting from desired or accidental physical interactions between the robot and its environment must be detected by a set of dedicated sensors and, if necessary, propagated in order to generate predictable – potentially safe – reactive robot behaviors. Mobile and floating-base robots, in particular, can exploit their locomotion sys-

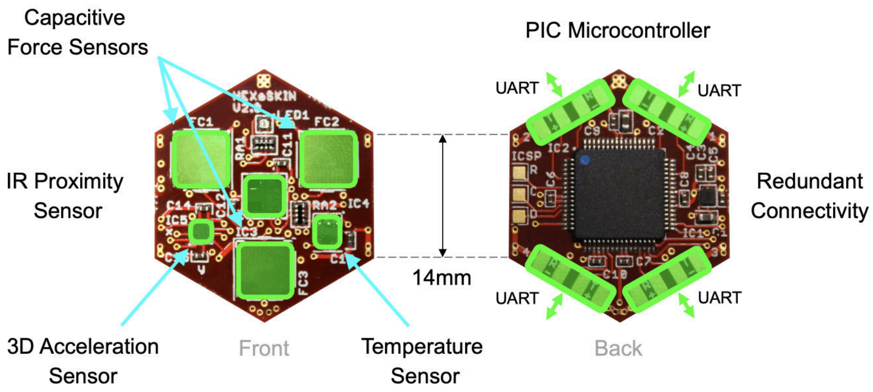


Figure 1 The artificial skin used in this work.

tem to provide enhanced compliance capabilities, although it is worth mentioning that a vast majority of robots currently exhibit very limited perception capabilities of their close-contact environment and, as such, prove to be potentially dangerous for humans or objects they fail to detect. Control robustness on the other hand is intrinsically linked to the ability of a robot to autonomously assess the changes occurring on its own body in terms of dynamic properties or coupling with the external environment. This may for instance arise during the manipulation of an unknown payload. As dynamic parameter biases are identified, they must either be counteracted or on the contrary exploited in order to increase tracking performance and stability by taking suitable control actions.

1.2. Thesis Contributions

This work aims at providing a set of control and estimation algorithms leveraging multimodal tactile feedback in the context of whole-body human-robot physical interaction, state estimation, and dynamic parameter identification. As the details of how the tactile information is generated and provided to the controller are out of the scope of this work, they will only be addressed superficially. The artificial skin used in this thesis consists of identical, hexagonally shaped circuits (or *skin cells*), connected to each other with flexible PCBs and assembled into patches of different sizes (c.f Fig. 1). An extensive description of the skin can be found in [6]. One of the most notable features of this artificial skin technology is its multimodality, each skin cell being capable of simultaneously measuring four different tactile quantities, namely pressure (through a set of three capacitive force sensors [8]), pre-touch (using an infrared proximity sensor), temperature (with a dedicated temperature sensor), vibrations and accelerations (using a 3D accelerometer). The data generated by each skin patch is transmitted via high-speed serial communication to a FPGA module – the Tactile Section Unit (TSU) – before begin broadcasted to the host computer at a maximum frequency of 250Hz via Gigabit Ethernet. In this manner, a temporal resolution of $4ms$ can be achieved, which is similar to the $5ms$ temporal resolution of Human skin. The main control and estimation problems related to the use of multimodal information will be presented and discussed in detail. It is worth emphasizing that the algorithms developed during this work are meant to be used by any robot equipped with standard sensors (e.g. IMU, force/torque, joint encoders) and any type of artificial skin providing multimodal information and spatial calibration. The contributions of this work can be summarized in three main blocks, namely the algorithms for tactile whole-body compliance, algorithms for tactile-based distributed state estimation and eventually the algorithms for tactile-based inertial parameter identification.

1.2.1. Contributions to Whole-Body Human-Robot Physical Interaction

We propose a new control method, providing mobile and floating-base robots with whole-body compliance capabilities, in response to multi-contact physical interactions with their environment. The external forces applied to the robot, as well as their localization on its kinematic tree, are measured using a multimodal, self-configuring and self-calibrating artificial skin. We formulate a compliance control law in Cartesian space, as a set of quadratic optimization problems, solved in parallel for each limb involved in the interaction process. This specific formulation makes it possible to determine the torque commands required to generate the desired reactive behaviors, while taking the robot kinematic and dynamic constraints into account. When a given limb fails to produce the desired compliant behavior, the generalized force residual at the considered contact points is propagated to a parent limb in order to be adequately compensated. Hence, the robot's compliance range can be extended in a manner that is both robust and easily adjustable. The experiments performed on a dual-arm velocity-controlled mobile manipulator, show that the proposed methodology is robust to both nullspace interactions and robot physical constraints.

1.2.2. Contributions to State Estimation

1.2.2.1 Estimation of High-Order Motion Derivatives Using the Distributed Inertial Feedback from and Artificial Skin

A well-known issue related to dynamic control and online identification techniques is the need for proper joint acceleration measurements. In this work, we propose a new estimation method for first and second-order kinematics for floating-base robots, based on highly redundant distributed inertial feedback. The linear acceleration of each robot link is measured at multiple points using a multimodal, self-configuring and self-calibrating artificial skin. The proposed algorithm is two-fold: i) the skin acceleration data is fused at the link level for state dimensionality reduction; ii) the estimated values are then fused limb-wise with data from the joint encoders and the main inertial measurement unit (IMU), using a Sigma-point Kalman filter. In this manner, it is possible to estimate the joint velocities and accelerations while avoiding the lag and noise amplification phenomena associated with conventional numerical derivation approaches. Experiments performed on the right arm and torso of a REEM-C humanoid robot, demonstrate the consistency of the proposed estimation method.

1.2.2.2 Data Reduction through Optimal Sensor Selection

Multiple approaches to the estimation of high-order motion derivatives for innovative control applications now rely on the data collected by redundant arrays of inertial sensors mounted on robots, with promising results. However, most of these works suffer from scalability issues induced by the considerable amount of data generated by such large-scale distributed sensor systems. In this work, we propose a new adaptive sensor-selection algorithm, for distributed inertial measurements. Our approach consists in using the data of a subset of sensors, selected among a larger collection of inertial sensing elements covering a rigid robot link. The sensor selection process is formulated as an optimization problem, and solved using a projected gradient heuristics. The proposed method can run online on a robot and be used to recalculate the selected sensor arrangement on the fly when physical interaction or potential sensor failure is detected. The tests performed on a simulated UR5 industrial manipulator covered with a multimodal artificial skin, demonstrate the consistency and performance of the proposed sensor-selection algorithm.

1.2.3. Contributions to Inertial Parameter Identification

1.2.3.1 Quantitative Performance Assessment of Identification Algorithms

Although covering an entire humanoid robot with an artificial skin has multiple advantages, it unfortunately tends to modify the robot dynamic properties as the number of cells increases. For instance, the total mass of the skin system mounted on our REEM-C humanoid robot adds an extra 6.5 kg to the initial robot mass, and shifts its center of mass to the front by several centimeters. To ensure suitable control performance and stabilization, the new set of robot dynamic parameters must be identified. Inertial parameter identification is a widely explored topic, with multiple contributions. We here review, analyze and compare a range of state-of-the-art approaches to inertial parameter identification in the context of robotics. We introduce "*BIRDy* (**B**enchmark for **I**dentification of **R**obot **Dy**namics)", an open-source Matlab toolbox, allowing a systematic and formal performance assessment of the considered identification algorithms on either simulated or real serial robot manipulators. Seventeen of the most widely used approaches found in the scientific literature are implemented and compared to each other, namely: the Inverse Dynamic Identification Model with Ordinary, Weighted, Iteratively Reweighted and Total Least-Squares (IDIM-OLS, -WLS, -IRLS, -TLS); the Instrumental Variables method (IDIM-IV), the Maximum Likelihood (ML) method; the Direct and Inverse Dynamic Identification Model approach (DIDIM); the Closed-Loop Output Error (CLOE) method; the Closed-Loop Input Error (CLIE) method; the Direct Dynamic Identification Model with Nonlinear Kalman Filtering (DDIM-NKF), the Adaline Neural Network (AdaNN), the Hopfield-Tank Recurrent Neural Network (HTRNN) and eventually a set of Physically Consistent (PC-) methods allowing the enforcement of parameter physicality using Semi-Definite Programming, namely the PC-IDIM-OLS, -WLS, -IRLS, PC-IDIM-IV, and PC-DIDIM. BIRDy is robot-agnostic and features a complete inertial parameter identification pipeline, from the generation of symbolic kinematic and dynamic models to the identification process itself. This includes functionalities for excitation trajectory computation as well as the collection and pre-processing of experimental data. In this work, the proposed methods are first evaluated in simulation, following a Monte Carlo scheme on models of the 6-DoF TX40 and RV2SQ industrial manipulators, before being tested on the real robot platforms.

The robustness, precision, computational efficiency and context of application of the different methods are investigated and discussed.

1.2.3.2 Online Physically-Consistent Inertial Parameter Identification with Geometric Relaxation and Local Excitation

Online identification of a robot's dynamic parameters is key to adaptability. In this chapter, we propose an online physically consistent batch identification paradigm, enabling regular update of the dynamic parameters of the robot potentially coupled to an external payload. The physical consistency of the estimated parameters, usually expressed as a set of linear matrix inequalities (LMI), thereby requiring the use of expensive Semi-Definite Programming techniques (SDP), is here formulated recursively by adopting a new parameter representation. As a matter of fact, this allows reformulating the identification problem as a sequence of constrained quadratic optimization problem (QP), which can then be solved in real-time on the robot within a dedicated thread. To provide the robot controller with a continuous stream of physically consistent parameters between two consecutive updates of the identification thread, we propose a geometric interpolation method along geodesic path of the physicality constraint, formulated as a Riemannian manifold. In the context of a manipulation task, we demonstrate how the tactile feedback provided by the artificial skin can be leveraged in order to refine the parameter estimates of the coupled system made by the robot and its payload. The experiments performed in simulation on BIRDy demonstrate the validity of the proposed approach. Preliminary experiments results performed on a humanoid robot covered with an artificial skin demonstrate its ability to run online within a robot control framework.

1.3. Thesis Outline

This thesis is structured as follows.

- Chapter 2 covers briefly the fundamentals in terms of parameter identification and hierarchical control of floating base robots that will be used in this thesis.
- Chapter 3 addresses the issue of whole-body compliant motion generation using the force and proximity feedback of an artificial skin.
- Chapter 4 explores the use of whole-body distributed inertial feedback for joint motion derivative estimation.
- Chapter 5 formally compares the different existing inertial parameter identification methods applied in robotics and discusses about potential ways to leverage tactile feedback in the context of dynamic parameter identification.
- Chapter 6 discusses the different aspects of online physically-consistent inertial parameter identification in the context of floating-base robots.
- Finally, Chapter 7 summarizes the main results of this thesis and provides a brief discussion of future research directions.

Chapter 2

Fundamentals of Inertial Parameters Identification and Hierarchical Control

This chapter provides a brief overview on a set of key theoretical concepts that are relevant to the derivations of this thesis. The main approaches to inertial parameter identification of fixed and floating base mechanisms are presented and discussed, along with elements of constrained hierarchical control of floating base robots. This chapter is structured as follows: section 2.1 provides an overview of robot modeling and control in the context of identification. Section 2.2 provides a theoretical overview over the most widely used inertial parameter identification algorithms in the field of robotics, highlighting their differences, weak points and discussing the potential implications in terms of expected accuracy, robustness and runtime performance. Section 2.3 discusses the different approaches to enforcing physicality conditions within a parameter identification process. Section 2.4 then covers the particularities related to the identification of floating base mechanisms. Finally, section 2.5 provides an overview of constrained hierarchical control techniques applied to floating-base robots.

2.1. Robot Modeling and Control in the Context of Inertial Parameter Identification

This section introduces some of the most fundamental concepts of inertial parameter identification and discusses the issues of structural and data rank deficiency.

2.1.1. Inverse and Direct Dynamic Models

The inverse dynamic model (IDM) of a rigid serial robot manipulator with n-degrees-of-freedom relates the joint-space motion quantities $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}} \in \mathbb{R}^n$, to the generalized forces $\boldsymbol{\tau}_{idm} \in \mathbb{R}^n$ applied to the system by the actuators and the external environment. The IDM can be derived using the Euler-Lagrange formalism (c.f. [9] and Appendix A.2), resulting in the following equation

$$\boldsymbol{M}(\boldsymbol{\chi}, \boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{\chi}, \boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{\chi}, \boldsymbol{q}) + \boldsymbol{\zeta}(\boldsymbol{\chi}, \dot{\boldsymbol{q}}) = \boldsymbol{\tau}_{idm}, \tag{1}$$

where $\boldsymbol{M}(\boldsymbol{\chi}, \boldsymbol{q}) \in \mathbb{R}^{n\times n}$ denotes the generalized inertia matrix, $\boldsymbol{C}(\boldsymbol{\chi}, \boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}^{n\times n}$ the Coriolis and centripetal effects matrix, $\boldsymbol{g}(\boldsymbol{\chi}, \boldsymbol{q}) \in \mathbb{R}^n$ the gravitational torque vector, and $\boldsymbol{\zeta}(\boldsymbol{\chi}, \dot{\boldsymbol{q}}) \in \mathbb{R}^n$ the friction vector. This equation is parameterized by the vector $\boldsymbol{\chi} = [\boldsymbol{\chi}_1^\top \boldsymbol{\chi}_2^\top \cdots \boldsymbol{\chi}_n^\top]^\top \in \mathbb{R}^p$ concatenating the standard dynamic parameters of each robot link j, expressed as

$$\begin{aligned} \boldsymbol{\chi}_j \;=\; & [XX_j,\; XY_j,\; XZ_j,\; YY_j,\; YZ_j,\; ZZ_j,\; MX_j,\; \cdots \\ & \qquad \cdots\; MY_j,\; MZ_j,\; M_j,\; Ia_j,\; Fv_j,\; Fc_j\;]^\top, \end{aligned} \tag{2}$$

where $XX_j, XY_j, XZ_j, YY_j, YZ_j, ZZ_j$ are the elements of the inertia tensor $\boldsymbol{L}_j$ of link j, expressed at the link origin $\mathcal{L}_j$, Ia_j refers to inertia of the actuator and transmission system, M_j is the link mass, X_j, Y_j, Z_j are the coordinates of the link Center-of-Mass in $\mathcal{L}_j$ and MX_j, MY_j, MZ_j are the corresponding first moments [10]. Let $\boldsymbol{h}(\boldsymbol{\chi}, \boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}^n$ be the vector that regroups the Coriolis, gravitational and frictional effects, namely $\boldsymbol{h}(\boldsymbol{\chi}, \boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{C}(\boldsymbol{\chi}, \boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{\chi}, \boldsymbol{q}) + \boldsymbol{\zeta}(\boldsymbol{\chi}, \dot{\boldsymbol{q}})$. The direct dynamic model (DDM), allowing to calculate the joint accelerations $\ddot{\boldsymbol{q}}$ as a function of the joint positions $\boldsymbol{q}$, joint velocities $\dot{\boldsymbol{q}}$, generalized forces $\boldsymbol{\tau}_{idm}$ and of the vector $\boldsymbol{\chi}$ of dynamic parameters, can then be written as

$$\ddot{\boldsymbol{q}} = \boldsymbol{M}^{-1}(\boldsymbol{\chi}, \boldsymbol{q}) \left(\boldsymbol{\tau}_{idm} - \boldsymbol{h}(\boldsymbol{\chi}, \boldsymbol{q}, \dot{\boldsymbol{q}})\right) . \tag{3}$$

From equation (3), it appears that the DDM has a nonlinear dependance in terms of the robot's state vector $\boldsymbol{x}$, defined as $\boldsymbol{x} = [\dot{\boldsymbol{q}}^\top \; \boldsymbol{q}^\top]^\top \in \mathbb{R}^{2n}$, and of its inertial parameter vector $\boldsymbol{\chi}$. By contrast, the IDM has the relevant property of being linear in the $\boldsymbol{\chi}$, and can thus be reformulated as

$$\boldsymbol{\tau}_{idm} = \boldsymbol{Y}_{\boldsymbol{\chi}}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q})\boldsymbol{\chi} \, , \tag{4}$$

where $\boldsymbol{Y}_{\boldsymbol{\chi}}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q}) = \partial \boldsymbol{\tau}_{idm} / \partial \boldsymbol{\chi} \in \mathbb{R}^{n \times p}$ denotes the closed-form expression of the Jacobian matrix of $\boldsymbol{\tau}_{idm}$ with respect to $\boldsymbol{\chi}$, often referred to as the *model regressor*. Note that the detailed derivation of the full robot dynamic model and of the corresponding regressor matrix is made available to the reader in Appendix A. It is worth noting that the equality in equation (4) only holds provided that the friction term $\boldsymbol{\zeta}(\boldsymbol{\chi}, \dot{\boldsymbol{q}})$ is also linear with respect to the parameter vector[1]$\boldsymbol{\chi}$ and that the vectors $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$ and $\boldsymbol{\tau}_{idm}$ are noise-free. Under these assumptions, dynamic parameters identification can be considered as an inverse problem. Sampling (4) at multiple different time epochs along a given state trajectory results in an over-determined *observation system* of linear equations in $\boldsymbol{\chi}$. Considering that such system is inherently rank defficient, convergence to a statistically and physically consistent solution turns out to be difficult in practice. This will be further discussed in the next section.

2.1.2. Resolution of Data and Structural Rank Deficiencies

As explained in [11], the rank of the sampled observation system is a function of two factors, namely the kinematics of the considered robot and nature of the collected data samples. As a matter of fact, the intrinsic geometrical properties of robot manipulators naturally imply that some of the dynamic parameters within the $\boldsymbol{\chi}$ vector simply cannot be identified, as they have strictly no influence on the actual movements of the robot, regardless of the followed trajectory [12]. In the same manner, some parameters can only be identified jointly because of their combined influence on the system. From a formal point of view, these issues can be characterized as a *structural rank deficiency* of the regression matrix $\boldsymbol{Y}_{\boldsymbol{\chi}}$. To solve this issue, it is necessary to perform a set of column-rearrangements within $\boldsymbol{Y}_{\boldsymbol{\chi}}$, eventually leading to the set of $b \leq p$ parameters referred to as the *base parameters*, and that are actually linear combinations of the standard parameters $\boldsymbol{\chi}$. The other aspect of the problem lies in the fact that

[1] In case this is not verified (4) is only a first-order approximation of a potentially much more complex system.

data rank deficiency of the observation system is a direct consequence of noisy or improper data samples that do not adequately *excite* the model parameters (i.e., that do not sufficiently reveal the influence of these parameters on the system's dynamics). This issue can most of the time be solved by having the robot track a trajectory that is specifically designed to excite the base parameters [13, 14, 15, 16, 17, 18, 19, 20, 21]. A possible approach to the design of such a trajectory is to parametrize Fourier series or b-splines at the joint level. Data rank deficiency can then be formulated as a nonlinear optimization problem, adjusting the trajectory parameters based, most of the time, on the conditioning of the observation matrix.

Base inertial parameters are defined by [22] and [23] as the minimal set of dynamic parameters which is sufficient to completely describe the – intrinsically constrained – dynamics of a robot mechanism. As explained in [10], the base parameter vector $\boldsymbol{\beta} \in \mathbb{R}^b$ can be obtained by performing a set of rearrangements in the symbolic expressions of (4)[2]. In [24], an alternative computation method was proposed based on Fourier series decomposition of the robot dynamic equations. Dedicated numerical methods based, for example, on QR decomposition can also be used. Consider the observation matrix $\boldsymbol{W_\chi}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q}) \in \mathbb{R}^{(n \cdot N) \times p}$ constructed by stacking the samples of $\boldsymbol{Y_\chi}$ obtained from a randomly generated set of $N >> 1$ distinct joint values:

$$\boldsymbol{W_\chi P} = \boldsymbol{Q} \begin{bmatrix} \boldsymbol{R} \\ \\ \mathbf{0}_{(r-b)\times p} \end{bmatrix} \tag{5}$$

Where for $r = nN$, $\boldsymbol{Q} \in \mathbf{O}(r)$, $\boldsymbol{R} \in \mathbb{R}^{b\times p}$ is an upper-triangular matrix of rank $b \leq p$ and $\boldsymbol{P} \in \mathbf{O}(p)$ is a permutation matrix chosen – *by default* – so that the diagonal values of $\boldsymbol{R}$ are arranged in decreasing order. Denoting by $\overline{\boldsymbol{P}} \in \mathbb{R}^{p\times b}$ and $\underline{\boldsymbol{P}} \in \mathbb{R}^{p\times(p-b)}$ the first b and last $p - b$ columns of $\boldsymbol{P}$ leads to

$$\begin{aligned}
\underbrace{\begin{bmatrix} \boldsymbol{W_\chi}\overline{\boldsymbol{P}} & \boldsymbol{W_\chi}\underline{\boldsymbol{P}} \end{bmatrix}}_{\boldsymbol{W_\chi P}} &= \underbrace{\begin{bmatrix} \overline{\boldsymbol{Q}} & \underline{\boldsymbol{Q}} \end{bmatrix}}_{\boldsymbol{Q}} \begin{bmatrix} \overline{\boldsymbol{R}} & \underline{\boldsymbol{R}} \\ \\ \mathbf{0}_{(r-b)\times p} & \end{bmatrix} && \text{(6a)} \\
&= \begin{bmatrix} \overline{\boldsymbol{Q}}\,\overline{\boldsymbol{R}} & \overline{\boldsymbol{Q}}\,\underline{\boldsymbol{R}} \end{bmatrix} && \text{(6b)} \\
&= \overline{\boldsymbol{Q}}\,\overline{\boldsymbol{R}} \begin{bmatrix} \mathbf{1}_{b\times b} & \overline{\boldsymbol{R}}^{-1}\underline{\boldsymbol{R}} \end{bmatrix} && \text{(6c)} \\
&= \boldsymbol{W_\chi}\overline{\boldsymbol{P}} \begin{bmatrix} \mathbf{1}_{b\times b} & \overline{\boldsymbol{R}}^{-1}\underline{\boldsymbol{R}} \end{bmatrix} && \text{(6d)}
\end{aligned}$$

where $\overline{\boldsymbol{R}} \in \mathbb{R}^{b\times b}$ and $\underline{\boldsymbol{R}} \in \mathbb{R}^{b\times(p-b)}$ respectively denote the first b and the last $p - b$ columns of $\boldsymbol{R}$ and where $\mathbf{1}_{b\times b}$ is the $b \times b$ identity matrix.

[2] Note that in this context, some of the performed symbolic simplifications are only made possible by the use of proximal DH convention.

Since $\boldsymbol{P}$ is orthogonal, one can write

$$\boldsymbol{W}_{\boldsymbol{\chi}}\boldsymbol{\chi} = \begin{bmatrix} \boldsymbol{W}_{\boldsymbol{\chi}}\overline{\boldsymbol{P}} & \boldsymbol{W}_{\boldsymbol{\chi}}\underline{\boldsymbol{P}} \end{bmatrix} \begin{bmatrix} \overline{\boldsymbol{\chi}} \\ \underline{\boldsymbol{\chi}} \end{bmatrix} \tag{7}$$

with $\overline{\boldsymbol{\chi}} = \overline{\boldsymbol{P}}^{\top}\boldsymbol{\chi}$ and $\underline{\boldsymbol{\chi}} = \underline{\boldsymbol{P}}^{\top}\boldsymbol{\chi}$, yielding

$$\begin{aligned} \boldsymbol{W}_{\boldsymbol{\chi}}\boldsymbol{\chi} &= \boldsymbol{W}_{\boldsymbol{\chi}}\overline{\boldsymbol{P}} \begin{bmatrix} \mathbf{1}_{b\times b} & \overline{\boldsymbol{R}}^{-\mathbf{1}}\underline{\boldsymbol{R}} \end{bmatrix} \begin{bmatrix} \overline{\boldsymbol{\chi}} \\ \underline{\boldsymbol{\chi}} \end{bmatrix} && \text{(8a)} \\ &= \boldsymbol{W}_{\boldsymbol{\chi}}\overline{\boldsymbol{P}}\,\boldsymbol{\beta} && \text{(8b)} \end{aligned}$$

and hence the corresponding *non-bijective* mapping between the base parameter vector $\boldsymbol{\beta}$ and the standard parameter vector $\boldsymbol{\chi}$

$$\boldsymbol{\beta} = \underbrace{\begin{bmatrix} \overline{\boldsymbol{P}}^{\top} & \overline{\boldsymbol{R}}^{-\mathbf{1}}\underline{\boldsymbol{R}}\,\underline{\boldsymbol{P}}^{\top} \end{bmatrix}}_{\mho}\boldsymbol{\chi} \tag{9}$$

Note that – as explained in [25] – a bijective map $\mathfrak{m}$ can still be defined between $(\boldsymbol{\beta}, \underline{\boldsymbol{\chi}})$ and $\boldsymbol{\chi}$ as

$$\begin{bmatrix} \boldsymbol{\beta} \\ \underline{\boldsymbol{\chi}} \end{bmatrix} = \mathfrak{m}(\boldsymbol{\chi}) = \underbrace{\begin{bmatrix} \mathbf{1}_{b\times b} & \overline{\boldsymbol{R}}^{-\mathbf{1}}\underline{\boldsymbol{R}} \\ \mathbf{0}_{(p-b)\times b} & \mathbf{1}_{(p-b)\times(p-b)} \end{bmatrix} \boldsymbol{P}^{\top}}_{\boldsymbol{\Omega}}\,\boldsymbol{\chi} \tag{10}$$

in which case the inverse mapping is

$$\boldsymbol{\chi} = \mathfrak{m}^{-1}(\boldsymbol{\beta}, \underline{\boldsymbol{\chi}}) = \underbrace{\boldsymbol{P} \begin{bmatrix} \mathbf{1}_{b\times b} & -\overline{\boldsymbol{R}}^{-\mathbf{1}}\underline{\boldsymbol{R}} \\ \mathbf{0}_{(p-b)\times b} & \mathbf{1}_{(p-b)\times(p-b)} \end{bmatrix}}_{\boldsymbol{\Omega}^{-1}} \begin{bmatrix} \boldsymbol{\beta} \\ \underline{\boldsymbol{\chi}} \end{bmatrix} \tag{11}$$

The problem of robot identification can hence be formulated as estimating the value of $\boldsymbol{\beta}$ such that the dynamic behavior of the model matches that of the actual robot while it is tracking a persistent exciting trajectory. It is worth noting that as some of the base parameters may only have a minor influence on the robot dynamics, they can be neglected in practice. The corresponding reduced set of parameters, referred to as "*essential*" parameters in [26] can be identified using singular value decomposition [3] but will not be considered in this work.

[3] Non-essential parameter estimates can easily be identified in a Monte Carlo Simulation (MCS) as they typically exhibit a high variance.

2.1.3. Low-Level Control Strategy

Robots being double-integrator systems, they are naturally unstable in open-loop and must therefore be operated and identified in closed-loop. Since most electrically actuated robots use Direct-Current (DC) or Brushless Direct Current (BLDC) motors, the control structure at the joint level usually consists in a cascade of PD or PID regulators, the position and velocity loops providing a reference for the current-torque loop (c.f. [10]). As pointed out in [27, 28], provided that the low-level current loop has a sufficient bandwidth,[4] and considering the linear mapping between the current and the torque within DC and BLDC motors, its transfer function can be expressed as a static gain in the characteristic frequency range of the rigid robot dynamics[5]. In the case of a PD controller, the resulting control torque applied to the robot can be then typically be computed as:

$$\boldsymbol{\tau}_{ctrl} = \boldsymbol{K}_{\tau}\boldsymbol{K}_{p}\left(\boldsymbol{q}_{d} - \boldsymbol{q}\right) + \boldsymbol{K}_{\tau}\boldsymbol{K}_{d}\left(\dot{\boldsymbol{q}}_{d} - \dot{\boldsymbol{q}}\right) + \boldsymbol{\tau}_{ff} \tag{12}$$

where $\boldsymbol{K}_{\tau}$, $\boldsymbol{K}_{p}$ and $\boldsymbol{K}_{d} \in \mathbb{R}^{n\times n}$ are the diagonal drive, position and velocity gain matrices and $\boldsymbol{\tau}_{ff}$ is a feed-forward term. Note that the drive gain matrix $\boldsymbol{K}_{\tau}$ contains the combined influence of the static gains $\boldsymbol{K}_{v}$ of the current amplifiers, gear ratios $\boldsymbol{K}_{r}$ and electromagnetic motor torque constants $\boldsymbol{K}_{m}$, and can thus be written as $\boldsymbol{K}_{\tau} = \boldsymbol{K}_{v}\boldsymbol{K}_{r}\boldsymbol{K}_{m}$ (c.f. [27]).

2.2. Inertial Parameter Identification Algorithms

This section aims at presenting some of the most frequently used algorithms in the context of inertial parameter identification of robotic systems. The particularities of each approach are discussed from a theoretical point of view and compared to each other.

2.2.1. Ordinary, Weighted and Iteratively Re-weighted Least Squares (IDIM-OLS, -WLS, -IRLS)

Let $\boldsymbol{Y}_{\beta}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q})$ be the regression matrix of a considered robot relative to its base inertial parameters. $\boldsymbol{Y}_{\beta}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q})$ is obtained by column rearrangements of the robot regressor matrix $\boldsymbol{Y}_{\chi}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q})$ following $\boldsymbol{Y}_{\beta}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q}) = \boldsymbol{Y}_{\chi}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q})\overline{\boldsymbol{P}}$, with $\overline{\boldsymbol{P}}$ the permutation matrix defined in 2.1.2. Then (1) can be rewritten in the following form:

$$\boldsymbol{\tau}_{idm} = \boldsymbol{Y}_{\beta}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q})\boldsymbol{\beta}\,, \tag{13}$$

In practice, because of the uncertainties caused by measurement noises and modeling errors, the actual torque $\boldsymbol{\tau}$ differs from $\boldsymbol{\tau}_{idm}$ by an error $\boldsymbol{e} \in \mathbb{R}^{n}$ so that (13) becomes

$$\boldsymbol{\tau} = \boldsymbol{Y}_{\beta}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q})\boldsymbol{\beta} + \boldsymbol{e}\,. \tag{14}$$

[4] Typically above 500Hz. This is usually verified on most Direct current (DC) or Brushless Direct-Current (BLDC) actuators since the torque reference is adjusted at the PWM frequency, i.e 16-40kHz.

[5] Typically less than 10Hz for industrial robots as explained in [27, 28].

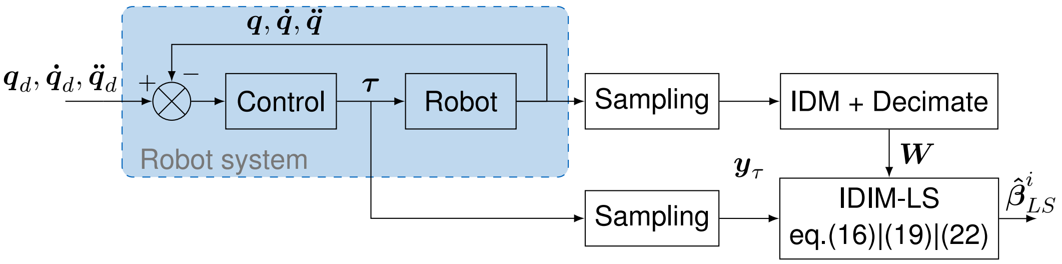

$\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$
$\boldsymbol{q}_d, \dot{\boldsymbol{q}}_d, \ddot{\boldsymbol{q}}_d$
+
−
Control
$\boldsymbol{\tau}$
Robot
Robot system
Sampling
IDM + Decimate
$\boldsymbol{W}$
$\boldsymbol{y}_\tau$
Sampling
IDIM-LS
eq.(16)|(19)|(22)
$\hat{\boldsymbol{\beta}}^i_{LS}$

Figure 2 Block diagram of the IDIM-OLS, -WLS, -IRLS and -TLS identification methods.

Equation (14) represents the Inverse Dynamic Identification Model (IDIM) relation (c.f. [27] and the references therein). The IDIM is sampled at a rate f while the robot is tracking exciting trajectories during an experiment. For N collected samples, an over-determined linear system of $n \cdot N$ equations and b unknowns is obtained

$$\boldsymbol{y}_\tau = \boldsymbol{W}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q})\boldsymbol{\beta} + \boldsymbol{\varepsilon}\,, \tag{15}$$

where $\boldsymbol{y}_\tau \in \mathbb{R}^{n \cdot N}$ is the sampled vector of $\boldsymbol{\tau}$; $\boldsymbol{W}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q}) \in \mathbb{R}^{(n \cdot N) \times b}$ is the sampled matrix of $\boldsymbol{Y}_{\boldsymbol{\beta}}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q})$[6], referred to as *observation matrix*; and $\boldsymbol{\varepsilon} \in \mathbb{R}^{n \cdot N}$ is the sampled vector of $\boldsymbol{e}$. Note that ε is here assumed to be serially uncorrelated, zero-mean and heteroskedastic, with a diagonal covariance matrix $\boldsymbol{\Sigma}$. This choice is justified in [29, 30] by the fact that robots are nonlinear multi-input multi-output (MIMO) systems. The most popular approach to solving (15) consists in computing the weighted least-squares estimate $\hat{\boldsymbol{\beta}}_{WLS}$ and associated covariance matrix $\boldsymbol{\Sigma}_{WLS}$

$$\begin{aligned} \hat{\boldsymbol{\beta}}_{WLS} &= (\boldsymbol{W}^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{W})^{-1} \boldsymbol{W}^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{y}_\tau\,, \\ \boldsymbol{\Sigma}_{WLS} &= \left(\boldsymbol{W}^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{W}\right)^{-1}\,, \end{aligned} \tag{16}$$

The diagonal terms Σ_{jj} of $\boldsymbol{\Sigma}$ can be evaluated from the ordinary least-squares solution of (15), following the guidelines of [11][7]

$$\forall j \in \{1 \cdots n\},\ \Sigma_{jj} = \frac{\left\|\varepsilon_{OLS_j}\right\|}{N - b}. \tag{17}$$

where ε_{OLS_j} denotes the IDIM-OLS sampled error vector for joint j. This method is both simple to implement and computationally efficient. In addition, it was successfully applied to multiple existing systems such as cars [31], electrical motors [32] or compactors [33]. It is worth noting that the vulnerability of OLS and WLS to outliers can be addressed using a specific Huber estimator (c.f. [34, 35]). This is referred to as the Iteratively Re-weighted Least-Squares (IRLS) in [36]. The resulting IDIM-IRLS method takes the form of an iterative process, which consists in applying additional penalty to the outliers, in the form of a dedicated weight vector $\boldsymbol{v}^i \in \mathbb{R}^{n \cdot N}$ and a weight matrix $\boldsymbol{\Upsilon}^i = [\boldsymbol{v}^i, \cdots, \boldsymbol{v}^i] \in \mathbb{R}^{(n \cdot N) \times b}$ for each iteration

[6] For the sake of compactness, $\boldsymbol{W}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q})$ will be noted $\boldsymbol{W}$ in the rest of this work.

[7] A correlation between the measured joint torque signals will result in non-negligible off-diagonal terms in $\boldsymbol{\Sigma}$. However these terms can still be estimated using for instance the method of [21].

i, in order to eventually mitigate their contribution to the final result

$$\begin{aligned}\boldsymbol{W}^{*i} &= \boldsymbol{\Upsilon}^{i} \circ \boldsymbol{W} \\ \boldsymbol{y}_{\boldsymbol{\tau}}^{*i} &= \boldsymbol{v}^{i} \circ \boldsymbol{y}_{\boldsymbol{\tau}}\end{aligned} \tag{18}$$

where $\circ$ refers to the element-wise multiplication operator. Accordingly the IDIM-IRLS estimate $\hat{\boldsymbol{\beta}}_{IRLS}^{i}$ at iteration i is given by

$$\hat{\boldsymbol{\beta}}_{IRLS}^{i} = (\boldsymbol{W}^{*i\top}\boldsymbol{\Sigma}^{-1}\boldsymbol{W}^{*i})^{-1}\boldsymbol{W}^{*i\top}\boldsymbol{\Sigma}^{-1}\boldsymbol{y}_{\tau}^{*i}\,. \tag{19}$$

The weight vector $\boldsymbol{v}^{i}$ is updated following

$$\boldsymbol{v}^{i} = \mathrm{Min}\,(\boldsymbol{v}^{i-1}, \Lambda(\boldsymbol{\varepsilon}_{IRLS}^{i})) \tag{20}$$

where $\mathrm{Min}(\cdot)$ is the element wise $\min$ operator and $\Lambda : \mathbb{R}^{n\cdot N} \to [0,1]^{n\cdot N}$ is a tailor made weight function. The process is iterated until convergence of the weights. This method was successfully tested on robotic systems in [37, 21]. It is critically important to note that the LS estimates in general, will be unbiased if and only if the observation matrix $\boldsymbol{W}$ is uncorrelated with the error term $\boldsymbol{\varepsilon}$, or in other words, that the following equality holds

$$\mathrm{E}(\boldsymbol{W}^{\top}\boldsymbol{\varepsilon}) = \boldsymbol{0}\,. \tag{21}$$

Unfortunately, the presence of uncorrelated random components within the observation matrix does not allow this hypothesis to be validated in practice. Noise sensitivity is especially problematic in the context of robotic systems as the joint accelerations, obtained by double time differentiation of the noisy encoder data, often exhibit poor Signal-to-Noise Ratio (SNR) with multiple outliers. A workaround to this issue is to filter the joint measurement signals as suggested by [11] where a pragmatic and tailor-made data-filtering process is proposed. Nevertheless, this requires the knowledge of the bandwidth of the position closed-loop and special attention due to the bias induced by the filter, see [38, 27, 39] for more details. It is worth noting that in the case of a periodic excitation trajectory with a known characteristic frequency, it is possible to use Fourier analysis tools in order to perform frequency domain filtering as suggested in [40, 17]. The other alternative is to use identification methods that are robust against a violation of (21).

2.2.2. Total Least-Squares (IDIM-TLS)

The issue of noisy observation matrix can, in theory, be tackled using the Total Least-Squares (IDIM-TLS) approach. As exposed in [41] and [42], the TLS estimate $\hat{\boldsymbol{\beta}}_{TLS}$ can be computed using a singular value decomposition of the augmented matrix $\boldsymbol{X} = [\boldsymbol{W}\ \boldsymbol{y}_{\tau}] \in \mathbb{R}^{(n\cdot N)\times(b+1)}$ as

$$\boldsymbol{X} = [\boldsymbol{U}_{W}\ \boldsymbol{U}_{y}]\begin{bmatrix}\boldsymbol{S} & 0 \\ 0 & S_{min}\end{bmatrix}\begin{bmatrix}\boldsymbol{V}_{WW} & \boldsymbol{V}_{Wy} \\ \boldsymbol{V}_{yW} & V_{yy}\end{bmatrix}^{\top}, \tag{22a}$$

$$\hat{\boldsymbol{\beta}}_{TLS} = -\boldsymbol{V}_{Wy}V_{yy}^{-1} \tag{22b}$$

where $\boldsymbol{V}_{Wy} \in \mathbb{R}^b$ and $V_{yy} \in \mathbb{R}^\star$. Following [43, 44], the covariance $\boldsymbol{\Sigma}_{TLS}$ of the TLS estimate $\hat{\boldsymbol{\beta}}_{TLS}$ can be approximated as

$$\boldsymbol{\Sigma}_{TLS} \approx \left(1 + \left\|\hat{\boldsymbol{\beta}}_{TLS}\right\|_2^2\right) \frac{S_{min}^2}{n \cdot N} \left(\boldsymbol{W}^\top \boldsymbol{W} - S_{min}^2 \boldsymbol{I}\right)^{-1} \tag{23}$$

where $S_{min} \in \mathbb{R}$ is the minimum non-zero singular value of $\boldsymbol{X}$. Only a few research papers actually explore the potential use of total least-squares in the context of robot dynamic parameters identification. In [29], the authors observed that the IDIM-TLS estimate is biased if the joint data is not suitably filtered. Moreover, they noted that although TLS convergence occurs with adequate data filtering, it does not outperform ordinary and weighted least-squares. Motivated by [45] and [46], in [47] part 4.2.6, the author raised some general comments on the IDIM-TLS method and stressed that its disappointing performances can be explained by the fact that it is based on the hypothesis that the noise terms have all the same variances whereas this assumption is violated in the case of an improper data-filtering according to [48]. It would in fact be more appropriate to use the Generalized Total Least-Squares (GTLS) method popularized by Van Huffel and Vandewalle, [49, 43]. However, in order to properly use the GTLS method, the covariance matrix of noises involved in $\boldsymbol{W}$ must be known. Unfortunately, if such information is easily accessible for linear systems, it is not for robots. Indeed, besides the noise amplification effect induced by the process of numerical time derivation of the joint angles – used for joint velocity and acceleration computation – the IDM involves non-linear functions such as the sine, cosine, square or sign operators, making the calculation of this covariance matrix difficult, if not intractable. This comment explains the reason why the TLS or GTLS methods are seldom employed in the context of robot identification (see e.g. [41, 50]) whereas they are widely considered in signal processing, see e.g. [51] and the references therein.

2.2.3. Inverse Dynamics Identification Model (IDIM) with Instrumental Variables (IV)

First introduced by [52] in the context of econometrics, the Instrumental Variables (IV) approach to parameter identification consists in defining an *instrument matrix* $\boldsymbol{Z} \in \mathbb{R}^{(n \cdot N) \times b}$ such that

$$\mathrm{E}(\boldsymbol{Z}^\top \boldsymbol{W}) \quad \text{is full column rank,} \tag{24a}$$

$$\mathrm{E}(\boldsymbol{Z}^\top \boldsymbol{\varepsilon}) = 0, \tag{24b}$$

which means that $\boldsymbol{Z}$ is both well correlated with the observation matrix $\boldsymbol{W}$ and uncorrelated with the error term $\boldsymbol{\varepsilon}$, see e.g. [53, 54, 29]. In this case, one obtains

$$\boldsymbol{Z}^\top \boldsymbol{y}_\tau = \boldsymbol{Z}^\top \boldsymbol{W}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q})\boldsymbol{\beta} + \boldsymbol{Z}^\top \boldsymbol{\varepsilon} \tag{25}$$

and the IV estimates given by

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{Z}^\top \boldsymbol{W})^{-1} \boldsymbol{Z}^\top \boldsymbol{y}_\tau \tag{26}$$

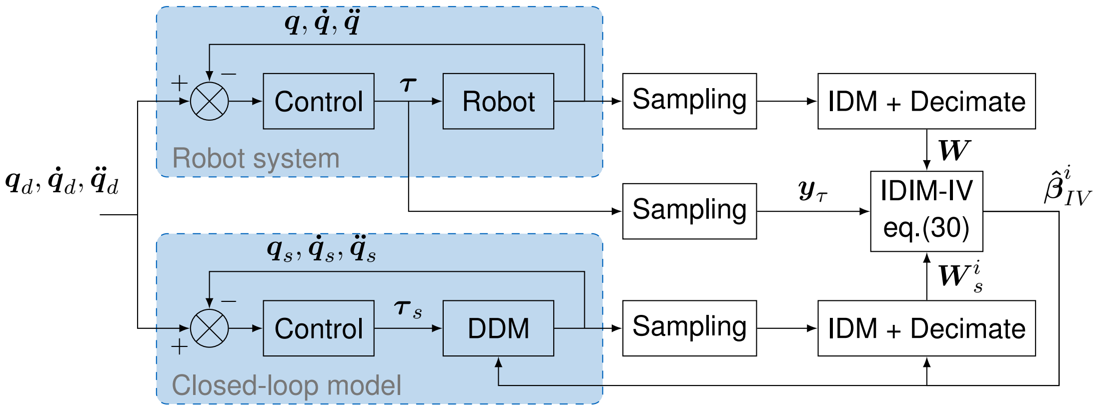

$\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$
+
−
Control
$\boldsymbol{\tau}$
Robot
Robot system
$\boldsymbol{q}_d, \dot{\boldsymbol{q}}_d, \ddot{\boldsymbol{q}}_d$
Sampling
IDM + Decimate
$\boldsymbol{W}$
Sampling
$\boldsymbol{y}_\tau$
IDIM-IV eq.(30)
$\hat{\boldsymbol{\beta}}^i_{IV}$
$\boldsymbol{q}_s, \dot{\boldsymbol{q}}_s, \ddot{\boldsymbol{q}}_s$
−
+
Control
$\boldsymbol{\tau}_s$
DDM
$\boldsymbol{W}^i_s$
Sampling
IDM + Decimate
Closed-loop model

Figure 3 Block diagram of the IDIM-IV identification method.

are consistent. The use of IV for identification of robotic systems is reported in several approaches (see for instance [55, 56, 29, 57, 39, 58, 59] and the references therein). The critical issue here lies in the construction of the instrument matrix $\boldsymbol{Z}$ fulfilling (24a) and (24b). In this context, an ideal candidate for the $\boldsymbol{Z}$ instrument matrix appears to be the observation matrix denoted $\boldsymbol{W}_s$ filled with simulated data that are the outputs of an auxiliary model, [54]. As explained in [29], for robot identification, the auxiliary model is the direct dynamic model (DDM) which is simulated assuming the same controller and reference trajectories that the one applied to the actual robot (c.f. Fig. 3), and by using the IV estimate $\hat{\boldsymbol{\beta}}^{i-1}_{IV}$ obtained at the previous iteration. This defines an iterative algorithm. At iteration i and time epoch t_k, the vector of simulated joint accelerations $\ddot{\boldsymbol{q}}^i_s(t_k)$ is computed as

$$\ddot{\boldsymbol{q}}^i_s(t_k) = \boldsymbol{M}^{-1}(\hat{\boldsymbol{\beta}}^{i-1}_{IV}, \boldsymbol{q}^i_s(t_k)) \left(\boldsymbol{\tau}^i_s(t_k) - \boldsymbol{h}(\hat{\boldsymbol{\beta}}^{i-1}_{IV}, \boldsymbol{q}^i_s(t_k), \dot{\boldsymbol{q}}^i_s(t_k)) \right) . \tag{27}$$

By successive numerical integration of eq. (27) one can obtain the vector of joint simulated positions and velocities denoted $(\boldsymbol{q}^i_s, \dot{\boldsymbol{q}}^i_s)$, respectively, from which it is then possible to build the instrument matrix $\boldsymbol{Z}^i \in \mathbb{R}^{(n \cdot N) \times b}$ by stacking the regression matrices $\boldsymbol{Y}_\beta(\boldsymbol{q}^i_s, \dot{\boldsymbol{q}}^i_s, \ddot{\boldsymbol{q}}^i_s)$ obtained from the samples of $\boldsymbol{q}^i_s, \dot{\boldsymbol{q}}^i_s, \ddot{\boldsymbol{q}}^i_s$ at each t_k as

$$\boldsymbol{Z}^i = \boldsymbol{W}^i_s = \boldsymbol{W}\left(\boldsymbol{q}^i_s, \dot{\boldsymbol{q}}^i_s, \ddot{\boldsymbol{q}}^i_s\right) . \tag{28}$$

Assuming that there are no modeling errors, the simulated joint positions, velocities and accelerations tend to the noise-free estimates denoted $(\boldsymbol{q}_{nf}, \dot{\boldsymbol{q}}_{nf}, \ddot{\boldsymbol{q}}_{nf})$, respectively, yielding

$$\boldsymbol{Z}^i = \boldsymbol{W}^i_{nf} = \boldsymbol{W}\left(\boldsymbol{q}^i_{nf}, \dot{\boldsymbol{q}}^i_{nf}, \ddot{\boldsymbol{q}}^i_{nf}\right) \forall i \, . \tag{29}$$

Note that $\boldsymbol{Z}^i$ given by eq. (29) naturally complies with the conditions eq. (24a) and eq. (24b). In [29, 57, 39, 58] the IDIM-IV estimates are formulated in a weighted recursive manner as:

$$\hat{\boldsymbol{\beta}}^i_{IV} = (\boldsymbol{W}^{i\top}_s \boldsymbol{\Sigma}^{-1} \boldsymbol{W})^{-1} \boldsymbol{W}^{i\top}_s \boldsymbol{\Sigma}^{-1} \boldsymbol{y}_\tau \tag{30}$$

where $\boldsymbol{W}^i_s$ is given by (28).

Once the IDIM-IV method has converged, the covariance matrix of the IDIM-IV estimates is given by

$$\boldsymbol{\Sigma}_{IV} = \left(\boldsymbol{W}_s^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{W}\right)^{-1}. \tag{31}$$

It should be emphasized that the consistency of the closed-loop simulation in IDIM-IV relies on the assumption that the controller of the real robot and that of the simulated robot are the same. A difference in the control strategy will result in a different command $\boldsymbol{\tau}_s$ being issued for a given state, eventually resulting in a bias in the parameter estimate. The results of [29, 57, 47, 39, 58] suggest that the convergence of the IDIM-IV method is more robust against noise than IDIM-OLS/-WLS/-TLS and also faster[8] than Output-Error methods presented later in this thesis. More generally since both the IDM and DDM are calculated from Newton's laws, it seems more natural to simulate the DDM in order to construct $\boldsymbol{Z}$ rather than computing the covariance matrix of noise terms involved in $\boldsymbol{W}$ and $\boldsymbol{y}$. In [57, 39] the authors proved that IDIM-IV gives excellent results provided that the low-level controller is well-identified. Nevertheless, the robustness against high noise levels has not yet been suitably investigated and would deserve deeper treatment.

2.2.4. The Maximum Likelihood (ML) Identification Method

Investigated in [16, 60, 40, 61], the Maximum Likelihood (ML) parameter identification algorithm aims at tackling the issue of noisy torque and joint angle measurements. These works are the first ones actually addressing the issue of noisy observation matrices. In [60], the authors present an improvement of the original ML approach presented in [16]. The ML criterion they adopt can be formulated as

$$\hat{\boldsymbol{\beta}}_{ML} = \arg\min_{\boldsymbol{\beta}} \frac{1}{2} \sum_{k=1}^{N} \boldsymbol{\varepsilon}^\top(t_k) \left(\boldsymbol{G}_k \boldsymbol{\sigma}_k^2 \boldsymbol{G}_k^\top\right)^{-2} \boldsymbol{\varepsilon}(t_k), \tag{32}$$

where $\boldsymbol{\sigma}_k^2 \in \mathbb{R}^{4n}$ denotes the diagonal variance matrix of the k^{th} sample[9], $\boldsymbol{\varepsilon}(t_k)$ is the error at the k^{th} torque sample defined in (15); and $\boldsymbol{G}_k = \partial \boldsymbol{\varepsilon}(t_k)/\partial \boldsymbol{s}_k \in \mathbb{R}^{n \times 4 \cdot n}$ is the Jacobian matrix of this error relative to the measurement vector $\boldsymbol{s}_k = \left[\boldsymbol{q}(t_k)^\top \ \dot{\boldsymbol{q}}(t_k)^\top \ \ddot{\boldsymbol{q}}(t_k)^\top \ \boldsymbol{\tau}(t_k)^\top\right]^\top \in \mathbb{R}^{4 \cdot n}$. Because $\boldsymbol{G}_k$ involves the IDM, this ML approach can be called IDIM-ML. In the IDIM-ML approach, the authors suggest to construct the vector $\boldsymbol{s}_k$ by averaging the original measurements of $\boldsymbol{q}(t_k),\ \dot{\boldsymbol{q}}(t_k),\ \ddot{\boldsymbol{q}}(t_k)$ and $\boldsymbol{\tau}(t_k)$ over N_{real} realizations. Then, $\boldsymbol{s}_k$ is used to construct the following observation matrix denoted by $\overline{\boldsymbol{W}}$. It must be noticed that, provided N_{real} is big enough and the variances of the original measurements are finite, the Lyapunov criterion ensures that $\boldsymbol{s}_k$ is close to a Gaussian distribution. Note also that it can be even further assumed that $\boldsymbol{s}_k$ tends to the noise-free original measurements as N_{real} grows and the noise level is *reasonable* which means that one has $\overline{\boldsymbol{W}} \to \boldsymbol{W}_{nf}$. Interestingly, this ML approach is somehow related with the IDIM-IV method developed in [29]. Indeed, the authors suggest to build an instrumental matrix such that $\boldsymbol{Z} = \boldsymbol{W}_{nf}$ by using simulated data and the IDM

[8] It is worth noting that convergence is slower than IDIM-OLS since simulation of the DDM is required.

[9] This allows in particular to account for the effects of time differentiation in the joint signal in terms of noise amplification.

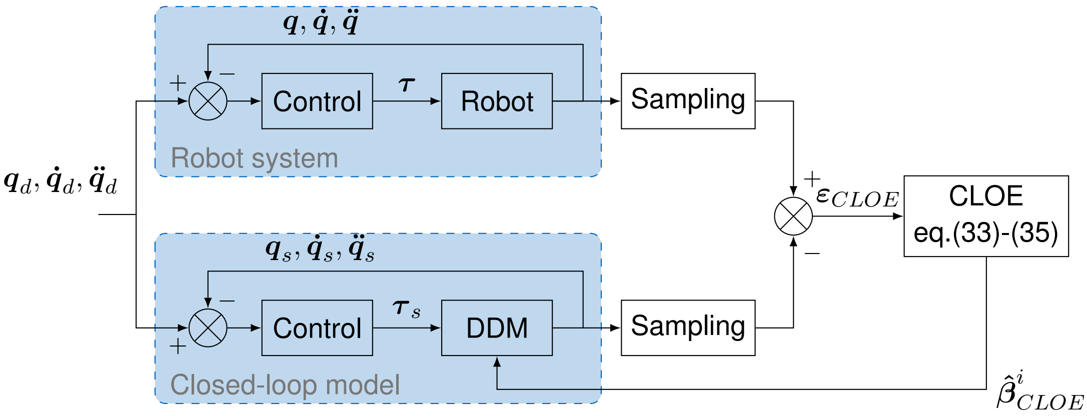


Figure 4 Block diagram of the CLOE identification method.

while in [60] the authors suggest to construct $\overline{W}$ with $\overline{W} \to W_{nf}$. Furthermore, in [62], the author establishes some interesting relationships between IV and ML approaches. It is thus expected that IDIM-IV and ML approaches give similar results. However, in [16] a *prewhitening* process has been carried out in order to remove all the coloring and correlation of the measurement noise. Without this *prewhitening* process which can be connected with the use of the *decimate* filter, the statistical assumptions made on the noise may be violated making the ML approach potentially unable to provide consistent estimates. Moreover, it should also be stressed that the joint torques in these approaches are measured via *current-shunt monitors* instead of being the outputs of low-level controllers as usually done in [26]. So, if the ML method does not require the simulation of the DDM, it could prove more sensitive to noise and is more time-consuming than IDIM-IV.

2.2.5. Closed-Loop Output Error (CLOE)

As indicated by its name, an Output Error method consists in finding the parameter vector $\boldsymbol{\beta}$ that minimizes the value function $J(\boldsymbol{\beta})$ defined as the squared L2-norm of the error between the output $\boldsymbol{y}$ of a system to be identified and the output $\boldsymbol{y}_s$ of its model (see [63, 64] and the references therein)

$$J(\boldsymbol{\beta}) = \|\boldsymbol{y} - \boldsymbol{y}_s\|_2^2 \,, \tag{33}$$

In the case of a robot we define $\boldsymbol{y} = \left[\boldsymbol{q}(t_1)^\top \;\cdots\; \boldsymbol{q}(t_N)^\top\right] \in \mathbb{R}^{n \cdot N}$ (resp. $\boldsymbol{y}_s = \left[\boldsymbol{q}_s(t_1)^\top \;\cdots\right.$ $\left.\cdots\; \boldsymbol{q}_s(t_N)^\top\right] \in \mathbb{R}^{n \cdot N}$). Note that Cartesian space formulations of the error function are also possible, see for instance [65]. The minimization of (33) is a nonlinear LS-optimization problem solved by running iterative algorithms such as the gradient or Newton methods which are based on a first- or a second-order Taylor's expansion of the value function $J(\boldsymbol{\beta})$. The unknown parameters are therefore updated iteratively so that the simulated model output fits the measured system output, with

$$\hat{\boldsymbol{\beta}}^{i+1} = \hat{\boldsymbol{\beta}}^{i} + \boldsymbol{\Delta}\hat{\boldsymbol{\beta}}^{i} \,, \tag{34}$$

where $\boldsymbol{\Delta}\hat{\boldsymbol{\beta}}^{i} \in \mathbb{R}^b$ is the innovation vector at iteration i.

Output Error identification can be carried out in Open Loop or in Closed Loop. However, robots being double-integrator systems, the Open-Loop Output Error (OLOE) turns out to be seldom used in practice compared to the Closed Loop Output Error (CLOE) as it is highly sensitive to initial conditions (c.f. [27] for a more detailed discussion on the topic). In CLOE, the simulated data are obtained by integrating the DDM in (3) assuming the same control law for both the actual and simulated robots – without gain adjustments – and using $\hat{\boldsymbol{\beta}}_{i-1}$ the estimate of $\boldsymbol{\beta}$ calculated at iteration $i-1$. The general principle of the CLOE method is illustrated in Fig.4. Let the simulated joint positions $\boldsymbol{q}_s$ be the model outputs. At time t_k the error to be minimized is given by

$$\boldsymbol{e}_{CLOE}(t_k, \boldsymbol{\beta}) = \boldsymbol{q}(t_k) - \boldsymbol{q}_s(t_k, \boldsymbol{\beta})\,, \tag{35}$$

Then, if a classical Gauss-Newton algorithm is chosen, after data sampling and data filtering, the following over-determined system is obtained at iteration i:

$$\boldsymbol{\Delta y}(\boldsymbol{q}) = \boldsymbol{\Phi}^i_{CLOE}\boldsymbol{\Delta}\beta^i_{CLOE} + \varepsilon^i_{CLOE}\,, \tag{36}$$

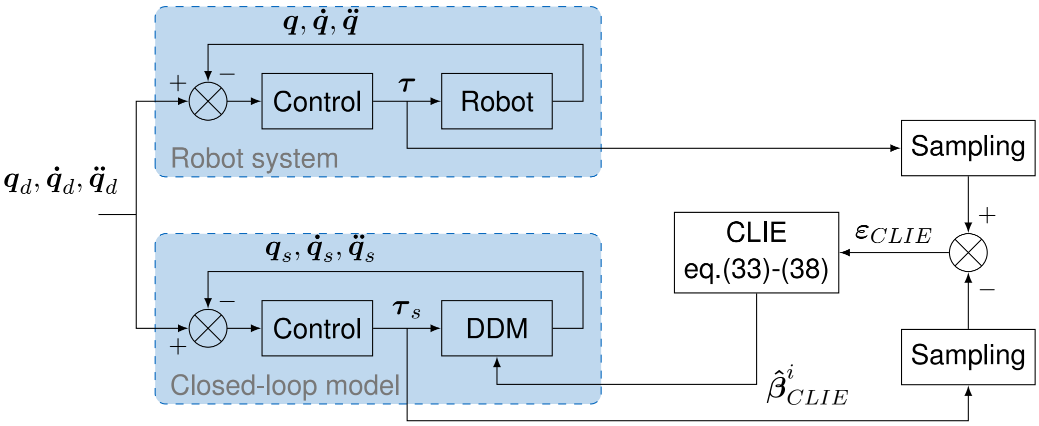
where $\boldsymbol{\Delta y}(\boldsymbol{q}) \in \mathbb{R}^r$ is the vector built from the sampling of $\boldsymbol{e}_{CLOE}(t, \boldsymbol{\beta})$; $\boldsymbol{\Phi}^i_{CLOE} \in \mathbb{R}^{(r\times b)}$ is the matrix built from the sampling of $\boldsymbol{G}_{\boldsymbol{q}_s} = \partial \boldsymbol{q}_s / \partial \beta|_{\beta=\hat{\beta}^i_{CLOE}} \in \mathbb{R}^{(n\times b)}$, the Jacobian matrix of $\boldsymbol{q}_s$ evaluated at $\hat{\beta}^i_{CLOE}$; and the term $\varepsilon^i_{CLOE} \in \mathbb{R}^r$ is the vector built from the sampling of the residuals of the Taylor series expansion. Then, $\boldsymbol{\Delta}\hat{\beta}^i$, the LS estimate of $\boldsymbol{\Delta}\beta^i_{CLOE}$ at iteration i is calculated with (36). Note that numerically computing the Jacobian $\boldsymbol{G}_{\boldsymbol{q}_s}$ with finite differences requires $b+1$ model simulations. Therefore, CLOE is expected to be computationally expensive compared to IDIM-OLS or IDIM-IV. Once CLOE has converged, the covariance matrix of the CLOE estimates is given by

$$\boldsymbol{\Sigma}_{CLOE} = \left(\boldsymbol{\Phi}^\top_{CLOE}\boldsymbol{\Sigma}_q^{-1}\boldsymbol{\Phi}_{CLOE}\right)^{-1}\,, \tag{37}$$

where $\boldsymbol{\Sigma}_q$ is the variance matrix of the joint position measurement noise. Although CLOE is less sensitive to initial conditions, it turns out to be less responsive to changes in the parameters as explained in [66]. As a result, CLOE is expected to converge slowly and potentially to local minima. Note that alternative optimization methods using the derivative-free Nelder-Mead nonlinear simplex method, Genetic Algorithm (GA), Particle Swarm Optimization (PSO) can potentially be applied to tackle this issue. In [66] for example, the authors used the *fminsearch* Matlab function which makes use of the Nelder-Mead simplex algorithm. According to the results presented in [66], although the simplex method appears to be more robust than the classic Levenberg-Marquardt method, it requires an even higher computational effort.

2.2.6. Closed-Loop Input Error (CLIE)

The CLIE method can be seen as a variation of the CLOE method where the simulated torque is being used instead of the simulated position in equation (33), resulting in $\boldsymbol{y} = \left[\boldsymbol{\tau}(t_1)^\top \;\cdots\; \boldsymbol{\tau}(t_N)^\top\right] \in \mathbb{R}^{n\cdot N}$ and $\boldsymbol{y}_s = \left[\boldsymbol{\tau}_s(t_1)^\top \;\cdots\; \boldsymbol{\tau}_s(t_N)^\top\right] \in \mathbb{R}^{n\cdot N}$. The general principle of the CLIE method is illustrated in Fig.5. In this case, the error function which must be



$\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$
+
−
Control
$\boldsymbol{\tau}$
Robot
Robot system
$\boldsymbol{q}_d, \dot{\boldsymbol{q}}_d, \ddot{\boldsymbol{q}}_d$
Sampling
+
CLIE
eq.(33)-(38)
ε_{CLIE}
−
$\boldsymbol{q}_s, \dot{\boldsymbol{q}}_s, \ddot{\boldsymbol{q}}_s$
−
+
Control
$\boldsymbol{\tau}_s$
DDM
Closed-loop model
Sampling
$\hat{\boldsymbol{\beta}}^i_{CLIE}$

Figure 5 Block diagram of the Closed Loop Input Error (CLIE) identification method.

minimized at time t_k has the following expression

$$\boldsymbol{e}_{CLIE}(t_k, \boldsymbol{\beta}) = \boldsymbol{\tau}(t_k) - \boldsymbol{\tau}_s(t_k, \boldsymbol{\beta})\,, \tag{38}$$

and relation (36) thus becomes

$$\boldsymbol{\Delta y}(\boldsymbol{\tau}) = \boldsymbol{\Phi}^i_{CLIE}\boldsymbol{\Delta\beta}^i_{CLIE} + \varepsilon^i_{CLIE}\,, \tag{39}$$

where $\boldsymbol{\Delta y}(\boldsymbol{\tau}) \in \mathbb{R}^r$ is the vector built from the sampling of $\boldsymbol{e}_{CLIE}(t, \boldsymbol{\beta})$; $\boldsymbol{\Phi}^i_{CLIE} \in \mathbb{R}^{(r\times b)}$ is the matrix built from the sampling of $\boldsymbol{G}_{\boldsymbol{\tau}_s} = \partial\boldsymbol{\tau}_s/\partial\boldsymbol{\beta}|_{\boldsymbol{\beta}=\hat{\boldsymbol{\beta}}^i_{CLIE}} \in \mathbb{R}^{(n\times b)}$, the Jacobian matrix of $\boldsymbol{\tau}_s$ evaluated at $\hat{\boldsymbol{\beta}}^i_{CLIE}$ (often referred to as *input sensitivity*); and $\varepsilon^i_{CLIE} \in \mathbb{R}^r$ is the vector built from the sampling of the residuals of the Taylor series expansion. Then, $\boldsymbol{\Delta}\hat{\boldsymbol{\beta}}^i$, the LS estimate of $\boldsymbol{\Delta\beta}^i_{CLIE}$ at iteration i is calculated with (39). Once CLIE has converged, the covariance matrix of the estimate is given by

$$\boldsymbol{\Sigma}_{CLIE} = \left(\boldsymbol{\Phi}^\top_{CLIE}\boldsymbol{\Sigma}^{-1}\boldsymbol{\Phi}_{CLIE}\right)^{-1}. \tag{40}$$

This method was successfully implemented in [28] and formally compared to CLOE, DIDIM and IDIM-OLS. From this work, it appears that although CLIE outperforms both IDIM-OLS and CLOE in terms of accuracy, it has a similar computational complexity as CLOE, as a direct consequence of the finite difference Jacobian matrix computation. The authors demonstrated that CLIE could in fact be thought of as a frequency weighting of the CLOE method by the controller's gains[10]. As a result, although the two estimators are asymptotically equivalent, CLIE proves in practice to be more sensitive to the changes in parameters than CLOE, thereby inducing better convergence properties.

2.2.7. Direct and Inverse Dynamic Identification Model (DIDIM)

In [27], a new algorithm termed Direct and Inverse Dynamic Identification Model (DIDIM) was proposed. The DIDIM method can be seen as a variation of the CLIE algorithm where

[10] This property can be verified in practice by re-injecting (12) into the expression of the sensitivity matrix $\boldsymbol{G}_{\boldsymbol{\tau}_s}$.

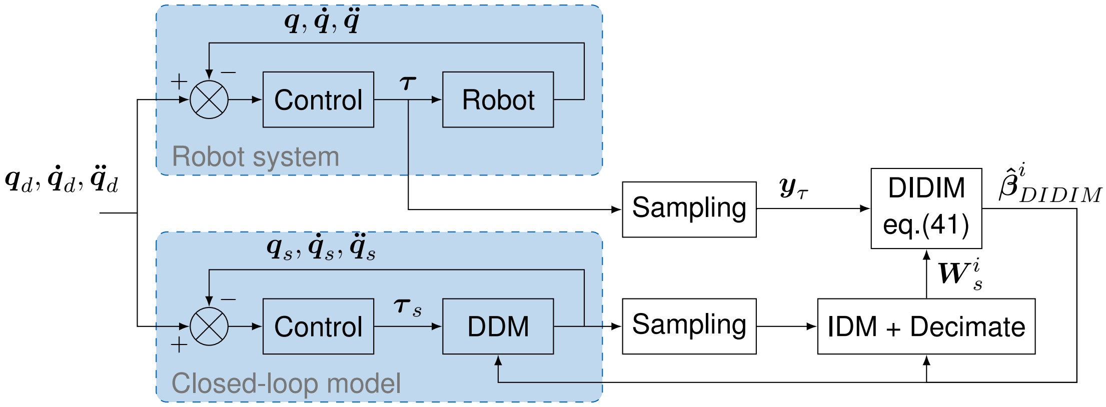


Figure 6 Block diagram of the DIDIM identification method.

some judicious approximations, made in the torque Jacobian matrix computation, yield $G_{\tau_s} = \partial\tau_s/\partial\beta|_{\beta=\hat{\beta}^i_{DIDIM}} \approx Y_\beta(\ddot{q}_s, \dot{q}_s, q_s)$. The initial non-linear LS problem at iteration i hence turns into a much simpler linear LS problem allowing the DIDIM estimates to be calculated as

$$\hat{\beta}^i_{DIDIM} = (W^{i\top}_s \Sigma^{-1} W^i_s)^{-1} W^{i\top}_s \Sigma^{-1} y_\tau\,, \tag{41}$$

where $W^i_s = W(q^i_s, \dot{q}^i_s, \ddot{q}^i_s)$ is the observation matrix constructed with the joint simulated position, joint velocities and joint acceleration obtained from the closed-loop simulation of the DDM with $\hat{\beta}^{i-1}_{DIDIM}$ as explained in section 2.2.3. The general principle of DIDIM is illustrated in Fig.6. Once DIDIM method has converged, the covariance matrix of the DIDIM estimates is given by

$$\Sigma_{DIDIM} = \left(W^\top_s \Sigma^{-1} W_s\right)^{-1}. \tag{42}$$

According to the results gathered in [67, 27, 28], the DIDIM algorithm converges significantly faster than the CLOE and CLIE methods as it only requires a single robot simulation per iteration. In [28], the authors also demonstrate that DIDIM has a similar precision as CLIE. It is interesting to note that the OE methods are generally less sensitive to noise than IDIM-OLS methods since only simulated data is used to build W^i_s. Until now, no formal comparison with IDIM-IV was carried out although [68] provides some elements of discussion, suggesting that the two methods have similar performances. However, further investigations must be conducted in order to refine the conclusions made in [68].

2.2.8. Direct Dynamics Identification Model (DDIM) and Nonlinear Kalman Filtering (NKF)

Widely used for state estimation purpose, Kalman filtering techniques can also be exploited in the context of parameter identification. As explained in [69], identification can be carried out in two different manners, denoted respectively as *dual method* (c.f. [70, 71]) and *joint method* (c.f. [38, 72, 73, 74, 75]). In the first approach, the system state and parameters are identified separately, within two concurrent Kalman filter instances, while in the second one, state and parameters are estimated simultaneously, within a single Kalman filter featuring an

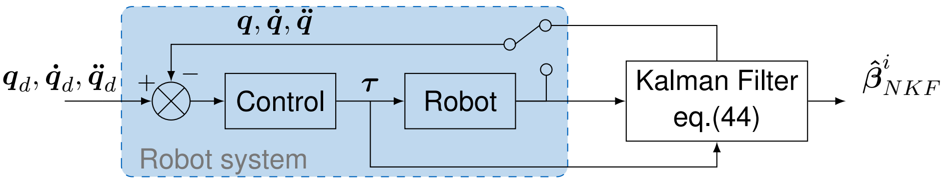

$\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$
$\boldsymbol{q}_d, \dot{\boldsymbol{q}}_d, \ddot{\boldsymbol{q}}_d$
+
−
Control
$\boldsymbol{\tau}$
Robot
Kalman Filter
eq.(44)
$\hat{\boldsymbol{\beta}}^i_{NKF}$
Robot system

Figure 7 Block diagram of the DDIM-NKF identification methods.

augmented state representation. With the notable exception of [71], approaches to robot dynamic parameters identification found in the scientific literature are usually based on the joint filtering paradigm. Since this approach allows accounting for the statistical coupling between the state and the parameters, as suggested by [76], it is therefore expected to be significantly more robust than the dual filtering method. In the joint filtering approach, at time t_k, the state vector $\boldsymbol{x}_k = \left[\dot{\boldsymbol{q}}(t_k)^\top \; \boldsymbol{q}(t_k)^\top\right]^\top \in \mathbb{R}^{2\cdot n}$ and the current estimate of base parameters $\hat{\boldsymbol{\beta}}^{k-1}_{KF}$ are stacked column-wise into a higher-dimensional state vector $\boldsymbol{z}_k = \left[x_k^\top \; \hat{\boldsymbol{\beta}}^{k-1\top}_{KF}\right]^\top \in \mathbb{R}^{2\cdot n+b}$ resulting into the following set of update equations:

$$\boldsymbol{z}_{k+1} = \boldsymbol{\Gamma}(\boldsymbol{z}_k) + \boldsymbol{v}_k \,, \tag{43a}$$

$$\boldsymbol{y}_k = \boldsymbol{S}\boldsymbol{z}_k + \boldsymbol{w}_k \,, \tag{43b}$$

where $\boldsymbol{v}_k = \left[\boldsymbol{n}_k^\top \; \boldsymbol{r}_k^\top\right]^\top \sim \mathcal{N}(\boldsymbol{0}_{(2\cdot n+b)\times 1}, \boldsymbol{\Sigma}_{\boldsymbol{vv}})$ is the process noise; $\boldsymbol{w}_k \sim \mathcal{N}(\boldsymbol{0}_{n\times 1}, \boldsymbol{\Sigma}_{\boldsymbol{ww}})$ is the measurement noise; $\boldsymbol{\Sigma}_{\boldsymbol{vv}} \in \mathbb{R}^{(2\cdot n+b)\times(2\cdot n+b)}$, $\boldsymbol{\Sigma}_{\boldsymbol{ww}} \in \mathbb{R}^{n\times n}$ are respectively the process noise and measurement noise covariance matrices; $\boldsymbol{S} = [\boldsymbol{0}_{n\times n} \; \boldsymbol{1}_{n\times n} \; \boldsymbol{0}_{n\times b}] \in \mathbb{R}^{n\times(2\cdot n+b)}$ is a selection matrix; $\boldsymbol{1}_{n\times n}$ denotes the $(n \times n)$ identity matrix, and $\boldsymbol{0}_{n\times b}$ the $(n \times b)$ zero matrix. Similarly to [38], the nonlinear state transition function $\boldsymbol{\Gamma}(\boldsymbol{z}_k)$ is given by the robot direct dynamic model (DDM) as

$$\mathrm{DDM}(\boldsymbol{z}_k) = \boldsymbol{M}^{-1}(\hat{\boldsymbol{\beta}}^{k-1}_{KF}, \boldsymbol{q}(t_k))(\boldsymbol{\tau}(t_k) - \boldsymbol{h}(\hat{\boldsymbol{\beta}}^{k-1}_{KF}, \boldsymbol{q}(t_k), \dot{\boldsymbol{q}}(t_k))) \tag{44a}$$

$$\boldsymbol{\Gamma}(\boldsymbol{z}_k) = \underbrace{\begin{bmatrix} \dot{\boldsymbol{q}}(t_k) \\ \boldsymbol{q}(t_k) \\ \hat{\boldsymbol{\beta}}^k_{KF} \end{bmatrix}}_{\boldsymbol{z}_k} + \underbrace{\begin{bmatrix} \mathrm{DDM}(\boldsymbol{z}_k) \\ \dot{\boldsymbol{q}}(t_k) \\ \boldsymbol{0}_{b\times 1} \end{bmatrix}}_{\dot{\boldsymbol{z}}_k} \cdot \delta t \,. \tag{44b}$$

In [71], the authors used a different state representation, assuming that the noise level in the joint encoders was negligible, and that the joint motion derivatives could therefore be computed independently. In this context, the state update equation (44) can be simplified into[11] $\boldsymbol{\Gamma}(\boldsymbol{z}_k) = \hat{\boldsymbol{\beta}}^k_{KF}$, with $\boldsymbol{z}_k = \hat{\boldsymbol{\beta}}^{k-1}_{KF}$ while the measurement prediction equation consist of the IDM, computed using the Recursive Newton Euler (RNE) algorithm (c.f. [77]). In practice, the DDM non-linearity in (43)-(44) can be addressed in several different manners. We here

[11] This is not the exact formulation of [71], as the authors use sigmoid barrier functions to enforce physical consistency. This aspect is discussed in greater details in section 2.3.

briefly present some of the most widely used approaches, namely the Extended Kalman Filter (EKF), the Sigma-Point Kalman Filter (SPKF) and the Particle Filter (PF). In the EKF, the prior estimate is propagated through a first-order linearized version of the robot dynamics. Although the resulting computational cost is low, it should be noted that the first order linearization induces a bias in the posterior mean and covariance of the estimate. This bias can be non-negligible when the considered system is highly nonlinear. In SPKF implementations such as the Unscented Kalman Filter (UKF) or the Central Difference Kalman Filter (CDKF), a set of deterministic samples of the prior estimate (assumed to be normally distributed) are propagated through the true nonlinear dynamics of the system. In this manner, the non-linearities can be taken into account up to the second order. Although this method significantly reduces the bias on the estimates of the posterior mean and covariance, it must be noted that it has a higher computational cost than the EKF. As a matter of fact, the computational cost of a SPKF is usually several times greater than that of EKF although it remains on the same order of magnitude [76]. Finally in a Particle Filter (PF) no prior assumption is made on the nature of the estimate distribution. The latter is in fact sampled using a Monte-Carlo method, hence resulting in enhanced robustness, even to severe non-linearities and non-Gaussian noises, but at the expense of computational cost, as the number of particles has a direct influence on the precision of the estimate (c.f. [78, 79]). The full derivation of the EKF and SPKF algorithms is provided in Appendix C. The reader is referred to [76] for a more in-depth discussion and comparison between the different filters.

In the context of parameter identification, it is worth noting that $\boldsymbol{\tau}(t_k)$ in equation (44a) may either denote the torque measured during the experiments – as it is for example in the case of [38, 73] – or the control-torque applied to the simulated closed-loop system during the time-update step of the Kalman filter. In this manner, parameter identification can still be achieved when torque measurements are not available or only available at a low sampling rate, but that the robot control-structure and control-parameters are known. Note however that in the latter case the control loop is updated at a the robot control frequency f_c. As a result, several control iterations can potentially be executed between two updates of the Kalman filter. To the best of our knowledge, this was so far never applied to the field of parameter identification.

2.2.9. Parameter identification using an Adaline Neural Network (AdaNN)

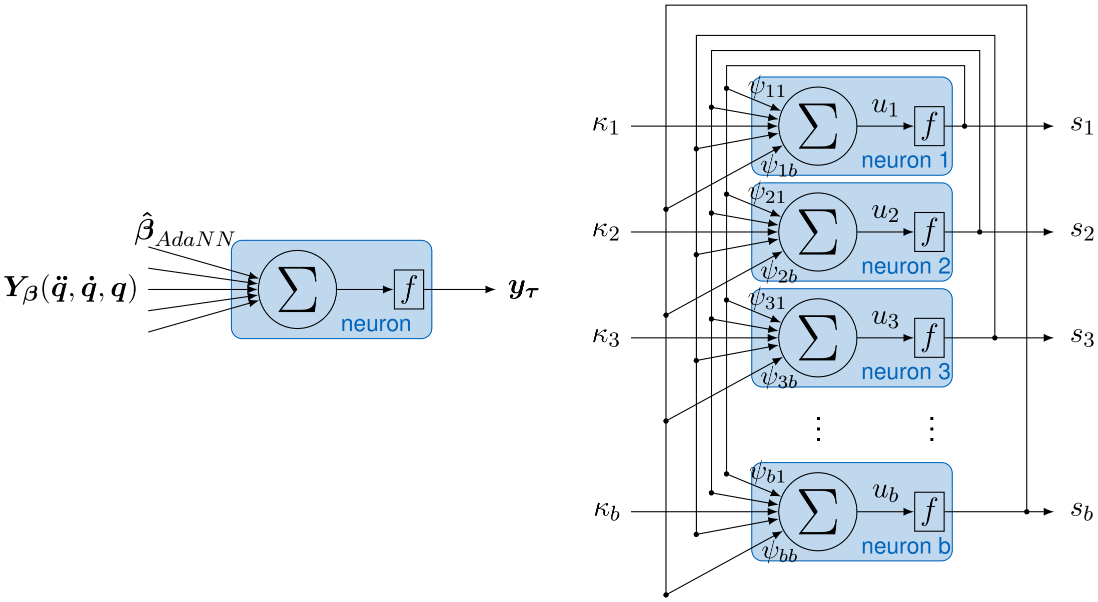
The Adaline (ADAptive LInear NEuron) uses stochastic gradient learning to converge to the IDIM-OLS estimate. The estimator has the following dynamic equation

$$\dot{\hat{\boldsymbol{\theta}}}_{AdaNN} = -\eta \nabla(\boldsymbol{e}^{\top}\boldsymbol{e}) \tag{45}$$

where $\boldsymbol{e}$ is the error defined in equation (14).

The parameter estimate $\hat{\boldsymbol{\beta}}^{k}_{AdaNN}$ at epoch t_k can hence be expressed recursively as

$$\hat{\boldsymbol{\beta}}^{k}_{AdaNN} = \hat{\boldsymbol{\beta}}^{k-1}_{AdaNN} + \eta \boldsymbol{Y}^{k\top} \left(\boldsymbol{\tau}(t_k) - \boldsymbol{Y}^{k}\hat{\boldsymbol{\beta}}^{k-1}_{AdaNN}\right) \tag{46}$$



(i) AdaNN Identification.

(ii) HTRNN Identification.

Figure 8 Adaline and Hopfield Neural Networks used for Inertial Parameter Identification.

where $\boldsymbol{Y}^k = \boldsymbol{Y}_\beta(\ddot{\boldsymbol{q}}(t_k), \dot{\boldsymbol{q}}(t_k), \boldsymbol{q}(t_k))$ is the observation matrix of the real system at epoch t_k, $\boldsymbol{\tau}(t_k)$ is the corresponding torque, and $\eta \in \mathbb{R}$ is referred to as the learning rate. A single "neuron" thus allows full model identification (c.f. Fig.8i). As several *passes* over the whole data-set may be necessary to achieve proper convergence, the sequence of observations $\boldsymbol{Y}^k$ and $\boldsymbol{\tau}(t_k)$ has to be randomly reshuffled in order to avoid cycles [80]. Applications of the Adaline method to robot identification was investigated in [81, 72]. The results suggest that the main advantage of Adaline over IDIM-OLS could be its ability to run online.

2.2.10. Parameter Identification with Hopfield-Tank Recurrent Neural Networks (HTRNN)

Hopfield-Tank recurrent neural networks are dynamic systems made of a set of N interconnected units, or *neurons*. Each neuron i behaves as an integrator coupled with a specific nonlinear *activation function* f as depicted in Fig.8ii. In its original formulation [82], the neuron continuous state equation is expressed as

$$\frac{du_i(t)}{dt} = \sum_{j=1}^{N} \psi_{ij} s_j(t) - \frac{u_i(t)}{R_i C_i} + \kappa_i, \tag{47}$$

where $u_i(t)$ is the internal state of neuron i, $\kappa_i \in \mathbb{R}$ is its input bias, $\psi_{ij} \in \mathbb{R}$ is the connection weight of neuron i with neuron j, $s_j(t) = f(u_j(t)) \in [-1, 1]$, and where $R_i, C_i \in \mathbb{R}_+^\star$ are design parameters corresponding to a resistance and a capacitance respectively. Similarly to [83, 81, 72] we consider in this work the discrete Abe's formulation of Hopfield-Tank neural networks (originally described in [84]) since, as explained in [85, 83], it is particularly suitable for parametric optimization and parameter identification purpose. In Abe's formulation, the activation function is a hyperbolic tangent:

$$s_j = f(u_j) = \alpha_j \tanh\left(\frac{u_j}{\vartheta_j}\right), \ \forall \alpha_j, \vartheta_j \in \mathbb{R}_+^\star. \tag{48}$$

This leads to the following discrete neuron state equation:

$$u_i^{k+1} = u_i^k + \eta \sum_{j=1}^{N} \psi_{ij} \alpha_j \tanh(\frac{u_j^k}{\vartheta_j}) + \kappa_i, \tag{49}$$

where $\eta \in \mathbb{R}_+^\star$ is a tuning parameter often referred to as the learning rate of the network.

It can be demonstrated (c.f. [86, 83]) that such a dynamic system is asymptotically Lyapunov-stable, and that its natural evolution converges toward the minimum of the following energy function:

$$\begin{aligned} E &= -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \psi_{ij} s_i s_j + \sum_{i=1}^{N} \kappa_i s_j, && \text{(50a)} \\ &= -\frac{1}{2} \boldsymbol{s}^\top \boldsymbol{\Psi} \boldsymbol{s} + \boldsymbol{s}^\top \boldsymbol{\kappa}, && \text{(50b)} \end{aligned}$$

where $\boldsymbol{\Psi} = (\Psi_{ij})_{i,j=1\cdots N} \in \mathbb{R}^{N \times N}$, $\boldsymbol{s} = [s_1 \ s_2 \ \cdots \ s_N]^\top \in \mathbb{R}^N$ and $\boldsymbol{\kappa} = [\kappa_1 \ \kappa_2 \ \cdots \ \kappa_N]^\top \in \mathbb{R}^N$. Under these conditions, the process of parameter identification can be thought of as matching the L2-norm of the parameter error ε of equation (15) with the energy function (50) of the Hopfield network. This can be achieved by selecting:

$$\begin{cases} \boldsymbol{\Psi} = -\boldsymbol{W}^\top \boldsymbol{W} \\ \boldsymbol{\kappa} = -\boldsymbol{W}^\top \boldsymbol{y}_\tau \\ \boldsymbol{s} = \hat{\boldsymbol{\beta}}_{HTRNN} \end{cases} \tag{51}$$

Hence the possibility of using Hopfield-Tank networks for optimization purposes and more specifically for parameter identification (see for instance [87, 88, 89, 90, 91, 85, 92, 83, 81, 72] and the references therein). The resulting neural network will have as many neurons as base dynamic parameters to identify, and the corresponding parameter estimate $\hat{\boldsymbol{\beta}}_{HTRNN}$ will converge to the ordinary least-squares (IDIM-OLS) estimate, provided that the appropriate solution range is selected. This range should therefore be carefully adjusted. This is made possible by properly tuning the parameter α_j in (49) to be in the range of the expected parameter values.

2.3. Enforcing Physical Consistency in the Context of Inertial Parameter Identification

This section reviews the main criteria for determining whether a set of identified parameters has physical meaning or not and then discusses possible approaches to enforcing physicality within an identification process.

2.3.1. Mathematical Formulation of the Physicality Conditions

The possibility of enforcing physical consistency as part of a parameter identification process has been the subject of extensive research over the past two decades in robotics. Early contributions, such as [93, 94, 25], formulated the parameter physicality of a given robot link j, as a set of *positivity constraints* on its mass M_j, viscous-Coulomb friction parameters Fv_j, Fc_j and on its transmission-chain inertia Ia_j, as well as a *positivity-definition constraint* over its inertia tensor $\boldsymbol{I}_j$, expressed at the CoM, namely

$$\forall \text{ link } j : \begin{cases} M_j > 0, Fv_j > 0, Fc_j > 0, Ia_j > 0 \\ \boldsymbol{I}_j \succ 0. \end{cases} \tag{52}$$

As highlighted in [25], parameter identification usually relies on frame-dependant linear regression techniques, requiring the identified quantities to be expressed relative to the same reference frame. As a result, the positivity-definition constraint acting on the inertia matrix $\boldsymbol{I}_j$ of a robot link j has to be reformulated in terms of the inertia matrix $\boldsymbol{L}_j$ relative to the link's frame of reference $\mathcal{L}_j$. This is made possibe using the *Huygens-Steiner* theorem[12], yielding

$$\boldsymbol{I}_j = \boldsymbol{L}_j - M_j^{-1} \left[\boldsymbol{mp}_j\right]_\times^\top \left[\boldsymbol{mp}_j\right]_\times , \tag{53}$$

where $\boldsymbol{mp}_j = [MX_j,\ MY_j,\ MZ_j]^\top \in \mathbb{R}^3$ denotes the vector of link's first moments, expressed relative to the link's frame of reference $\mathcal{L}_j$, and $\forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^3\ [\boldsymbol{u}]_\times \boldsymbol{v} = \boldsymbol{u} \times \boldsymbol{v}$. The physicality constraints in (52) can then be reformulated as

$$\forall \text{ link } j : \begin{cases} M_j > 0, Fv_j > 0, Fc_j > 0, Ia_j > 0 \\ \boldsymbol{L}_j - M_j^{-1} \left[\boldsymbol{mp}_j\right]_\times^\top \left[\boldsymbol{mp}_j\right]_\times \succ 0. \end{cases} \tag{54}$$

It is worth noting that prior [25], approaches to physically consistent inertial parameter identification were usually leveraging the equivalence between the positivity-definition constraint on the inertia matrix $\boldsymbol{I}_j$ a set of strict positivity constraints on its eigenvalues $\breve{I}_{xxj}, \breve{I}_{yyj}$ and $\breve{I}_{zzj}$. These eigenvalues are in fact the moments of inertia along the so-called link's *principal axes of inertia* (c.f. Appendix. A.1), yielding

$$\boldsymbol{I}_j = \boldsymbol{R}_j \breve{\boldsymbol{I}}_j \boldsymbol{R}_j^\top \succ 0 \equiv \breve{I}_{xxj} > 0, \breve{I}_{yyj} > 0, \breve{I}_{zzj} > 0, \tag{55}$$

[12] Further details on the *Huygens-Steiner* theorem are made available to the reader in Appendix A.1.

where $\breve{\boldsymbol{I}}_j = \mathrm{diag}(\breve{I}_{xxj}, \breve{I}_{yyj}, \breve{I}_{zzj})$ and $\boldsymbol{R}_j \in SO(3)$ is the rotation matrix relating the frame $\mathcal{I}_j$ of the principal axes of inertia, to the link's Center-of-Mass frame $\mathcal{C}_j$. In this context, [94] proposed a reformulation of IDIM-OLS as a nonlinearly-constrained optimization problem, solved using Sequential Quadratic Programming (SQP), [95] proposed to project the unconstrained parameter vector obtained through Bayesian estimation onto the set of physically consistent parameters, while [71] suggested to directly identify the elements of $\breve{\boldsymbol{I}}_j$ instead of $\boldsymbol{L}_j$, using sigmoid functions within an EKF identification loop to smoothly bound the parameter estimates to physicality[13]. Conversely, [25] proposed reformulating IDIM-LS under physicality constraints in a Semi-Definite Programming (SDP) perspective[14], writing (54) in the form of a Linear Matrix inequality (LMI). As a matter of fact, observing that (53) is in fact the Schur complement of a matrix $\boldsymbol{D}_{Lj}(\boldsymbol{\chi}) \in \mathbb{S}^{6\times 6}$ allows rewriting the physicality constraints (54) as[15]

$$\boldsymbol{D}_j(\boldsymbol{\chi}) = \begin{bmatrix} \boldsymbol{D}_{Lj}(\boldsymbol{\chi}) & \boldsymbol{0}_{6\times 3} \\ \boldsymbol{0}_{3\times 6} & \boldsymbol{D}_{Aj}(\boldsymbol{\chi}) \end{bmatrix} \succ 0, \tag{56}$$

such that

$$\boldsymbol{D}_{Lj}(\boldsymbol{\chi}) = \begin{bmatrix} \boldsymbol{L}_j & [\boldsymbol{mp}_j]_\times^\top \\ [\boldsymbol{mp}_j]_\times & M_j \boldsymbol{1}_{3\times 3} \end{bmatrix}, \boldsymbol{D}_{Aj}(\boldsymbol{\chi}) = \begin{bmatrix} Fv_j & 0 & 0 \\ 0 & Fc_j & 0 \\ 0 & 0 & Ia_j \end{bmatrix} \tag{57}$$

where $\boldsymbol{0}_{3\times 3}, \boldsymbol{1}_{3\times 3}$ respectively denote the 3×3 zero and identity matrices. In [97, 98, 99], the authors went one step further by noticing that (54) and (56) could still potentially lead to inconsistent link mass distributions. They consequently defined a *density-realizability* criterion in the form of an additional triangle-inequality condition to be fulfilled alongside with (54), namely

$$\begin{cases} \breve{I}_{xxj} + \breve{I}_{yyj} > \breve{I}_{zzj} \\ \breve{I}_{yyj} + \breve{I}_{zzj} > \breve{I}_{xxj} \\ \breve{I}_{zzj} + \breve{I}_{xxj} > \breve{I}_{yyj} \end{cases} \equiv \mathrm{tr}(\breve{\boldsymbol{I}}_j) > 0 \equiv \mathrm{tr}(\boldsymbol{I}_j) > 0 \tag{58}$$

where $\mathrm{tr}\,(\cdot)$ denotes the trace operator. A reformulation $\boldsymbol{D}'_{Lj}(\boldsymbol{\chi}) \in \mathbb{S}^{4\times 4}$ of the term $\boldsymbol{D}_{Lj}(\boldsymbol{\chi})$ within the constraint matrix $\boldsymbol{D}_j(\boldsymbol{\chi})$ was eventually proposed to account for density realizability constraints alongside with (54)

$$\boldsymbol{D}'_{Lj}(\boldsymbol{\chi}) = \begin{bmatrix} \underbrace{\frac{1}{2}\mathrm{tr}\,(\boldsymbol{L}_j)\boldsymbol{1}_{3\times 3} - \boldsymbol{L}_j}_{\tilde{\boldsymbol{L}}_j} & \boldsymbol{mp}_j \\ \boldsymbol{mp}_j^\top & M_j \end{bmatrix} \succ 0. \tag{59}$$

It is here worth noting that $\boldsymbol{L}_j$ can be reconstructed from $\tilde{\boldsymbol{L}}_j$ (and hence $\boldsymbol{D}'_{Lj}(\boldsymbol{\chi})$) as

$$\boldsymbol{L}_j = \mathrm{tr}\left(\tilde{\boldsymbol{L}}_j\right)\boldsymbol{1}_{3\times 3} - \tilde{\boldsymbol{L}}_j \tag{60}$$

[13] It is worth pointing out that such an approach could also possibly be applied to the HTRNN process by modifying the hyperbolic tangent activation function, although, to the best of our knowledge, this was never done.

[14] The reader is invited to read [96] for a comprehensive introduction on SDP.

[15] Further details on the Schur complement are made available to the reader in Appendix B.2.

2.3.2. Physically Consistent Identification of the Base Inertial Parameters

As discussed in section 2.1.2 of this thesis, inertial parameter identification of an articulated mechanism inherently suffers from structural rank deficiencies, eventually resulting in an infinite number of solutions, some of which can be physically inconsistent. Identifying a vector $\boldsymbol{\beta}$ of base inertial parameters, obtained from a set of appropriate linear combinations of the vector $\boldsymbol{\chi}$ of standard inertial parameters, allows to overcome the issue of structural rank deficiencies but does not by any mean guarantee physical consistency. To do so, it is necessary to adjust the mathematical formulation of the parameter physicality constraints. This is made possible by means of a bijective transformation $\mathfrak{m}$ proposed by Sousa et al. [25], defined in (10)-(11), and mapping $(\boldsymbol{\beta}, \underline{\boldsymbol{\chi}})$ to $\boldsymbol{\chi}$, where the vector $\underline{\boldsymbol{\chi}}$ denotes the set of unidentifiable standard parameters, eventually yielding

$$\boldsymbol{D}_j^{\boldsymbol{\beta}}(\boldsymbol{\beta}, \underline{\boldsymbol{\chi}}) \succ 0 \equiv \boldsymbol{D}_j(\mathfrak{m}^{-1}(\boldsymbol{\beta}, \underline{\boldsymbol{\chi}})) \succ 0 \tag{61}$$

where $\boldsymbol{D}_j^{\boldsymbol{\beta}}(\boldsymbol{\beta}, \underline{\boldsymbol{\chi}})$ denotes the LMI formulation of the base parameter physicality constraints. Sousa et al. moreover proposed a set of two alternative formulations supporting physically consistent identification of the base inertial parameters. The first, formulation consists in finding the base parameter vector $\hat{\boldsymbol{\beta}}_{PC-WLS}$ which is as close as possible, in the least square sense, to the IDIM-LS solution while at the same time complying with the physicality constraints defined in (59)

$$\begin{aligned}\hat{\boldsymbol{\beta}}_{PC-WLS} =& \underset{\boldsymbol{\beta}, \underline{\boldsymbol{\chi}}}{\arg\min} \left\| \hat{\boldsymbol{\beta}}_{WLS} - \boldsymbol{\beta} \right\|_2^2 \\ & s.t.\ \boldsymbol{D}_j^{\boldsymbol{\beta}}(\boldsymbol{\beta}, \underline{\boldsymbol{\chi}}) \succ 0\ \forall j = \{1 \cdots n\}\end{aligned} \tag{62}$$

Although valid, this formulation has the notable drawback of converging to sub-optimal solutions, due to the fact that the IDIM-LS and the physicality identification problems in (16) and (62) are decoupled. As an alternative, Sousa et al. proposed to fuse (16) and (62) in the form of a single SDP which is equivalent[16] to

$$\begin{aligned}\hat{\boldsymbol{\beta}}_{PC-WLS} =& \underset{\boldsymbol{\beta}, \underline{\boldsymbol{\chi}}}{\arg\min} \left(\boldsymbol{W}\boldsymbol{\beta} - \boldsymbol{y}_\tau\right)^\top \boldsymbol{\Sigma}^{-1} \left(\boldsymbol{W}\boldsymbol{\beta} - \boldsymbol{y}_\tau\right) \\ & s.t.\ \boldsymbol{D}_j^{\boldsymbol{\beta}}(\boldsymbol{\beta}, \underline{\boldsymbol{\chi}}) \succ 0\ \forall j = \{1 \cdots n\}\end{aligned} \tag{63}$$

A more general SDP reformulation of IDIM-OLS and -WLS, that will be referred to as Physically-Consistent (PC-) in this thesis, makes the bijective mapping between base and standard

[16] The exact formulation of Sousa et al., in [25] is actually given in a cannonical SDP form that complies with most dedicated solvers. The problem formulation adopted in this thesis is more "human friendly" but can only be implemented "as is" in a limited set of solvers.

parameters visible, in the form of an invertible matrix $\boldsymbol{\Omega}$, defined in (11)

$$\begin{aligned}\hat{\boldsymbol{\beta}}_{WLS} =& \underset{\boldsymbol{\beta},\underline{\boldsymbol{\chi}}}{\arg\min}\left(\boldsymbol{W}\boldsymbol{\beta}-\boldsymbol{y}_{\tau}\right)^{\top}\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{W}\boldsymbol{\beta}-\boldsymbol{y}_{\tau}\right)\\ & s.t.\ \boldsymbol{\chi}=\boldsymbol{\Omega}^{-1}\left[\boldsymbol{\beta}^{\top}\ \underline{\boldsymbol{\chi}}^{\top}\right]^{\top}\\ & \boldsymbol{D}_j(\boldsymbol{\chi})\succ 0\ \forall j=\{1\cdots n\}\end{aligned} \tag{64}$$

This formulation will be useful in what follows, in particular to support the discussion on marginal physicality. It is worth mentioning that the very structure of IDIM-IRLS (c.f. [21]), IDIM-IV and DIDIM (c.f. [100]) also makes it possible to seamlessly integrate the LMI physicality constraints by actually solving a SDP at each iteration of the corresponding algorithms. For instance, the PC-DIDIM algorithm (proposed in [100]) can be implemented by solving at each iteration i

$$\begin{aligned}\hat{\boldsymbol{\beta}}^{i}_{DIDIM} =& \underset{\boldsymbol{\beta}^i,\underline{\boldsymbol{\chi}}}{\arg\min}\left(\boldsymbol{W}^i_s\boldsymbol{\beta}^i-\boldsymbol{y}_{\tau}\right)^{\top}\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{W}^i_s\boldsymbol{\beta}^i-\boldsymbol{y}_{\tau}\right)\\ & s.t.\ \boldsymbol{\chi}=\boldsymbol{\Omega}^{-1}\left[\boldsymbol{\beta}^{\top}\ \underline{\boldsymbol{\chi}}^{\top}\right]^{\top}\\ & \boldsymbol{D}_j(\boldsymbol{\chi})\succ 0\ \forall j=\{1\cdots n\}\end{aligned} \tag{65}$$

instead of (41). Similarly, the PC-IDIM-IV algorithm can be implemented by solving the following SDP at each iteration i

$$\begin{aligned}\hat{\boldsymbol{\beta}}^{i}_{IV} =& \underset{\boldsymbol{\beta}^i,\underline{\boldsymbol{\chi}}}{\arg\min}\left(\boldsymbol{A}^i\boldsymbol{\beta}^i-\boldsymbol{b}^i\right)^{\top}\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{A}^i\boldsymbol{\beta}^i-\boldsymbol{b}^i\right)\\ & s.t.\ \boldsymbol{\chi}=\boldsymbol{\Omega}^{-1}\left[\boldsymbol{\beta}^{\top}\ \underline{\boldsymbol{\chi}}^{\top}\right]^{\top}\\ & \boldsymbol{D}_j(\boldsymbol{\chi})\succ 0\ \forall j=\{1\cdots n\}\end{aligned} \tag{66}$$

where

$$\begin{cases}\boldsymbol{A}^i=\boldsymbol{W}^i_s(\boldsymbol{W}^{i\top}_s\boldsymbol{W}^i_s)^{-1}\boldsymbol{W}^{i\top}_s\boldsymbol{W}\\ \boldsymbol{b}^i=\boldsymbol{W}^i_s(\boldsymbol{W}^{i\top}_s\boldsymbol{W}^i_s)^{-1}\boldsymbol{W}^{i\top}_s\boldsymbol{y}_{\boldsymbol{\tau}}\end{cases} \tag{67}$$

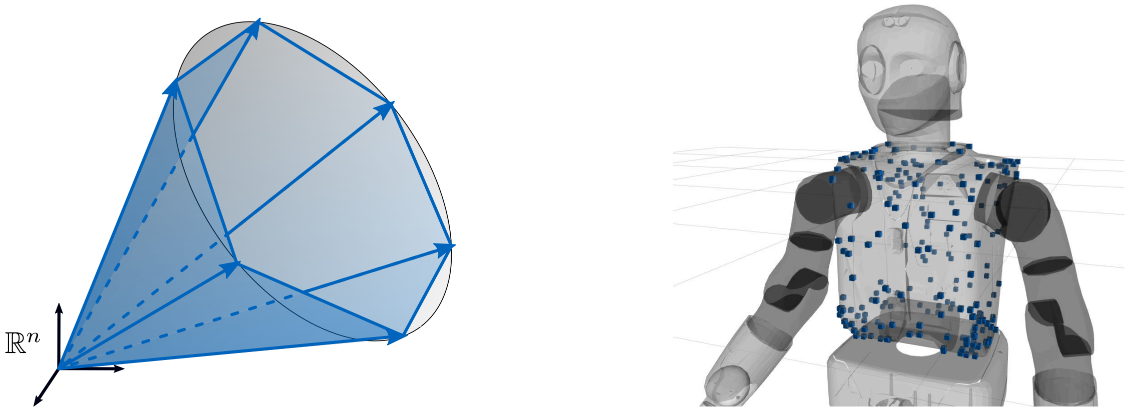
It is worth noting that [101, 102] proposed a linearization of both the positivity-definition and density realizability constraints applied to $\boldsymbol{I}_j$, in the form of a mass-positivity constraint applied to a homogeneous distribution of $N_{\rho,j}$ independent point-masses located within each robot link j (c.f. Fig.9ii), yielding

$$\boldsymbol{\chi}_j=\boldsymbol{Q}_j\boldsymbol{\rho}_j \tag{68}$$

where $\boldsymbol{\rho}_j=\left[\rho_{j,1},\cdots,\rho_{j,N_{\rho,j}}\right]^{\top}\in\mathbb{R}^{N_{\rho,j}}$ concatenates the masses of each point $\rho_{j,k},\ k\in[1,\cdots,N_{\rho,j}]$ on link j and $\boldsymbol{Q}_j=\left[\boldsymbol{\chi}_{j,1},\cdots,\boldsymbol{\chi}_{j,N_{\rho,j}}\right]^{\top}\in\mathbb{R}^{10\times N_{\rho,j}}$ denotes the map between the point-mass distribution and the link standard inertial parameter vector $\boldsymbol{\chi}_j$. Extended to the whole robot, one eventually gets

$$\boldsymbol{\chi}=\boldsymbol{Q}\boldsymbol{\rho} \tag{69}$$

where $\boldsymbol{Q}=\begin{bmatrix}\boldsymbol{Q}_1 & \boldsymbol{0} & \boldsymbol{0}\\ \boldsymbol{0} & \ddots & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{Q}_n\end{bmatrix}\in\mathbb{R}^{10k\times k_{\rho}}$ and $\boldsymbol{\rho}=[\boldsymbol{\rho}_1,\cdots,\boldsymbol{\rho}_n]^{\top}\in\mathbb{R}^{N_{\rho}}$. The physicality constraints of (59) can hence be approximated by a set of positivity constraints on the point



(i) Polyhedral approximation of the physical consistency cone.

(ii) Detail of a point mass distribution on the torso of a REEM-C robot.

Figure 9 Detail of a point mass distribution on the torso link of our REEM-C robot.

masses of the vector $\boldsymbol{\rho}$, namely

$$\boldsymbol{\rho} > \mathbf{0} \tag{70}$$

Considering that the LMI (59) can be represented as a membership condition to an opened convex cone – since $\forall t > 0, \boldsymbol{A} \in \mathbb{S}^{k\times k}_{++},\ t\boldsymbol{A} \in \mathbb{S}^{k\times k}_{++}$, where $\mathbb{S}^{k\times k}_{++}$ denotes the set of symmetric positive definite matrices of dimension k – the linear constraint (70) can be thought of as a polytopic approximation of (59) (c.f. Fig.9i). Provided that the number of point-masses is "high enough", PC-IDIM-OLS and -WLS can thus be reformulated as Quadratic Programs (QP) with a good precision, eventually yielding

$$\begin{aligned} \boldsymbol{\rho}^{\star} = & \arg\min_{\boldsymbol{\rho}} \left(\boldsymbol{W}\boldsymbol{Q}\boldsymbol{\rho} - \boldsymbol{y}_{\tau}\right)^{\top} \boldsymbol{\Sigma}^{-1} \left(\boldsymbol{W}\boldsymbol{Q}\boldsymbol{\rho} - \boldsymbol{y}_{\tau}\right) \\ s.t. \qquad & \boldsymbol{\rho} > \mathbf{0} \end{aligned} \tag{71}$$

The standard parameter vector can finally be computed as $\boldsymbol{\chi} = \boldsymbol{Q}\boldsymbol{\rho}^{\star}$. It should be emphasized that although small to medium size QP are generally extremely fast to resolve, dimentionality can here rapidly become problematic, as in particular shown in [103], since a good approximation of a link mass distribution typically requires several hundreds of samples.

2.3.3. Preventing Marginal Physicality and Data Rank Deficiency

As highlighted in [104, 103], imposing parameter physicality using a set of rigid constraints within an optimization process may eventually lead to situations where the parameter estimates lie at the very border of the physical consistency spectrahedron. This phenomenon, referred to as *marginal physicality*, typically occurs when the unconstrained estimate lies outside the physicality region[17], as it might for instance, be the case when the observation matrix $\boldsymbol{W}$ is ill-conditioned, due to a lack of excitation or to excessive noise in $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$. In [94, 105] for instance, the authors proposed adding a relaxation term within the IDIM-OLS cost function, minimizing the Euclidean distance between the current parameter estimate $\boldsymbol{\chi}$ and the set of

[17] This can be seen as a consequence of the the convexity of the problem.

initial standard parameters χ_0 of the robot. In this case, (64) would be reformulated as

$$
\begin{aligned}
\hat{\boldsymbol{\beta}}_{OLS} =& \underset{\boldsymbol{\beta},\underline{\boldsymbol{\chi}}}{\arg\min}\, \|\boldsymbol{W}\boldsymbol{\beta} - \boldsymbol{y}_{\tau}\|_2^2 + \underbrace{\lambda\,\|\boldsymbol{\chi} - \boldsymbol{\chi}_0\|_2^2}_{\text{Euclidean relaxation}} \\
& s.t.\ \boldsymbol{\chi} = \boldsymbol{\Omega}^{-1}\left[\boldsymbol{\beta}^{\top}\ \underline{\boldsymbol{\chi}}^{\top}\right]^{\top} \\
& \boldsymbol{D}_j(\boldsymbol{\chi}) \succ 0\ \forall j = \{1 \cdots n\},
\end{aligned} \tag{72}
$$

where $\lambda << 1$ is a tuning quantity. Note that a similar concept was proposed in [106] using hierarchical optimization. In practice, this term allows "shifting" the current parameter estimates toward the CAD original estimates which, by essence, are physically consistent. Observing that the set of physically consistent parameters, verifying (59) could be given a Riemannian manifold structure, associated with a suitable metric, [104, 103] proposed to replace the Euclidean distance used in the relaxation process by a non-Euclidean distance, thereby shifting the problem along the physicality manifold, yielding

$$
\begin{aligned}
\hat{\boldsymbol{\beta}}_{OLS} =& \underset{\boldsymbol{\beta},\underline{\boldsymbol{\chi}}}{\arg\min}\, \|\boldsymbol{W}\boldsymbol{\beta} - \boldsymbol{y}_{\tau}\|_2^2 + \underbrace{\lambda \boldsymbol{d}\left(\boldsymbol{\chi} - \boldsymbol{\chi}_0\right)^2}_{\text{Geometric relaxation}} \\
& s.t.\ \boldsymbol{\chi} = \boldsymbol{\Omega}^{-1}\left[\boldsymbol{\beta}^{\top}\ \underline{\boldsymbol{\chi}}^{\top}\right]^{\top} \\
& \boldsymbol{D}_j(\boldsymbol{\chi}) \succ 0\ \forall j = \{1 \cdots n\},
\end{aligned} \tag{73}
$$

where the term $\boldsymbol{d}\left(\cdot\right)$ refers to a distance function on the Riemannian manifold made by the set of physically consistent parameters.

2.4. Parameter identification of Floating-Base Mechanisms

This section addresses the particularities related to the inertial parameter identification of floating base mechanisms. The specific conditions of their excitation are also presented.

2.4.1. Specificities of Floating-Base Mechanisms in the Context of Inertial Parameter Identification

Over the past decade, a set of notable contributions have investigated the issue of inertial parameter identification in the context of floating-based systems, both for robotics [107, 108, 109] and medical [110, 111, 112] application. In essence, these works are inspired by the pioneering research of Dubowsky et al. [113] on the parameter identification for space robotic systems. The inverse dynamic model of a floating base robot can be derived from the dynamic model of a fixed base mechanism (1) by taking into account the Cartesian dynamics of the floating base formulated as a set of six additional DoF, yielding

$$
\begin{bmatrix} \boldsymbol{M}_b(\boldsymbol{\chi},\boldsymbol{x}) & \boldsymbol{M}_{bq}(\boldsymbol{\chi},\boldsymbol{x},\boldsymbol{q}) \\ \boldsymbol{M}_{qb}(\boldsymbol{\chi},\boldsymbol{x},\boldsymbol{q}) & \boldsymbol{M}_q(\boldsymbol{\chi},\boldsymbol{q}) \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{x}} \\ \ddot{\boldsymbol{q}} \end{bmatrix} + \begin{bmatrix} \boldsymbol{h}_b(\boldsymbol{\chi},\boldsymbol{x},\dot{\boldsymbol{x}}) \\ \boldsymbol{h}_q(\boldsymbol{\chi},\boldsymbol{q},\dot{\boldsymbol{q}}) \end{bmatrix} = \begin{bmatrix} \boldsymbol{0}_{6\times 1} \\ \boldsymbol{\tau} \end{bmatrix} + \sum_{i=1}^{c} \begin{bmatrix} \boldsymbol{J}_{b_i}^{\top} \\ \boldsymbol{J}_{q_i}^{\top} \end{bmatrix} \boldsymbol{w}_i\,. \tag{74}
$$

In this equation, the upper six rows describe the unactuated floating base dynamics in Cartesian space while the n lower rows describe the actuated joint-space dynamics. Accordingly $\boldsymbol{x} \in \mathbb{R}^6$ corresponds to the vector of general coordinates representing the position and orientation of the floating base in Cartesian space, $\boldsymbol{M}_b(\boldsymbol{\chi}, \boldsymbol{x}) \in \mathbb{S}^{6\times 6}$, $\boldsymbol{M}_q(\boldsymbol{\chi}, \boldsymbol{q}) \in \mathbb{S}^{n\times n}$ and $\boldsymbol{M}_{bq}(\boldsymbol{\chi}, \boldsymbol{x}, \boldsymbol{q}) = \boldsymbol{M}_{qb}^\top(\boldsymbol{\chi}, \boldsymbol{x}, \boldsymbol{q}) \in \mathbb{R}^{6\times n}$ are the inertia matrices describing the floating base dynamics, robot body dynamics and the coupling between the floating base and the robot body, respectively. Similarly, $\boldsymbol{h}_b(\boldsymbol{\chi}, \boldsymbol{x}, \dot{\boldsymbol{x}})$ and $\boldsymbol{h}_q(\boldsymbol{\chi}, \boldsymbol{q}, \dot{\boldsymbol{q}})$ denote the bias force vector including the Coriolis, Centripetal, gravity and – possible – friction terms of the floating base and of the robot mechanism. $c \in \mathbb{N}$ is the number of contact points between the robot and its environment, $\boldsymbol{\tau} \in \mathbb{R}^n$ is the vector of control torques, applied by the different robot joints, $\boldsymbol{w}_i$ denotes the i^{th} contact wrench between the robot and its environment and $\boldsymbol{J}_{b_i} \in \mathbb{R}^{6\times 6}$ $\boldsymbol{J}_{q_i} \in \mathbb{R}^{6\times n}$ are the Jacobian matrices, whose transposed allows to map the contact wrenches to the floating base and to the different robot links, respectively. The key idea of floating-base system identification is that, similar to the case of fixed-base robotic systems, the IDM (74) can be expressed linearly with respect to the corresponding standard inertial parameter vector $\boldsymbol{\chi}$:

$$\begin{bmatrix} \boldsymbol{0}_{6\times 1} \\ \\ \boldsymbol{\tau} \end{bmatrix} = \underbrace{\begin{bmatrix} \boldsymbol{Y}_{b\boldsymbol{\chi}}(\ddot{\boldsymbol{x}}, \dot{\boldsymbol{x}}, \boldsymbol{x}, \ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q}) \\ \\ \boldsymbol{Y}_{q\boldsymbol{\chi}}(\ddot{\boldsymbol{x}}, \dot{\boldsymbol{x}}, \boldsymbol{x}, \ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q}) \end{bmatrix}}_{\boldsymbol{Y}_{\boldsymbol{\chi}}} \boldsymbol{\chi} + \sum_{i=1}^{c} \begin{bmatrix} \boldsymbol{J}_{b_i}^\top \\ \\ \boldsymbol{J}_{q_i}^\top \end{bmatrix} \boldsymbol{w}_i \,. \tag{75}$$

As previously stated, it should be highlighted that both parts of the regression matrix tend to be subject to structural rank deficiencies. Nevertheless the methods used for base parameter or essential parameter computation remain valid, eventually leading to the following identification model:

$$\begin{bmatrix} \boldsymbol{0}_{6\times 1} \\ \\ \boldsymbol{\tau} \end{bmatrix} = \underbrace{\begin{bmatrix} \boldsymbol{Y}_{b\boldsymbol{\beta}}(\ddot{\boldsymbol{x}}, \dot{\boldsymbol{x}}, \boldsymbol{x}, \ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q}) \\ \\ \boldsymbol{Y}_{q\boldsymbol{\beta}}(\ddot{\boldsymbol{x}}, \dot{\boldsymbol{x}}, \boldsymbol{x}, \ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q}) \end{bmatrix}}_{\boldsymbol{Y}_{\boldsymbol{\beta}}} \boldsymbol{\beta} + \sum_{i=1}^{c} \begin{bmatrix} \boldsymbol{J}_{b_i}^\top \\ \\ \boldsymbol{J}_{q_i}^\top \end{bmatrix} \boldsymbol{w}_i \,. \tag{76}$$

It follows that a direct identification of the robot parameters is possible using one of the algorithms presented in 2.2 along with the combined measurements of joint motion derivatives, contact forces [114] and joint torques [107, 108, 115, 116]. It should be emphasized that such a direct approach turns out to be rather cumbersome in practice, as floating base mechanisms are by design more likely to be subject to joint backlash or complex nonlinear friction phenomena, notwithstanding that joint torque measurements – often obtained through noisy current observers – are generally unreliable, if not simply unavailable, especially in the context of human body link identification. However these obstacles can be overcome by noticing in (76) that the first six terms referring to the floating base unactuated dynamics *have no dependence in the joint torque vector* $\boldsymbol{\tau}$. As highlighted by Venture et al. [110, 117, 109], it is therefore possible to identify the base parameter vector $\boldsymbol{\beta}$ using only the measurements of the contact wrenches $\boldsymbol{w}_i, i = \{1 \cdots c\}$ between the robot and its environment, combined with the joints and floating base motion derivatives

$$\mathbf{0}_{6\times 1} = \boldsymbol{Y_{b\beta}}(\ddot{\boldsymbol{x}}, \dot{\boldsymbol{x}}, \boldsymbol{x}, \ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q})\boldsymbol{\beta} + \sum_{i=1}^{c} \boldsymbol{J}_{b_i}^{\top} \boldsymbol{w}_i \,. \tag{77}$$

This method is particularly appealing as it eliminates the need for accurate joint transmission or friction models and also enables the identification of human body segments or floating base system hanged to a fixed force torque sensor [118]. It is worth emphasizing that this method has been shown mathematically to maintain structural identifiability [109] to the extent that $\dim(\text{col}(\boldsymbol{Y_{b\beta}})) = \dim(\text{col}(\boldsymbol{Y_\beta}))$. A number of studies have undertaken comparative analyses between this approach and the classic method [107, 109, 116]. Their results suggest that identification from the floating base regressor provides better predictions in terms of contact wrenches between a robot and its environment, while taking into account the upper part of the regressor seems to provide better predictions in terms of joint torques (sometimes at the expense of contact wrench prediction). It is here important to clarify that the use of the upper part of the regressor can only be done if appropriate joint torque data are available, for example, from a dedicated set of joint torque sensors [107, 116]. More generally, this implies that the way a robot is identified will ultimately depend on the goals of that identification process.

2.4.2. Excitability of Floating-Base Mechanisms under Constraints

Unlike static industrial robot manipulators, performing excitation motions on hyper-redundant floating base mechanisms for parameter identification purposes proves to be a rather difficult task [119]. This mainly stems from the significantly larger problem dimension as well as the multiple kinematic and dynamic constraints that must be accounted for throughout the process, such as robot stability, contact transition, friction cones or self-collision avoidance. It is also worth noting that in the context of human body identification, the inevitable issues of tracking precision and repeatability should also be accounted for. As a result, most of the standard exciting motion generation techniques used for the identification of fixed-base robots, such as [13, 120, 16, 121, 122], do not allow the generation of appropriate exciting behaviors in the case of more complex floating-base systems. The reference method in the context of human body segment identification was proposed by Venture at al. [123]. It consists in capturing excitation data from a set of motion sequences executed by a human subject – or possibly a humanoid robot – during a dedicated experiment [123, 124], and to build the problem regressor by only selecting the most relevant data samples. The full system's regressor matrix is divided into several column blocks, referred to as *sub-regressors*, and accounting for the different parts of the considered mechanism. In practice, each of these blocks can be excited separately, thereby considerably simplifying the data collection problem. Extending this paradigm to the identification of floating base robots requires accounting for the robot's stability and physical constraints. Relying on the sub-regressor approach proposed by

[123], Jovic et al. [20] were able to identify the dynamic parameters of a full humanoid robot using a combination of different walking motions along with a set of tailor-made upper-body motions, executed within a constrained robot control framework and filtered by a dedicated stabilizer in order to maintain balance during the experiments. It should be highlighted that such a method do not guaranty that the executed trajectories will excite the model in an optimal manner, thereby potentially raising rank deficiency issues.

To provide excitation guarantees and, at the same time, reduce the required amount of experiment data, several approaches, such as [125, 126, 127], investigated more systematic ways of generating Optimal Exciting Motions (OEM) both for robots and humans. In practice, formulating the trajectory generation process for floating base systems as an optimization problem proves to be a challenging task due to the high system dimentionality as well as the large number of constraints to be accounted for, thereby often yielding infeasibility or convergence to local minima. In Bonnet et al. [126, 127], these issues were successfully tackled by decoupling the trajectory and posture generation process. The authors made a distinction between the so called robot's *static* inertial parameters (i.e. link masses and CoM) whose identification only requires data collection from a set of static poses, and the *dynamic* inertial parameters (i.e. the terms of the links inertia matrices) which can only be identified from robot motions executed at non-zero velocities and accelerations. The resulting constrained optimization method generates a set of stable postures allowing to identify the static robot parameters as well as a set of posture transition motion sequences enabling identification of the dynamic inertial parameters. It is worth noting that this process is carried out while explicitly accounting for robot balance – by regularizing the ZMP excursion – and physical constraints. The resulting motion sequences can then be executed by the robot to perform identification. It should be highlighted that this process was validated both for humanoid robots and for real human subjects.

2.5. Hierarchical Control of Floating-Base Mechanisms under Constraints

As the practical implementation of inertial parameter identification techniques requires the prior development of a robust control environment allowing the simultaneous execution of multiple tasks subject to a set of specific constraints, we here elaborate on some aspects of the state of the art in this field.

2.5.1. Fundamentals of Redundancy Resolution and Hierarchical Task Execution in Robotics

2.5.1.1 Redundancy Resolution

Differential kinematics allows defining a mapping between the robot's n-dimensional joint motion derivatives $\dot{\boldsymbol{q}} = [\dot{q}_1, \dot{q}_2, \cdots, \dot{q}_n]^T, \ddot{\boldsymbol{q}} = [\ddot{q}_1, \ddot{q}_2, \cdots, \ddot{q}_n]^T \in \mathbb{R}^n$ and the corresponding quantities in the m-dimensional Cartesian space, denoted as $\dot{\boldsymbol{x}}, \ddot{\boldsymbol{x}} \in \mathbb{R}^m$:

$$\dot{\boldsymbol{x}} = \boldsymbol{J}\dot{\boldsymbol{q}} \tag{78a}$$

$$\ddot{\boldsymbol{x}} = \boldsymbol{J}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}\dot{\boldsymbol{q}} \tag{78b}$$

where $\boldsymbol{J}(\boldsymbol{q}) = \partial\dot{\boldsymbol{x}}/\partial\dot{\boldsymbol{q}}$ is referred to as the task Jacobian matrix [128]. Conversely, inverse kinematics allows mapping the Cartesian space variables to Joint space variables. In case the Jacobian is invertible, one gets:

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{-1}\dot{\boldsymbol{x}} \tag{79a}$$

$$\ddot{\boldsymbol{q}} = \boldsymbol{J}^{-1}\left(\ddot{\boldsymbol{x}} - \dot{\boldsymbol{J}}\dot{\boldsymbol{q}}\right) \tag{79b}$$

Although resolving the inverse kinematics appears problematic at first glance[18], suitable solutions can be found using generalized inverses of the Jacobian matrix [129]. In its general form, the generalized inverse $\boldsymbol{J}^{\#\boldsymbol{\Xi}}$ of $\boldsymbol{J}$ can be expressed as

$$\boldsymbol{J}^{\#\boldsymbol{\Xi}} = \boldsymbol{\Xi}^{-1}\boldsymbol{J}^\top\left(\boldsymbol{J}\boldsymbol{\Xi}^{-1}\boldsymbol{J}^\top\right)^{-1}, \tag{80}$$

where $\boldsymbol{\Xi} \in \mathbb{S}^{n\times n}$ is a weighting matrix. In fact, using such an inverse in (79) amounts to solving the following quadratic optimization (QP) problem with equality constraints:

$$\min_{\ddot{\boldsymbol{q}}} \quad \ddot{\boldsymbol{q}}^\top\boldsymbol{\Xi}\ddot{\boldsymbol{q}} \tag{81a}$$

$$\text{s.t.} \quad \boldsymbol{J}\ddot{\boldsymbol{q}} = \ddot{\boldsymbol{x}} - \dot{\boldsymbol{J}}\dot{\boldsymbol{q}} \tag{81b}$$

Although in the context of hyper-redundant robots this problem might have infinitely many solutions, several different approaches have been suggested in scientific literature to chose among them. The most common approach is to select the solution with *minimal Euclidean*

[18] This is in particular the case for hyper-redundant mechanisms such as humanoid robot, since the Jacobian matrix $\boldsymbol{J}$ is usually not invertible (or even square).

norm, thereby eliminating the most unsafe or physically unattainable solutions. As explained in [130] finding such a solution can be achieved by setting $\boldsymbol{\Xi}$ to identity in (80) resulting in the "*Moore-Penrose*" pseudo-inverse of the Jacobian:

$$\boldsymbol{J}^{\#} = \boldsymbol{J}^{\top}\left(\boldsymbol{J}\boldsymbol{J}^{\top}\right)^{-1} \tag{82}$$

Another alternative, proposed in [131], is to select the solution with minimal instantaneous kinetic energy (in the case of first order inverse kinematics (79a)) by seeing $\boldsymbol{\Xi}$ to the manipulator's generalized inertia matrix $\boldsymbol{M}(\boldsymbol{q})$, thereby resulting in the dynamically consistent pseudo-inverse:

$$\boldsymbol{J}^{\#\boldsymbol{M}} = \boldsymbol{M}^{-1}\boldsymbol{J}^{\top}\left(\boldsymbol{J}\boldsymbol{M}^{-1}\boldsymbol{J}^{\top}\right)^{-1} \tag{83}$$

Weighted generalized inverse approach is appealing as it provides a simple analytical framework for redundancy resolution. However, it has some limitations as infeasible joint velocities tends to be generated when the Jacobian matrix is *ill-conditioned*, which may, for example, arise when the manipulator is getting close to a *kinematic* singularity. This issue was partially addressed in [132] by using a damped inverse of the Jacobian matrix also known as Singularity Robust Inverse (SRI) or Ridge regression:

$$\boldsymbol{J}^{\#\rho} = \boldsymbol{J}^{\top}\left(\boldsymbol{J}\boldsymbol{J}^{\top} + \rho\mathbf{1}_{m\times m}\right)^{-1},\ \rho \in \mathbb{R}^{+} \tag{84}$$

where $\mathbf{1}_{m\times m}$ denotes the $m \times m$ identity matrix. Using such an inverse amounts to solve a so called *damped least-squares problem* instead of (81):

$$\min_{\ddot{\boldsymbol{q}}}\{\left\|\boldsymbol{J}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}\dot{\boldsymbol{q}} - \ddot{\boldsymbol{x}}\right\|_2^2 + \rho\left\|\ddot{\boldsymbol{q}}\right\|_2^2\} \tag{85}$$

Provided that the damping parameter ρ is large enough, this method ensures that *feasible solutions are obtained in the neighborhood of a singular configuration*. Nevertheless, problems may arise when the manipulator moves far from any singularity, as large values of ρ result in a sharp reduction of its precision. Therefore, the use of SRI always requires a compromise between task accuracy (improved by *decreasing* the value of ρ) and task feasibility (improved by *increasing* the value of ρ). As explained in [133], the effects of such a compromise can be mitigated by *dynamically* changing the value of ρ, as a function of the *distance to a singularity* [132] or to its time derivative [134].

2.5.1.2 Hierarchical Task Execution

On redundant robots, as a given task does not necessarily require all the robot's DoF, it is usually possible and even desirable to execute several tasks simultaneously, according to a specific hierarchy. This hierarchy can be soft, through the use of appropriate weights, or hard, meaning that a low priority task should not interfere with the execution of higher priority tasks:

$$\left.\begin{array}{rcl}\dot{\boldsymbol{x}}_1 & = & \boldsymbol{J}_1\dot{\boldsymbol{q}} \\ \Upsilon & \vdots & \Upsilon \\ \dot{\boldsymbol{x}}_n & = & \boldsymbol{J}_n\dot{\boldsymbol{q}}\end{array}\right\}\dot{\boldsymbol{q}} = ? \qquad (86) \qquad \left.\begin{array}{rcl}\ddot{\boldsymbol{x}}_1 & = & \boldsymbol{J}_1\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}_1\dot{\boldsymbol{q}} \\ \Upsilon & \vdots & \Upsilon \\ \ddot{\boldsymbol{x}}_n & = & \boldsymbol{J}_n\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}_n\dot{\boldsymbol{q}}\end{array}\right\}\ddot{\boldsymbol{q}} = ? \qquad (87)$$

According to [135], it is possible to express all the inverse kinematics solutions for a specific hierarchy level in the following form:

$$\ddot{\boldsymbol{q}} = \boldsymbol{J}^{\#}\left(\ddot{\boldsymbol{x}} - \dot{\boldsymbol{J}}\dot{\boldsymbol{q}}\right) + \boldsymbol{N}\ddot{\boldsymbol{q}}_N \tag{88}$$

where $\boldsymbol{N} = (\boldsymbol{I} - \boldsymbol{J}^{\#}\boldsymbol{J}) \in \mathbb{R}^{n \times n}$ is a projector in the nullspace of the Jacobian matrix $\boldsymbol{J}$ and $\ddot{\boldsymbol{q}}_N \in \mathbb{R}^n$ is an arbitrary vector with the dimension of the joint space. The meaning of this expression is that irrespective of the vector $\ddot{\boldsymbol{q}}_N$, the influence of $\boldsymbol{N}\ddot{\boldsymbol{q}}_N$ on the main task $\ddot{\boldsymbol{x}}$, will be deleted. As a result, a strict hierarchy can be obtained between $\ddot{\boldsymbol{x}}$ and the tasks from which the vector ϕ originates. This concept can be generalized to the case of a *n-level hierarchy*, by projecting low priority tasks into the nulls-pace of higher priority tasks [136, 137], yielding the following recursion

$$\begin{aligned}\boldsymbol{N}_0 &= \boldsymbol{1}_{n \times n},\ \ddot{\boldsymbol{q}}_0 = \boldsymbol{0}_{n \times 1} && \text{(89a)}\\ \hat{\boldsymbol{J}}_i &= \boldsymbol{J}_i\boldsymbol{N}_{i-1} && \text{(89b)}\\ \boldsymbol{N}_i &= \boldsymbol{N}_{i-1}\left(\boldsymbol{1}_{n \times n} - \boldsymbol{J}_{i-1}^{\#}\boldsymbol{J}_{i-1}\right) = \boldsymbol{N}_{i-1} - \hat{\boldsymbol{J}}_{i-1}^{\#}\hat{\boldsymbol{J}}_{i-1} && \text{(89c)}\\ \ddot{\boldsymbol{q}} &= \ddot{\boldsymbol{q}}_1 + \sum_{i=2}^{N}\underbrace{(\boldsymbol{J}_i\boldsymbol{N}_{i-1})}_{\hat{\boldsymbol{J}}_i}{}^{\#}\underbrace{\left(\ddot{\boldsymbol{x}}_i - \dot{\boldsymbol{J}}_i\dot{\boldsymbol{q}}_i - \boldsymbol{J}_i\ddot{\boldsymbol{q}}_{i-1}\right)}_{\hat{\ddot{\boldsymbol{x}}}_i} && \text{(89d)}\end{aligned}$$

where $\hat{\ddot{\boldsymbol{x}}}_i$ denote the compensated tasks, where the influence of higher priority levels, from 1 to $i-1$, is removed. Note that tasks hierarchies can also be defined at the torque level [137]:

$$\begin{aligned}\boldsymbol{N}_0 &= \boldsymbol{1}_{n \times n},\ \boldsymbol{\tau}_0 = \boldsymbol{0}_{n \times 1} && \text{(90a)}\\ \hat{\boldsymbol{J}}_i &= \boldsymbol{J}_i\boldsymbol{N}_{i-1} && \text{(90b)}\\ \boldsymbol{N}_i &= \boldsymbol{N}_{i-1}\left(\boldsymbol{1}_{n \times n} - \boldsymbol{J}_{i-1}^{\top}\boldsymbol{J}_{i-1}^{\#\top}\right) = \boldsymbol{N}_{i-1} - \hat{\boldsymbol{J}}_{i-1}^{\top}\hat{\boldsymbol{J}}_{i-1}^{\#\top} && \text{(90c)}\\ \boldsymbol{\tau} &= \boldsymbol{\tau}_1 + \sum_{i=2}^{N}\boldsymbol{N}_i\boldsymbol{\tau}_i && \text{(90d)}\end{aligned}$$

2.5.2. Hierarchical Task Execution Under Constraints

2.5.2.1 Hierarchical Task Execution as a Sequence of Quadratic Programs

The practical implementation of the recursive null-space projection approach ultimately proves tedious because it is in this case rather difficult[19] to enforce the robot's kinematic, dynamic and stability constraints – such as joint position, velocity, torque, self-collision avoidance, or contact friction cones – which is yet particularly critical in the context of floating-base robot control. It is also worth noting that the fusion of kinematic tasks with torque tasks also turns

[19] Although not impossible as proved in [138]

out to be challenging. An alternative approach to hierarchical task execution consists in formulating each task as a convex quadratic optimization problem subject to a convex set of linear constraints (usually referred to as Quadratic Program or QP in the literature). In this context, a lexicographic task hierarchy can be obtained by solving a sequence – or cascade – of QPs, the QPs of higher priority being solved first and their minimizers being used as constraints over lower priority QPs [139, 140, 141, 142, 143, 144].

As investigated in [144] the different tasks and constraints of a floating-base mechanism can be formulated as a set of affine equations of the joint acceleration $\left[\ddot{\boldsymbol{x}}^\top, \ddot{\boldsymbol{q}}^\top\right]^\top \in \mathbb{R}^{n+6}$, joint torque $\boldsymbol{\tau} \in \mathbb{R}^n$ and contact wrenches $\boldsymbol{w}_1, \cdots, \boldsymbol{w}_c \in \mathbb{R}^6$. Accordingly, defining the main optimization variable of the problem as an augmented vector $\boldsymbol{y} = \left[\ddot{\boldsymbol{x}}^\top, \ddot{\boldsymbol{q}}^\top, \boldsymbol{\tau}^\top, \boldsymbol{w}_1^\top, \cdots, \boldsymbol{w}_c^\top\right]^\top \in \mathbb{R}^{2n+6+6c}$ concatenating the configuration accelerations, control torques and contact forces [145], allows handling tasks or constraints defined at the kinematic, dynamic and force level, within the same framework by solving a single QP of the form

$$\begin{aligned} \boldsymbol{y}^\star = &\underset{\boldsymbol{y}}{\arg\min} \quad \|\boldsymbol{B}\boldsymbol{y} - \boldsymbol{b}\|^2 + \rho \|\boldsymbol{y}\|^2 \\ &\text{s.t.} \qquad\qquad \boldsymbol{A}\boldsymbol{y} \leq \boldsymbol{a} \end{aligned} \tag{91}$$

where, similar to (85), $\rho \in \mathbb{R}^+ << 1$ denotes a regularization term, ensuring that the Hessian matrix of the problem remains positive definite, and where the matrices $\boldsymbol{A} \in \mathbb{R}^{\alpha \times (2n+6+6c)}$ and $\boldsymbol{B} \in \mathbb{R}^{\beta \times (2n+6+6c)}$ along with the vectors $\boldsymbol{a} \in \mathbb{R}^\alpha$ and $\boldsymbol{b} \in \mathbb{R}^\beta$ define the inequality constraints and cost function of the problem, respectively.

Following [140, 143, 144], the general formulation for a strict lexicographic task execution framework based on QP makes use of a set of slack variables, typically one per hierarchy level i, here denoted as $\boldsymbol{\varsigma}_i \in \mathbb{R}^{\alpha_i}$ and $\boldsymbol{\iota}_i \in \mathbb{R}^{\beta_i}$, for inequality and equality constraints, respectively. In this context, the first hierarchy level of the cascade – with the highest priority – can be formulated as (91)

$$\begin{aligned} (\boldsymbol{y}_1^\star, \boldsymbol{\varsigma}_1^\star, \boldsymbol{\iota}_1^\star) = &\underset{\boldsymbol{y}_1, \boldsymbol{\varsigma}_1, \boldsymbol{\iota}_1}{\arg\min} \quad \|\boldsymbol{v}_1\|^2 + \|\boldsymbol{w}_1\|^2 + \rho \|\boldsymbol{y}_1\|^2 \\ &\text{s.t.} \qquad\qquad \boldsymbol{V}_1 (\boldsymbol{A}_1 \boldsymbol{y}_1 - \boldsymbol{a}_1) \leq \boldsymbol{\varsigma}_1 \\ &\qquad\qquad\quad\;\; \boldsymbol{W}_1 (\boldsymbol{B}_1 \boldsymbol{y}_1 - \boldsymbol{b}_1) = \boldsymbol{\iota}_1 \end{aligned} \tag{92}$$

where for a given hierarchy level i, the matrices $\boldsymbol{A}_i \in \mathbb{R}^{\alpha_i \times \gamma}$ and $\boldsymbol{B}_i \in \mathbb{R}^{\beta_i \times \gamma}$ along with the vectors $\boldsymbol{a}_i \in \mathbb{R}^{\alpha_i}$ and $\boldsymbol{b}_i \in \mathbb{R}^{\beta_i}$ define the set of inequality and equality constraints of the problem. These constraints can be weighted against each-other using a set of diagonal matrices $\boldsymbol{V}_i \in \mathbb{R}^{\alpha_i \times \alpha_i}$ and $\boldsymbol{W}_i \in \mathbb{R}^{\beta_i \times \beta_i}$ of appropriate dimension, thereby defining a soft hierarchy between the different tasks of a given level of the cascade.

In essence, the slack variables $\boldsymbol{\varsigma}_i$ and $\boldsymbol{\iota}_i$ describe the cost of violating a given constraint. When set as optimization variables and minimized by the solver, they ensure a soft constraints resolution thereby avoiding infeasibility issues. The cornerstone of the implementation of a lexicographic task hierarchy is the optimization of a different set of slack variables for each level of the cascade, using the optimal slacks $\varsigma_1^\star, \cdots, \varsigma_{i-1}^\star$ and $\boldsymbol{\iota}_1^\star, \cdots, \boldsymbol{\iota}_{i-1}^\star$ of the previous levels as hard constraints for the current level

$$\begin{aligned}
(\boldsymbol{y}_i^\star, \varsigma_i^\star, \boldsymbol{\iota}_i^\star) = \underset{\boldsymbol{y}_i, \varsigma_i, \boldsymbol{\iota}_i}{\arg\min} \quad & \|\varsigma_i\|^2 + \|\boldsymbol{\iota}_i\|^2 + \rho\|\boldsymbol{y}_i\|^2 \\
\text{s.t.} \quad & \boldsymbol{V}_i(\boldsymbol{A}_i\boldsymbol{y}_i - \boldsymbol{a}_i) \leq \varsigma_i \\
& \boldsymbol{W}_i(\boldsymbol{B}_i\boldsymbol{y}_i - \boldsymbol{b}_i) = \boldsymbol{\iota}_i \\
& \boldsymbol{V}_{i-1}(\boldsymbol{A}_{i-1}\boldsymbol{y}_i - \boldsymbol{a}_{i-1}) \leq \varsigma_{i-1}^\star \\
& \boldsymbol{W}_{i-1}(\boldsymbol{B}_{i-1}\boldsymbol{y}_i - \boldsymbol{b}_{i-1}) = \boldsymbol{\iota}_{i-1}^\star \\
& \vdots \\
& \boldsymbol{V}_1(\boldsymbol{A}_1\boldsymbol{y}_i - \boldsymbol{a}_1) \leq \varsigma_1^\star \\
& \boldsymbol{W}_1(\boldsymbol{B}_1\boldsymbol{y}_i - \boldsymbol{b}_1) = \boldsymbol{\iota}_1^\star
\end{aligned} \tag{93}$$

2.5.2.2 Improving QP Cascade Efficiency by Iterative Reparameterization

As highlighted in [146, 144], the efficiency of the cascade resolution can be improved by reducing the problem dimentionality from one hierarchy level to the other. This can be achieved in practice by restricting the search space of the quadratic program for a given hierarchy level i, to the kernel of the equality tasks of the previous levels, eventually resulting in the following solution space

$$\begin{aligned}
\boldsymbol{y}_i \quad = \quad & \boldsymbol{y}_{i-1}^\star + \boldsymbol{N}_{i-1}\boldsymbol{u}_i \\
\text{s.t.} \quad & \boldsymbol{V}_{i-1}(\boldsymbol{A}_{i-1}\boldsymbol{y}_i - \boldsymbol{a}_{i-1}) \leq \varsigma_{i-1}^\star \\
& \vdots \\
& \boldsymbol{V}_1(\boldsymbol{A}_1\boldsymbol{y}_i - \boldsymbol{a}_1) \leq \varsigma_1^\star
\end{aligned} \tag{94}$$

where $\boldsymbol{N}_i \in \mathbb{R}^{(2n+6+6c)\times n_z}$ is a ($n_z \in \mathbb{N}$)-dimensional null-space base of the previous level's equality constraints, concatenated into a single augmented matrix denoted as $\hat{\boldsymbol{B}}_i = \left[(\boldsymbol{W}_1\boldsymbol{B}_1)^\top, \cdots, (\boldsymbol{W}_{i-1}\boldsymbol{B}_{i-1})^\top\right]^\top$ and where the slack variable $\boldsymbol{u}_i \in \mathbb{R}^{n_z}$ parameterizes this nullspace. In Herzog et al. [144], $\boldsymbol{N}_i$ is obtained using a singular value decomposition of the augmented matrix $\hat{\boldsymbol{B}}_i$:

$$\hat{\boldsymbol{B}}_i = \begin{bmatrix}\boldsymbol{U}_R & \boldsymbol{U}_N\end{bmatrix} \begin{bmatrix}\sigma_1 & & & \\ & \ddots & & \\ & & \sigma_{2n+6c-n_z} & \\ & & & \boldsymbol{0}_{n_z\times n_z}\end{bmatrix} \begin{bmatrix}\boldsymbol{V}_R \\ \boldsymbol{V}_N = \boldsymbol{N}_i\end{bmatrix} \tag{95}$$

It is worth emphasizing that such computation can be made way more efficient using a full QR decomposition or even a LU decomposition of $\hat{\boldsymbol{B}}_i$. These aspects are being further discussed in the *mathematical toolbox* in appendix B.B.1. This iterative nullspace mapping effectively reduces the number of variables from one hierarchy level to the next. Finally as highlighted in [143], formulating the constraints as two-sided inequalities allows to further increase the computational efficiency as the upper and lower bounds constraints cannot be activated simultaneously within the solver, eventually yielding:

$$\begin{aligned}
(\boldsymbol{y}_i^\star, \boldsymbol{u}_i^\star, \varsigma_i^\star) = \underset{\boldsymbol{u}_i, \varsigma_i}{\arg\min} \left\| \boldsymbol{W}_i \left(\boldsymbol{B}_i \boldsymbol{y}_i - \boldsymbol{b}_i \right) \right\|^2 + \left\| \varsigma_i \right\|^2 + \varepsilon \left\| \boldsymbol{y}_i \right\|^2 \\
\text{s.t. } \boldsymbol{y}_i = \boldsymbol{y}_{i-1}^\star + \boldsymbol{Z}_{i-1} \boldsymbol{u}_i \\
\boldsymbol{V}_i \boldsymbol{a}_{i_{min}} \leq \boldsymbol{V}_i \boldsymbol{A}_i \boldsymbol{y}_i - \varsigma_i \leq \boldsymbol{V}_i \boldsymbol{a}_{i_{max}} \\
\boldsymbol{V}_{i-1} \boldsymbol{a}_{i-1_{min}} \leq \boldsymbol{V}_{i-1} \boldsymbol{A}_{i-1} \boldsymbol{y}_i - \varsigma_{i-1}{\star} \leq \boldsymbol{V}_{i-1} \boldsymbol{a}_{i-1_{max}} \\
\vdots \\
\boldsymbol{V}_1 \boldsymbol{a}_{1_{min}} \leq \boldsymbol{V}_1 \boldsymbol{A}_1 \boldsymbol{y}_i - \varsigma_1^\star \leq \boldsymbol{V}_1 \boldsymbol{a}_{1_{max}}
\end{aligned} \tag{96}$$

2.5.3. Task Formulation

Formulating tasks at the Cartesian acceleration level has multiple advantages. Although acceleration is a kinematic signal, it is tightly coupled with dynamics and thus allows bridging the gap between kinematic and dynamic tasks, as we will demonstrate later in this thesis. The different control tasks being defined as a set of desired Cartesian or joint-space position and velocity objectives, denoted as $\boldsymbol{x}_d \in \mathbb{R}^m$, the corresponding joint acceleration reference will have the following form

$$\ddot{\boldsymbol{x}}_r = \ddot{\boldsymbol{x}}_d + \boldsymbol{K}_d \left(\dot{\boldsymbol{x}}_d - \dot{\boldsymbol{x}} \right) + \boldsymbol{K}_p \left(\boldsymbol{x}_d - \boldsymbol{x} \right) \tag{97}$$

where $\ddot{\boldsymbol{x}}_r$ is the commanded acceleration, $\boldsymbol{x}_d$, $\boldsymbol{x}$ and their time derivatives denote the reference (resp. actual) position, velocity and accelerations, while the matrices $\boldsymbol{K}_p \in \mathbb{S}^{m \times m}$ and $\boldsymbol{K}_d \in \mathbb{S}^{m \times m}$ contain the stiffness and damping coefficients associated to the task. As explained in [147, 148], the gain matrices $\boldsymbol{K}_p$ and $\boldsymbol{K}_d$ are often selected so that the matrix $\begin{bmatrix} \boldsymbol{0}_{m \times m} & \boldsymbol{1}_{m \times m} \\ -\boldsymbol{K}_p & -\boldsymbol{K}_d \end{bmatrix}$ has eigenvalues with a strictly negative real part (i.e. is a Hurwitz matrix). On a single hierarchy level i, for the tasks denoted by the Jacobian matrices $\boldsymbol{J}_1, \boldsymbol{J}_2$ and $\boldsymbol{J}_k$ with weights $\boldsymbol{W}_{i1}, \boldsymbol{W}_{i2}$ and $\boldsymbol{W}_{ik}$, the cost of the QP can be formulated[20] as

$$\boldsymbol{W}_i \boldsymbol{B}_i = \begin{bmatrix} \boldsymbol{W}_{i1} \boldsymbol{J}_1 & \boldsymbol{0}_{m \times 3cg} \\ \boldsymbol{W}_{i2} \boldsymbol{J}_2 & \boldsymbol{0}_{m \times 3cg} \\ \vdots & \vdots \\ \boldsymbol{W}_{ik} \boldsymbol{J}_k & \boldsymbol{0}_{m \times 3cg} \end{bmatrix} \in \mathbb{R}^{km \times (n+6+3cg)}, \qquad \boldsymbol{W}_i \boldsymbol{b}_i = \begin{bmatrix} \boldsymbol{W}_{i1} \left(\ddot{\boldsymbol{x}}_{r_1} - \dot{\boldsymbol{J}}_1 \dot{\boldsymbol{q}} \right) \\ \boldsymbol{W}_{i2} \left(\ddot{\boldsymbol{x}}_{r_2} - \dot{\boldsymbol{J}}_2 \dot{\boldsymbol{q}} \right) \\ \vdots \\ \boldsymbol{W}_{ik} \left(\ddot{\boldsymbol{x}}_{r_k} - \dot{\boldsymbol{J}}_k \dot{\boldsymbol{q}} \right) \end{bmatrix} \in \mathbb{R}^{km} \tag{98}$$

where $\boldsymbol{W}_{i1}$ refer to diagonal blocks of the weight matrix $\boldsymbol{W}_i$. It is worth noting that torque tasks can also be integrated into this framework by minimizing the L2 norm of the difference

[20] Note that the term $\dot{\boldsymbol{J}}_i \dot{\boldsymbol{q}}$ is often referred to as the *normal acceleration*. This is in particular the case in [149].

between a reference torque $\boldsymbol{\tau}_r$ and the robot inverse dynamics:

$$\arg\min_{\ddot{\boldsymbol{q}}} \left\| \boldsymbol{M}_q \ddot{\boldsymbol{q}} + \boldsymbol{h}_q + \sum_{i=1}^{kcg} \boldsymbol{J}_{q_i}^{\top} \boldsymbol{\Gamma}_i \boldsymbol{\lambda}_i - \boldsymbol{\tau}_r \right\|^2 . \tag{99}$$

yielding

$$\boldsymbol{W}_i \boldsymbol{B}_i = \begin{bmatrix} \boldsymbol{W}_{i1} \boldsymbol{M}_q & -\boldsymbol{W}_{i1} \boldsymbol{J}_{q_1}^{\top} \boldsymbol{\Gamma}_1 & \dots & -\boldsymbol{W}_{i1} \boldsymbol{J}_{q_{c\times g}}^{\top} \boldsymbol{\Gamma}_{c\times g} \\ \boldsymbol{W}_{i2} \boldsymbol{M}_q & -\boldsymbol{W}_{i2} \boldsymbol{J}_{q_1}^{\top} \boldsymbol{\Gamma}_1 & \dots & -\boldsymbol{W}_{i2} \boldsymbol{J}_{q_{c\times g}}^{\top} \boldsymbol{\Gamma}_{c\times g} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{W}_{ik} \boldsymbol{M}_q & -\boldsymbol{W}_{ik} \boldsymbol{J}_{q_1}^{\top} \boldsymbol{\Gamma}_1 & \dots & -\boldsymbol{W}_{ik} \boldsymbol{J}_{q_{c\times g}}^{\top} \boldsymbol{\Gamma}_{c\times g} \end{bmatrix} \in \mathbb{R}^{kn\times(n+3cg)}, \tag{100}$$

and

$$\boldsymbol{W}_i \boldsymbol{b}_i = \begin{bmatrix} \boldsymbol{W}_{i1} \left(\boldsymbol{\tau}_{r_1} - \boldsymbol{h}_q \right) \\ \boldsymbol{W}_{i2} \left(\boldsymbol{\tau}_{r_2} - \boldsymbol{h}_q \right) \\ \vdots \\ \boldsymbol{W}_{ik} \left(\boldsymbol{\tau}_{r_k} - \boldsymbol{h}_q \right) \end{bmatrix} \in \mathbb{R}^{kn} \tag{101}$$

For instance in the context of a compliance task, the reference torque $\boldsymbol{\tau}_r$ can be formulated as:

$$\boldsymbol{\tau}_r = \boldsymbol{J}_q^{\top} \boldsymbol{f} \tag{102}$$

where $\boldsymbol{f}$ is an external force applied in a point of the robot's kinematic tree.

2.5.4. Constraint Formulation for Floating-Base Robots

2.5.4.1 Underactuation, wrench distribution, kinematic and dynamic constraints

The main set of constraints that must be guaranteed on any type of robot are the joint position, velocity and torque constraints, possibly combined with acceleration constraints:

$$\ddot{\boldsymbol{q}}_{min} \leq \ddot{\boldsymbol{q}} \leq \ddot{\boldsymbol{q}}_{max} \tag{103a}$$

$$\dot{\boldsymbol{q}}_{min} \leq \dot{\boldsymbol{q}} \leq \dot{\boldsymbol{q}}_{max} \tag{103b}$$

$$\boldsymbol{q}_{min} \leq \boldsymbol{q} \leq \boldsymbol{q}_{max} \tag{103c}$$

$$\boldsymbol{\tau}_{min} \leq \boldsymbol{\tau} \leq \boldsymbol{\tau}_{max} \tag{103d}$$

Given the nature of the optimization variable $\boldsymbol{y}$ within the stack of task, these constraints must be formulated either in terms of joint acceleration $\ddot{\boldsymbol{q}}$ or in terms of torque $\boldsymbol{\tau}$. Kinematic constraints at the joint velocity and joint position levels can easily be reformulated as acceleration constrains [143, 150, 151, 147], knowing the control loop period δt, and can thus be seamlessly integrated into (96)

$$\begin{aligned}\frac{\dot{\boldsymbol{q}}_{min}-\dot{\boldsymbol{q}}}{\delta t} &\leq \ddot{\boldsymbol{q}} \leq \frac{\dot{\boldsymbol{q}}_{max}-\dot{\boldsymbol{q}}}{\delta t} && \text{(104a)}\\ \frac{2\left(\boldsymbol{q}_{min}-\boldsymbol{q}-\dot{\boldsymbol{q}}\delta t\right)}{\delta t^2} &\leq \ddot{\boldsymbol{q}} \leq \frac{2\left(\boldsymbol{q}_{max}-\boldsymbol{q}-\dot{\boldsymbol{q}}\delta t\right)}{\delta t^2} && \text{(104b)}\end{aligned}$$

Although torque constraints can be expressed in a direct straight-forward manner [152], it might also be tempting to further reduce the dimentionality of the QP, considering that the inverse dynamics model, expressed as a constraint, allows removing the torque term from the vector $\boldsymbol{y}$, thereby increasing efficiency, as for example proposed in [151, 144, 147, 153]

$$\underbrace{-\boldsymbol{h}+\begin{bmatrix}\boldsymbol{0}_6\\ \boldsymbol{\tau}_{min}\end{bmatrix}}_{\boldsymbol{a}_{min}} \leq \underbrace{\begin{bmatrix}\boldsymbol{M} & -\boldsymbol{J}_1^\top & \dots & -\boldsymbol{J}_c^\top\end{bmatrix}}_{\boldsymbol{A}} \underbrace{\begin{bmatrix}\begin{bmatrix}\ddot{\boldsymbol{x}}\\ \ddot{\boldsymbol{q}}\end{bmatrix}\\ \boldsymbol{w}_1\\ \vdots\\ \boldsymbol{w}_c\end{bmatrix}}_{\boldsymbol{y}} \leq \underbrace{\begin{bmatrix}\boldsymbol{0}_6\\ \boldsymbol{\tau}_{max}\end{bmatrix}-\boldsymbol{h}}_{\boldsymbol{a}_{max}} \qquad \text{(105a)}$$

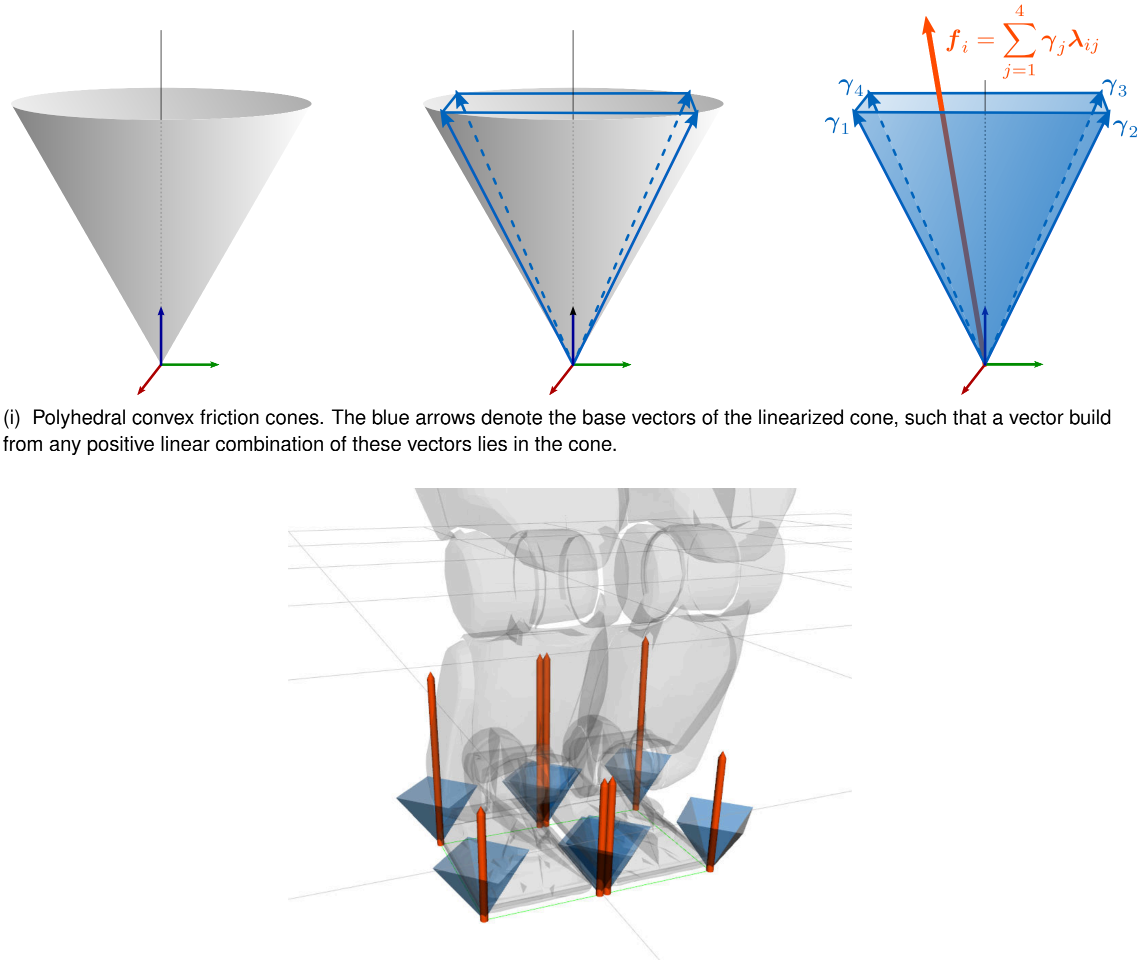
$$\text{where: } \underbrace{\begin{bmatrix}\boldsymbol{M}_b & \boldsymbol{M}_{bq}\\ \boldsymbol{M}_{qb} & \boldsymbol{M}_q\end{bmatrix}}_{\boldsymbol{M}} \begin{bmatrix}\ddot{\boldsymbol{x}}\\ \ddot{\boldsymbol{q}}\end{bmatrix} + \underbrace{\begin{bmatrix}\boldsymbol{h}_b\\ \boldsymbol{h}_q\end{bmatrix}}_{\boldsymbol{h}} = \begin{bmatrix}\boldsymbol{0}_{6\times 1}\\ \boldsymbol{\tau}\end{bmatrix} + \sum_{i=1}^{c} \underbrace{\begin{bmatrix}\boldsymbol{J}_{b_i}^\top\\ \boldsymbol{J}_{q_i}^\top\end{bmatrix}}_{\boldsymbol{J}_i^\top} \boldsymbol{w}_i\,. \qquad \text{(105b)}$$

It is worth noting that such formulation allows the contact wrench terms in the vector $\boldsymbol{y}$ to be optimally distributed, provided of course that the robot dynamic model and parameters are known with a sufficient precision.

2.5.4.2 Contact constraints

Floating base mechanisms are required to be capable of distributing their contact wrenches with the external environment for stabilization purposes. The main requirements for stability are here that the contact points used for robot stabilization purposes should not move, or in other words, the robot should not slide. This has to be encoded both kinematically and at the contact force level. From the kinematic level, each of the considered contact points should verify the following condition:

$$\boldsymbol{x}_i = \boldsymbol{x}_{cst} \Leftrightarrow \boldsymbol{J}_i\dot{\boldsymbol{q}} = \dot{\boldsymbol{x}}_i = \boldsymbol{0}_{6\times 1}, \forall i = \{1, \cdots, c\} \qquad \text{(106)}$$



(i) Polyhedral convex friction cones. The blue arrows denote the base vectors of the linearized cone, such that a vector build from any positive linear combination of these vectors lies in the cone.

(ii) Polyhedral convex friction cones with four generators (friction pyramids) on the REEM-C humanoid robot.

Figure 10 Detail of a point mass distribution on the torso link of our REEM-C robot.

It should also be emphasized in this case that using one single six-dimensional Jacobian per group of contact points is often more convenient and efficient than using the Jacobian matrices of each of these points. This expression can also be formulated at the acceleration level as

$$\boldsymbol{J}_i\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}_i\dot{\boldsymbol{q}} = \boldsymbol{0}\,. \tag{107}$$

Note that in practice, the previous formulation tends to be unstable and the following velocity formulation is often preferred [149, 150]:

$$\boldsymbol{S}_i\left(\boldsymbol{J}_i\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}_i\dot{\boldsymbol{q}}\right) = -\boldsymbol{S}_i\frac{1}{dt}\dot{\boldsymbol{x}}_i \tag{108}$$

where $\boldsymbol{S}_i \in \mathbb{S}^{6\times 6}$ is a selection matrix, allowing to relax some directions of the constraint for contact admittance control purposes. Besides the kinematic constraints, it is also essential that the contact wrenches distribution prevent the robot from slipping. Consequently, these must comply with unilaterality constraints and be moreover restricted to a set of appropriate friction cones. Friction cones constraints can be formulated at the wrench level or alternatively at the force level [154, 143], by decomposing each contact surfaces into a set of k

distinct contact points submitted to the linear forces $\boldsymbol{f} \in \mathbb{R}^3$ (c.f. Fig. 10ii). Although the distributed contact approach is more computationally expensive than the single contact wrench approach, it turns out to be more versatile in practice, as it is better fitted for partial foothold and allows, by means of unilateralism of the contact force, to enforce the motions of the CoP to lie inside the support polygon. To be expressed as constraints within the stack of task, the friction cones are linearized, each cone being in fact described as a number g of vectors $\gamma_j \in \mathbb{R}^3$ (c.f. Fig. 10i), serving as generators for the contact force. These generators are stored column-wise in a matrix $\mathbf{\Gamma}_i \in \mathbb{R}^{3\times g}$, and parameterized by a set of vectors $\boldsymbol{\lambda}_i \in \mathbb{R}^g$ in order to reconstruct the constrained set of contact forces $\boldsymbol{f}_i$

$$\boldsymbol{f}_i = \sum_{j=1}^{g} \gamma_j \boldsymbol{\lambda}_{ij} = \mathbf{\Gamma}_i \boldsymbol{\lambda}_i \in \mathbb{R}^3 \, , \tag{109}$$

eventually yielding the following reformulation of the constraint (105), for :

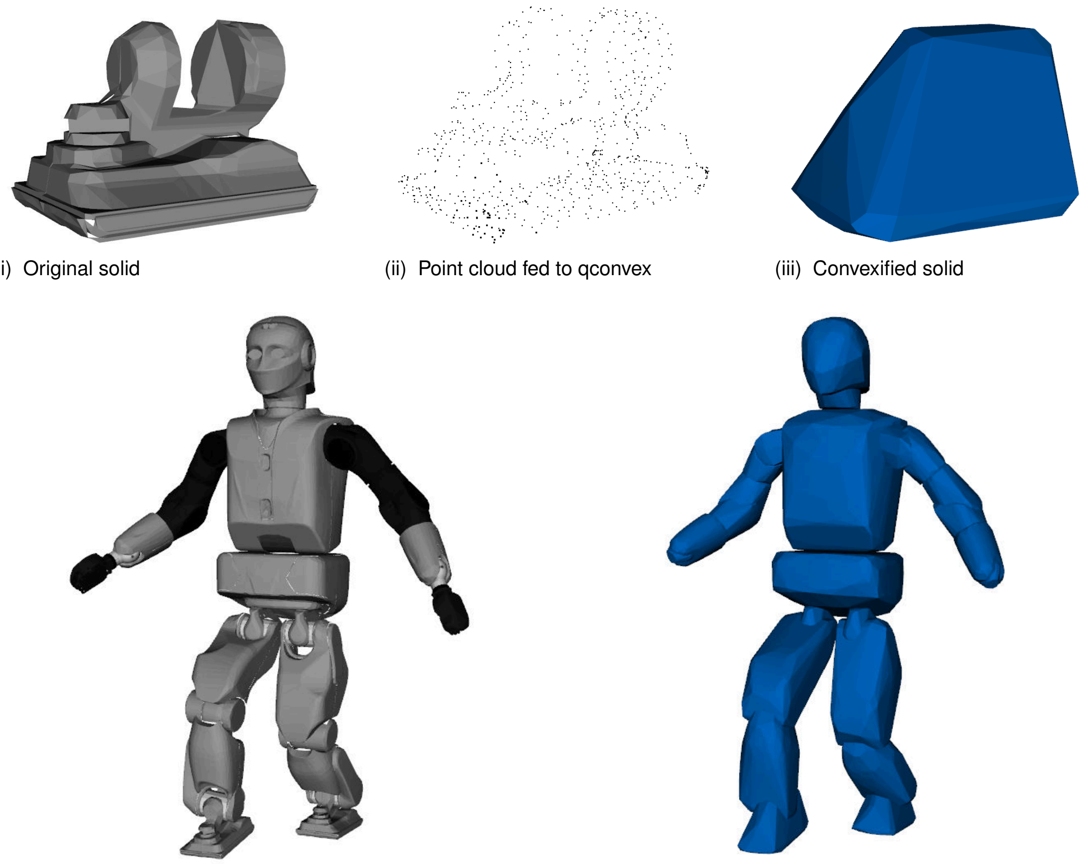
$$\underbrace{\begin{bmatrix} -\boldsymbol{h} + \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau}_{min} \end{bmatrix} \\ -\boldsymbol{S}_1\left(\dot{\boldsymbol{J}}_1 + \frac{1}{dt}\dot{\boldsymbol{x}}_1\right) \\ \vdots \\ -\boldsymbol{S}_c\left(\dot{\boldsymbol{J}}_c + \frac{1}{dt}\dot{\boldsymbol{x}}_c\right) \end{bmatrix}}_{\boldsymbol{a}_{min}} \leq \underbrace{\begin{bmatrix} \boldsymbol{M} & -\boldsymbol{G}_1^{\top}\mathbf{\Gamma}_1 & \cdots & -\boldsymbol{G}_{kcg}^{\top}\mathbf{\Gamma}_{kcg} \\ \boldsymbol{S}_1\boldsymbol{J}_1 & \mathbf{0}_{6\times g} & \cdots & \mathbf{0}_{6\times g} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{S}_c\boldsymbol{J}_c & \mathbf{0}_{6\times g} & \cdots & \mathbf{0}_{6\times g} \end{bmatrix}}_{\boldsymbol{A}} \underbrace{\begin{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{x}} \\ \ddot{\boldsymbol{q}} \end{bmatrix} \\ \boldsymbol{\lambda}_1 \\ \vdots \\ \boldsymbol{\lambda}_{kcg} \end{bmatrix}}_{\boldsymbol{y}} \leq \underbrace{\begin{bmatrix} \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau}_{max} \end{bmatrix} - \boldsymbol{h} \\ -\boldsymbol{S}_1\left(\dot{\boldsymbol{J}}_1 + \frac{1}{dt}\dot{\boldsymbol{x}}_1\right) \\ \vdots \\ -\boldsymbol{S}_c\left(\dot{\boldsymbol{J}}_c + \frac{1}{dt}\dot{\boldsymbol{x}}_c\right) \end{bmatrix}}_{\boldsymbol{a}_{max}}$$

where: $$\underbrace{\begin{bmatrix} \boldsymbol{M}_b & \boldsymbol{M}_{bq} \\ \boldsymbol{M}_{qb} & \boldsymbol{M}_q \end{bmatrix}}_{\boldsymbol{M}} \begin{bmatrix} \ddot{\boldsymbol{x}} \\ \ddot{\boldsymbol{q}} \end{bmatrix} + \underbrace{\begin{bmatrix} \boldsymbol{h}_b \\ \boldsymbol{h}_q \end{bmatrix}}_{\boldsymbol{h}} = \begin{bmatrix} \mathbf{0}_{6\times 1} \\ \boldsymbol{\tau} \end{bmatrix} + \sum_{i=1}^{kcg} \underbrace{\begin{bmatrix} \boldsymbol{G}_{b_i}^{\top} \\ \boldsymbol{G}_{q_i}^{\top} \end{bmatrix}}_{\boldsymbol{G}_i^{\top}} \mathbf{\Gamma}_i \boldsymbol{\lambda}_i \, . \tag{110}$$

the matrix $\boldsymbol{G}_i \in \mathbb{R}^{3\times n+6}$ denoting the linear part of the Jacobian matrix $\boldsymbol{J}_i \in \mathbb{R}^{6\times n+6}$ associated with contact point i.

2.5.4.3 Self-collision avoidance

Despite its apparent simplicity, the problem of self-collision avoidance has been the subject of much research over the last decade. Not only does this task requires to know, at each control epoch, the minimum Cartesian distance between a set of rigid bodies in a given robot, but more importantly, this distance must not fluctuate too abruptly when the robot links move or change their relative orientation in order to avoid causing unstable behaviors at the controller level. The typical approach to rapidly compute the minimal distance between two solids is to simplify them as set of convex bounding boxes. This was for instance proposed in [155, 156], while the distance continuity issue was investigated in [157, 156]. In practice, as presented in Figure 11, the different robot links, taken from the CAD model, can be transformed au-



(i) Original solid

(ii) Point cloud fed to qconvex

(iii) Convexified solid

(iv) Robot visual model

(v) Robot convex collision model

Figure 11 Illustration of the convex collision model of the REEM-C robot

tomatically into simplified convex volumes using a dedicated software library such as qhull [158]. These solids should then be regularized into Sphere-Torus-Patches Bounding Volumes (STP-BV) following the approach of [156]. Using the method of Benallegue, Escande et al. [159, 160] then allows performing fast queries of the minimum distance d between different STP-BV collision solids while avoiding abrupt distance fluctuations as the robot is moving. As a result, it becomes possible to execute stable collision monitoring between a set of predefined different robot bodies in a real-time control loop.

Let us denote by $\boldsymbol{x}_i \in \mathbb{R}^3$ and $\boldsymbol{x}_j \in \mathbb{R}^3$ the pair of closest points between two convex solids labeled as i and j. Then the distance d_{ij} between these two points will be expressed as $d_{ij} = \|\boldsymbol{x}_j - \boldsymbol{x}_i\|$ along the direction $\boldsymbol{n}_{ij} = \frac{1}{d_{ij}} (\boldsymbol{x}_j - \boldsymbol{x}_i)$. Accordingly the relative velocity $\dot{d}_{ij}$ can be formulated as

$$\dot{d} = \boldsymbol{n}_{ij}^{\top} (\boldsymbol{G}_j - \boldsymbol{G}_i)\, \dot{\boldsymbol{q}} = \boldsymbol{n}_{ij}^{\top} \boldsymbol{G}_{ij} \dot{\boldsymbol{q}} \tag{111}$$

yielding by time derivation:

$$\ddot{d} = \left(\dot{\boldsymbol{n}}_{ij}^{\top} \boldsymbol{G}_{ij} + \boldsymbol{n}_{ij}^{\top} \dot{\boldsymbol{G}}_{ij} \right) \dot{\boldsymbol{q}} + \boldsymbol{n}_{ij}^{\top} \boldsymbol{G}_{ij} \ddot{\boldsymbol{q}} \tag{112}$$

In [161, 150, 147], the authors proposed to formulate collision avoidance problem in the form of an inequality constraint of the form

$$\dot{d} \geq \zeta \frac{d - \delta_s}{\delta_i - \delta_s} \tag{113}$$

where following the notations of [150, 147] $\zeta \in \mathbb{R}_+$ refers to the velocity damping coefficient, $\delta_i \in \mathbb{R}_+$ is the distance from which the constraint starts to act and $\delta_s \in \mathbb{R}_+$ is the minimum allowed distance between the considered bodies. Time derivation then yields the acceleration level formulation of the constraint

$$\ddot{d} \geq \frac{1}{dt} \left(-\zeta \frac{d - \delta_s}{\delta_i - \delta_s} - \dot{d} \right) \tag{114}$$

As a result, and following the approach of [149], for a number Ns of monitored collisions, the collision avoidance constraint can be formulated as follows and included into the stack of task:

$$\underbrace{\begin{bmatrix} \dot{d}_1 + \dot{q}^\top \boldsymbol{G}_1^\top \left(\boldsymbol{n}_1 + \dot{\boldsymbol{n}}_1 dt\right) + \dot{q}^\top \dot{\boldsymbol{G}}_1^\top \boldsymbol{n}_1 dt \\ \dot{d}_2 + \dot{q}^\top \boldsymbol{G}_2^\top \left(\boldsymbol{n}_2 + \dot{\boldsymbol{n}}_2 dt\right) + \dot{q}^\top \dot{\boldsymbol{G}}_1^\top \boldsymbol{n}_2 dt \\ \vdots \\ \dot{d}_{Ns} + \dot{q}^\top \boldsymbol{G}_{Ns}^\top \left(\boldsymbol{n}_{Ns} + \dot{\boldsymbol{n}}_{Ns} dt\right) + \dot{q}^\top \dot{\boldsymbol{G}}_{Ns}^\top \boldsymbol{n}_{Ns} dt \end{bmatrix}}_{\boldsymbol{a}_{min}} \leq \underbrace{\begin{bmatrix} -\boldsymbol{n}_1^\top \boldsymbol{G}_1 dt & \boldsymbol{0}_{1\times 3cg} \\ -\boldsymbol{n}_2^\top \boldsymbol{G}_2 dt & \boldsymbol{0}_{1\times 3cg} \\ \vdots & \vdots \\ -\boldsymbol{n}_{Ns}^\top \boldsymbol{G}_{Ns} dt & \boldsymbol{0}_{1\times 3cg} \end{bmatrix}}_{\boldsymbol{A}} \begin{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{x}} \\ \ddot{\boldsymbol{q}} \end{bmatrix} \\ \boldsymbol{\lambda}_1 \\ \vdots \\ \boldsymbol{\lambda}_{c\times g} \end{bmatrix} \tag{115}$$

2.6. Summary

Inertial parameter identification is a vast and active research topic, with multiple contributions ranging from robotics to control in general. This chapter provides an overview of some of the most popular identification methods used in robotics, compiled after a systematic and rigorous analysis of the scientific literature. The common points and differences between these methods are highlighted and discussed from a theoretical perspective. Besides the essential steps involved in every identification process, the main challenges related to fixed and floating base robot identification, in terms of excitability, physicality, and rank deficiency of the observation matrix, are introduced and discussed. By also providing an introduction to the concepts of constrained hierarchical task execution on hyper-redundant floating-base mechanisms, this chapter is intended to serve as a theoretical foundation for the rest of this thesis.

Chapter 3

Distributed Tactile Feedback for Compliant Control Applications on Mobile and Floating-Base Robots

This chapter presents the contributions of this thesis in the field of whole-body compliance for human-robot physical interaction purposes. Exploiting the contact and precontact modalities of an artificial skin covering a mobile and a floating-base robot, we formulate the compliance control law at the contact point level, in the form of a set of acceleration tasks. This control objective is then included into a resolved torque control framework in order to generate suitable reactive motion on the robot. The components of interaction force that cannot produce suitable reactive motion can be identified and projected onto the mobile – or floating – base of the robot in order to extend its compliance range. This chapter is structured as follows: section 3.1 introduces the motivation and related works, while section 3.2 provides a formal description of the whole-body contact point compliance problem. Section 3.3 then describes the proposed approach for whole-body multiple-contact compliance with force propagation using tactile feedback and quadratic programming. Section 3.4 discusses the propagation strategy for the contact wrench residual. The experiments, as well as their results, are presented and discussed in Section 3.5. Finally, the section 3.6 provides a short conclusion on the accomplished work.

3.1. Towards Whole-Body Compliance for Potentially Safe Human-Robot Interaction

3.1.1. Motivation and Related Works

To be safely integrated within complex human environments, robots must be able to cope with a wide variety of physical interactions, whether desired or accidental, in a manner that is both safe and predictable to the external user [4]. Accordingly, these robots must be whole-body compliant, which means that the interaction forces applied to any part of their body must be propagated along their kinematic tree and consequently generate appropriate – potentially safe – reactive motions. Robot compliance has been extensively studied over the past two decades [162, 131, 163, 164, 165, 166, 167, 168, 169]. Commonly exploited in the industry for mechanical assembly purposes [170, 171] or the realization of cooperative tasks [172], it is now being increasingly used in the context of rehabilitation [173] and assistive robotics [174]. However, its generalization to multi-contact interaction scenarios between humans and robots remains difficult for two main reasons. First of all, in this context, compliance is no longer restricted to the end-effectors, as it is usually the case on most robots. Instead it must be guaranteed for an arbitrary number of contact points distributed throughout the entire body of the considered mechanism. As a result, such a robot must either have a soft mechanical structure, associated with a tailor-made actuation strategy, e.g., variable stiffness [165, 175, 176], or must otherwise be equipped with specific sensor modalities allowing suitable reactive behaviors to be emulated by its controller [177, 178, 179]. On the other hand, the generation of compliant behaviors must be guaranteed despite the kinematic and dynamic limitations of the considered robot, such as joint limits, torque limits, self-collision or singularities. In practice, compliance may not be achieved in all directions of space, due to kinematic or dynamic constraints. This is especially problematic for fixed-base robots, such as industrial manipulators, since any physical interaction with the root of the kinematic chain or with a singular axis of motion, systematically leads to the emergence of stiff configurations (c.f. Fig. 20).

In contrast, mobile and floating-base robots can exploit their locomotion system to continue propagating these forces, thereby potentially increasing their compliance range. The issue of coordination between the locomotor limbs and the rest of the robot must of course be properly addressed to ensure that the generated behaviors are consistent and – possibly[1] – safe for an external user. In the case of mobile robots, the typical approach is to use an admittance-controlled lower body [180] in combination with a passive or active-compliant upper body. The mobile base control interface is usually formulated in the operational space. This is, for example, the case in [181], where a force sensor placed at the end-effector of a single-arm mobile manipulator, provides direct control over the movements of its base. In [182, 183, 184, 178, 185, 186] the Cartesian position error between a compliant frame, linked to the robot's end-effector, and a fixed frame attached to its lower body is used by the mobile base controller to generate suitable reactive motions. Additional stability and posture objectives are included in [184] with good performances. Although this technique allows for greater flexibility – given that the mobile base motion generation is no longer restricted to the end-effector – it may be vulnerable to singularities since it relies on the compliant motion of the upper-body with respect to a fixed point in the base frame. In [187, 188, 189, 190, 191, 192, 193, 194], whole-body compliance is achieved by placing a force-torque sensor either at the mechanical coupling between the robot's upper and lower body or directly on its mobile base. Simple and naturally robust to singularities, this strategy may nevertheless be sensitive to the effects of internal forces, resulting from upper-body's inertia. Generating motion on the mobile base with a compliant upper body can also be achieved at the joint level. This is, for example, the case in [195, 196, 197] where some specific joints of a compliant upper body are assigned with a particular degree of freedom on the mobile base. This method works well provided that the arms do not get too far from their nominal posture, in which case, base movements may become inconsistent. In [198], an approach based on generalized inverse dynamics exploiting the feedback of dedicated joint torque sensors, for whole-body compliance was implemented on a dual-arm wheeled mobile

[1] Interaction safety is a topic of its own, involving a specific set of metrics to determine how dangerous physical contact between humans and robots really is. This topic is considered beyond the scope of this research.

manipulator. Although intrinsically compliant, robots using joint-torque feedback are in general vulnerable to singularities, since a force applied to a singular posture will not generate any detectable torque. In practice, realizing compliant behaviors on a stiff robot is often challenging since major structural modifications are necessary to provide the considered platform with either adequate sensory modalities (e.g. force-torque sensors), or with specific actuators (e.g. variable stiffness). In this context, the concept of whole-body tactile feedback is a relevant alternative, since covering an entire robot with an artificial skin to get suitable force or application point measurements only requires limited structural changes. Widely investigated over the past two decades, several promising prototypes have since then been successfully developed and tested, such as [199, 200, 201, 202, 203, 6, 179].

Although whole-body compliance was already widely investigated in the case of mobile manipulators [204, 5], it remains challenging in the case of floating-base legged robots, such as humanoid robots, due to tilting and instabilities issues. On a floating base robot, compliant reactive motion generation is usually achieved implicitly, at the stabilizer level. The incorrectly propagated components of upper-body interaction forces are not directly measured. Instead, their effect on the position of the robot's center of pressure (CoP) is assessed using the ankle force-torque sensors and utilized for step generation [205, 206]. As a result, the step execution has to be achieved at high velocities, since the robot is compensating for the effects of unpropagated forces, not for the forces as such. In this context, the use of a distributed tactile feedback appears to be relevant since, unlike proprioceptive feedback, it allows to directly measure the propagated components of external interaction forces applied to the robot and are likely to have the floating base move. These components can then be directly used in order to trigger suitable reactive motion on the locomotor limbs, well before the equilibrium of the robot is affected [207]. Therefore in this context, the use of artificial skin considerably simplifies the implementation of complex behaviors. Another important point is the integration of physical interaction forces within the robot control framework. Humanoid robots often have to deal with a wide range of potentially conflicting control tasks such as posture objective, stability and manipulation of a heavy object. Since these tasks cannot be entirely fulfilled at the same time, they have to be integrated into a hierarchical execution framework depending on their level of priority [208], the least important tasks being recursively projected into the nullspace of the most important tasks in order to avoid disturbing their execution. Ideally, the whole-body compliance task must also be integrated into such framework in order to properly distribute the reactive motions on the robot kinematic tree depending on the context of interaction. Therefore, it must be associated with a proper set of Jacobian matrices. The use of distributed tactile feedback is here relevant since it not only provides information about the contact force, but also about the position of each contact in the robot kinematic tree.

3.1.2. Proposed Approach and Contribution

We propose to formulate the compliance control law in Cartesian space. The desired reactive dynamics, defined at the contact point level is being tracked by the controller. Quadratic Programming (QP) makes it possible to compute a set of proper control references, either at the torque or acceleration level, thereby allowing the robot to follow the desired reactive behavior as accurately as possible, while intrinsically complying with a set of predefined physical constraints. Cartesian formulation makes it possible to conveniently capture the dynamic residuals between the desired and the actual reactive motions of each robot limb. These residuals are then propagated onto a parent kinematic chain – in our case the mobile or floating base of the robot – to be adequately compensated.

3.2. Whole-Body Tactile Compliance Task Formulation

Controlling the mechanical impedance of a robot manipulator can be achieved at the joint level [165], at the end-effector level [162] or at the contact point level [209, 210, 211]. Contact point compliance control (also referred to as contact point impedance control) is of particular relevance in the context of whole-body human-robot interaction since it allows to precisely adjust the robot reactive behaviors at any point of its kinematic chain [209]. The corresponding control objective can be formulated as determining the control reference $\boldsymbol{\tau}_r$ or $\ddot{\boldsymbol{q}}_r$ providing a specific contact point j, with a desired dynamic behavior ${}^0\ddot{\boldsymbol{x}}_{d_j}$, defined in Cartesian space by the following mass-spring-damper equation of motion:

$$ {}^0\boldsymbol{w}_j = \boldsymbol{\Lambda}_{d_j}\Delta^0\ddot{\boldsymbol{x}}_j + \boldsymbol{D}_{d_j}\Delta^0\dot{\boldsymbol{x}}_j + \boldsymbol{K}_{d_j}\Delta^0\boldsymbol{x}_j \tag{116} $$

In this equation, ${}^0\boldsymbol{w}_j \in \mathbb{R}^m$ is the external wrench applied to the contact point j on the robot's kinematic structure and $\boldsymbol{\Lambda}_{d_j}, \boldsymbol{D}_{d_j}, \boldsymbol{K}_{d_j} \in \mathbb{S}_+^{m\times m}$ are the corresponding desired inertia, damping and stiffness matrices. These matrices define the contact dynamics. The term $\Delta^0\boldsymbol{x}_j = \left({}^0\boldsymbol{x}_{d_j} - {}^0\boldsymbol{x}_{ref_j}\right) \in \mathbb{R}^m$ and its time derivatives are respectively the position, velocity, and acceleration errors between the desired mass-spring-damper contact dynamics $({}^0\boldsymbol{x}_{d_j}, {}^0\dot{\boldsymbol{x}}_{d_j}, {}^0\ddot{\boldsymbol{x}}_{d_j})$ – which must be tracked by the robot – and the reference dynamics $({}^0\boldsymbol{x}_{ref_j}, {}^0\dot{\boldsymbol{x}}_{ref_j}, {}^0\ddot{\boldsymbol{x}}_{ref_j})$, towards which the robot must converge when no force is applied to it. When interaction is limited to a single contact point the resulting compliance control law can be derived as:

$$ \boldsymbol{\tau}_d = \boldsymbol{M}\ddot{\boldsymbol{q}}_d + \boldsymbol{h} \tag{117a} $$

$$ \ddot{\boldsymbol{q}}_d = \boldsymbol{J}_j^{\#}\left({}^0\ddot{\boldsymbol{x}}_{d_j} - \dot{\boldsymbol{J}}_j\dot{\boldsymbol{q}}\right) + \boldsymbol{N}_j\ddot{\boldsymbol{q}}_N \tag{117b} $$

$$ {}^0\ddot{\boldsymbol{x}}_{d_j} = {}^0\ddot{\boldsymbol{x}}_{ref_j} + \boldsymbol{\Lambda}_{d_j}^{-1}\left[{}^0\boldsymbol{w}_j - \boldsymbol{D}_{d_j}\Delta^0\dot{\boldsymbol{x}}_j - \boldsymbol{K}_{d_j}\Delta^0\boldsymbol{x}_j\right] \tag{117c} $$

where $\boldsymbol{J}_j^{\#} \in \mathbb{R}^{n\times m}$ denotes the generalized inverse of $\boldsymbol{J}_j$, $\boldsymbol{N}_j = \boldsymbol{1}_{n\times n} - \boldsymbol{J}_j^{\#}\boldsymbol{J}_j \in \mathbb{R}^{n\times n}$ is the corresponding nullspace projector, and $\ddot{\boldsymbol{q}}_N \in \mathbb{R}^n$ is an arbitrary joint acceleration vector to be projected on the nullspace of $\boldsymbol{J}_j$. This control technique was successfully implemented in [209], using the dynamic residual method [212, 213] for contact force estimation and exploiting visual feedback for contact point localization. Its generalization to scenarios involving simultaneous interactions with $\rho \in [1, n]$ different robot links within the same kinematic chain can be achieved using an augmented Jacobian representation, eventually leading to the following reformulation of eq. (117b):

$$\ddot{\boldsymbol{q}}_d = \boldsymbol{J}_{aug}^{\#} \left({}^0\ddot{\boldsymbol{x}}_{d_{aug}} - \dot{\boldsymbol{J}}_{aug}\dot{\boldsymbol{q}} \right) + \boldsymbol{N}_{aug}\ddot{\boldsymbol{q}}_N \tag{118}$$

where $\boldsymbol{J}_{aug} = \begin{bmatrix} \boldsymbol{J}_1^\top & \cdots & \boldsymbol{J}_\rho^\top \end{bmatrix}^\top \in \mathbb{R}^{\rho m \times n}$ concatenates the ρ Jacobian matrices of the reference interaction points, and ${}^0\ddot{\boldsymbol{x}}_{d_{aug}} = \begin{bmatrix} {}^0\ddot{\boldsymbol{x}}_{d_1}{}^\top & \cdots & {}^0\ddot{\boldsymbol{x}}_{d_\rho}{}^\top \end{bmatrix}^\top \in \mathbb{R}^{\rho m}$ contains the corresponding desired Cartesian accelerations. It is worth noting that in the case of more complex mechanisms such as in particular, floating base robots, the different skin task objectives have to be integrated into a stack-of-tasks architecture, similar to that presented in section 2.5. In this context, it should be highlighted that the forces applied to the different contact points between the robot and its environment can potentially be used by the robot for stabilization purposes via the force distribution algorithm presented in section 2.5.3 of this thesis, in equation (105a). The main drawback in the context of complex multi-contact interaction stems from the problem dimensionality as the optimization variable will have to stack the different contact points. This is here considered as interesting potential future work.

3.3. Whole-Body Tactile Compliance Under Constraints

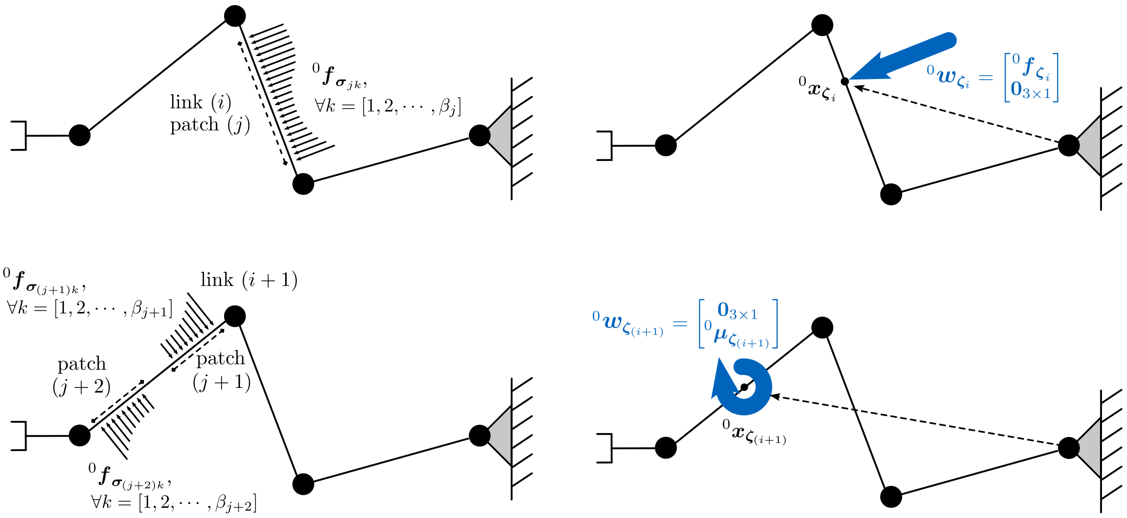
This section addresses the formulation of the whole-body tactile compliance problem. The treatment process of tactile information is presented, as well as the proposed method for generating control reference under constraints.

3.3.1. Assessing Tactile Interaction

When contacts with the external environment simultaneously occurs at multiple locations of a given robot link, the combined action of the resulting interaction forces must be adequately taken into account to generate suitable reactive motions according to eq. (118). In a contact-point compliance scenario, this requires a precise knowledge of both the forces applied to the robot and their localization on its kinematic chain. Since tactile feedback immediately provides information about the nature, location, and intensity of an arbitrary number of physical interactions, it is perfectly fitted for this task. In this work, we merge the artificial skin data into a single wrench vector, applied to the centroid of the considered link force distribution. The concept of force centroid can be thought of as a three-dimensional version of the zero moment point (ZMP) of a walking robot [214, 215], defined as the weighted average location $\boldsymbol{x}_{zmp}$ of the pressure efforts $\boldsymbol{p}(\boldsymbol{x}), \boldsymbol{x} \in \Omega$ applied to its feet:

$$\boldsymbol{x}_{zmp} = \frac{\displaystyle\int_\Omega \boldsymbol{x}\boldsymbol{p}(\boldsymbol{x})d\boldsymbol{x}}{\displaystyle\int_\Omega \boldsymbol{p}(\boldsymbol{x})d\boldsymbol{x}} \tag{119}$$

In our case, the location ${}^0\boldsymbol{x}_{\zeta_i}$ of the force centroid related with the rigid link i, is defined as a cell-wise discretized version of eq. (119). It is interesting to notice that the convex hull defined


link (i)
patch (j)
$^0\boldsymbol{f}_{\boldsymbol{\sigma}_{jk}}$, $\forall k = [1, 2, \cdots, \beta_j]$
$^0\boldsymbol{x}_{\zeta_i}$
$^0\boldsymbol{w}_{\zeta_i} = \begin{bmatrix} ^0\boldsymbol{f}_{\zeta_i} \\ \boldsymbol{0}_{3\times1} \end{bmatrix}$
$^0\boldsymbol{f}_{\boldsymbol{\sigma}_{(j+1)k}}$, $\forall k = [1, 2, \cdots, \beta_{j+1}]$
link $(i+1)$
patch $(j+2)$
patch $(j+1)$
$^0\boldsymbol{f}_{\boldsymbol{\sigma}_{(j+2)k}}$, $\forall k = [1, 2, \cdots, \beta_{j+2}]$
$^0\boldsymbol{w}_{\zeta_{(i+1)}} = \begin{bmatrix} \boldsymbol{0}_{3\times1} \\ ^0\boldsymbol{\mu}_{\zeta_{(i+1)}} \end{bmatrix}$
$^0\boldsymbol{x}_{\zeta_{(i+1)}}$

(i) Two examples of multi-contact interactions.

(ii) Resulting centroidal wrench corresponding to the presented multi-contact interactions.

Figure 12 Illustration of the equivalence between a given normal force distribution – measured by the artificial skin – and a single wrench applied to the force centroid of the considered rigid body.

by the surface of an artificial skin patch, and within which the force centroid is located, can be considered as the three-dimensional analogue to the support polygon of a walking robot. Let us consider the serial manipulator depicted in Fig. 12, whose rigid links i are covered by a number α_i of skin patches. Each patch $j = [1, 2, \cdots \alpha_i]$ of artificial skin contains a set of β_j *activated* skin cells, sending a stream of relevant force $^0\boldsymbol{f}_{\boldsymbol{\sigma}_{jk}} \in \mathbb{R}^3$ and position $^0\boldsymbol{x}_{\boldsymbol{\sigma}_{jk}} \in \mathbb{R}^3$ data. Then the set of all the $^0\boldsymbol{f}_{\boldsymbol{\sigma}_{jk}}$ applied to the corresponding contact points $^0\boldsymbol{x}_{\boldsymbol{\sigma}_{jk}}$ can be seen as equivalent to a unique wrench $^0\boldsymbol{w}_{\zeta_i} = \left[^0\boldsymbol{f}_{\zeta_i}^\top, {}^0\boldsymbol{\mu}_{\zeta_i}^\top \right]^\top$ applied to the force centroid $^0\boldsymbol{x}_{\zeta_i}$ of the rigid body i where

$$^0\boldsymbol{x}_{\zeta_i} = \frac{\displaystyle\sum_{j=1}^{\alpha_i}\sum_{k=1}^{\beta_j} \omega_{jk} \left\| ^0\boldsymbol{f}_{\boldsymbol{\sigma}_{jk}} \right\| {}^0\boldsymbol{x}_{\boldsymbol{\sigma}_{jk}}}{\displaystyle\sum_{j=1}^{\alpha_i}\sum_{k=1}^{\beta_j} \omega_{jk} \left\| ^0\boldsymbol{f}_{\boldsymbol{\sigma}_{jk}} \right\|} \tag{120a}$$

$$^0\boldsymbol{f}_{\zeta_i} = \sum_{j=1}^{\alpha_i}\sum_{k=1}^{\beta_j} \omega_{jk} {}^0\boldsymbol{f}_{\boldsymbol{\sigma}_{jk}} \tag{120b}$$

$$^0\boldsymbol{\mu}_{\zeta_i} = \sum_{j=1}^{\alpha_i}\sum_{k=1}^{\beta_j} \left(^0\boldsymbol{x}_{\boldsymbol{\sigma}_{jk}} - {}^0\boldsymbol{x}_{\zeta_i} \right) \times \left(\omega_{jk} {}^0\boldsymbol{f}_{\boldsymbol{\sigma}_{jk}} \right) \tag{120c}$$

3.3.2. Tactile Multiple-Contact Compliance

An important point to be noted in the last equations is the cell-wise weighting factor $\omega_{jk} \in \mathbb{R}_+^\star$, which allows implementing spatial variations in cell sensitivity within the same skin patch. This is a relevant feature of artificial skins and more generally of distributed sensor systems, which makes it possible to emulate the changes in sensory density observed on the biological skin. In this manner, the behavior of a robot can not only be adjusted limb-wise but also according to the location of interaction[2]. It is also worth mentioning that the associated computation cost overhead is in this case negligible. Furthermore, the possibility of adjusting ω_{jk} based on other sensory modalities of the robot skin, such as temperature or vibrations (accelerations), paves the way for the implementation of even more sophisticated behaviors and reflexes. Using this formulation, the centroidal interaction wrenches ${}^{0}\boldsymbol{w}_{\zeta_i}$ measured by the skin can be directly re-injected into eq. (117c) to generate a set of desired acceleration vectors ${}^{0}\ddot{\boldsymbol{x}}_{d_{\zeta_i}}$ (one for each link involved in the interaction process). An adequate torque command can then be issued using eq. (118) and eq. (117a). In this case the augmented variables are obtained by concatenating the corresponding centroidal quantities, for example $\boldsymbol{J}_{aug} = \begin{bmatrix} \boldsymbol{J}_{\zeta_1}^\top & \cdots & \boldsymbol{J}_{\zeta_\rho}^\top \end{bmatrix}^\top \in \mathbb{R}^{\rho m \times n}$ where $\boldsymbol{J}_{\zeta_i}$ is the Jacobian matrix associated with the centroid i and $\rho \in [1, n]$ is the total number of centroids. It is worth noting that a special attention must here be given to the computation of $\dot{\boldsymbol{J}}_{aug}$ as the evolution of $\boldsymbol{J}_{\zeta_i}$ does not only depends on the robot motions but also on the changes of centroid position in the considered rigid body. This link-wise formulation of the robot compliance control law is relevant since the computational complexity no longer depends on the number of activated skin cells as it was the case, for instance, in [204]. Yet in its current form, it does not account for the physical constraints of the considered robot.

3.3.3. Tactile Multi-Contact Compliance Under Constraints

Reformulating the issue of tactile multi-contact compliance as a quadratic optimization problem has many advantages. In particular, it allows to take into account the robot physical constraints directly within the control law, provided that they can be formulated as a set of affine inequalities. In practice, the compliance control law (117) can be thought of as finding the optimal control torque $\boldsymbol{\tau}^\star$ minimizing the acceleration residual ${}^{0}\ddot{\boldsymbol{x}}_{r_{\zeta_i}}^\top$, computed in eq. (3.4.1). From there it is straightforward to formulate the corresponding objective function as the regularized weighted squared $L2$-norm of the acceleration residual ${}^{0}\ddot{\boldsymbol{x}}_{r_{\zeta_i}}^\top$:

$$\boldsymbol{\tau}^\star = \underset{\boldsymbol{\tau}}{\arg\min} \quad \sum_{i=1}^{n} {}^{0}\ddot{\boldsymbol{x}}_{r_{\zeta_i}}^\top \boldsymbol{\Psi}_i \, {}^{0}\ddot{\boldsymbol{x}}_{r_{\zeta_i}} + \varepsilon \boldsymbol{\tau}^\top \boldsymbol{\tau} \tag{121}$$

In this equation, the variable $\varepsilon \in \mathbb{R}$ is a regularization term, allowing to avoid numerical singularities. This value is automatically set by the optimizer [216]. The weighting matrix $\boldsymbol{\Psi}_i = \operatorname{diag}(\psi_{ij}) \in \mathbb{R}_+^m$ plays an important role, by either enhancing or penalizing the influence of linear (respectively angular) components of the residual ${}^{0}\ddot{\boldsymbol{x}}_{r_{\zeta_i}}^\top$ within the global optimization process[3]. This is essential since the minimization of the linear acceleration residual is sometimes achieved at the expense of the angular acceleration residual. In this case, the

[2] Such a position-dependent weighting factor cannot be obtained using joint force-torque sensors. In this work, we set these weighting factors to 1.

[3] This is equivalent to the use of a weighted pseudo-inverse in eq. (118).

total wrench residual ${}^{0}\boldsymbol{w}_r$ of eq. (127) has to be reformulated as:

$$
{}^{0}\boldsymbol{w}_r = K \sum_{i=1}^{n} \begin{bmatrix} \overline{\boldsymbol{\Psi}}_i {}^{0}\boldsymbol{f}_{r_{\zeta_i}} \\ \underline{\boldsymbol{\Psi}}_i {}^{0}\boldsymbol{\mu}_{r_{\zeta_i}} + {}^{0}\boldsymbol{x}_{\zeta_i}(1:3) \times \overline{\boldsymbol{\Psi}}_i {}^{0}\boldsymbol{f}_{r_{\zeta_i}} \end{bmatrix} \tag{122}
$$

where (using Matlab notations) $\overline{\boldsymbol{\Psi}}_i = \boldsymbol{\Psi}_i(1:3, 1:3)$, $\underline{\boldsymbol{\Psi}}_i = \boldsymbol{\Psi}_i(4:6, 4:6)$, and $K \in \mathbb{R}_+$ is a tuning parameter.

3.3.4. Accounting for Robot Physical Limitations

As every mechanical system, robots are submitted to a set of kinematic or dynamic constraints, in particular, due to their geometry or actuation configuration. When a robot reaches one of its mechanical limits, the torque applied to the considered joints is automatically set to zero by internal safety checks, resulting in a temporary loss of one or several degrees of freedom. In practice, this is a risk to the external user since the controller is not aware that the robot can no longer move in a given direction of space. Nevertheless, maintaining compliance in this context is still possible, provided that the robot's constraints are properly taken into account in the main control loop. In this way, the forces applied in the vicinity of a mechanical limit can be identified and directly projected onto a parent limb (e.g. the mobile base), thereby generating suitable compliant reactive motions. Since most robots kinematic and dynamic constraints can be formulated as a set of inequalities, they can, therefore, be seamlessly taken into account in the previous QP formulation. As proposed in [217], we here consider a set of joint space constraints, namely torque, joint, and velocity limits:

$$
\begin{aligned}
-\boldsymbol{\tau}_u \quad &\leq \boldsymbol{\tau} \leq \quad \boldsymbol{\tau}_u && \text{(123a)} \\
\lambda\left(\boldsymbol{q}_{min} - \boldsymbol{q}\right) \quad &\leq \boldsymbol{\tau} \leq \quad \lambda\left(\boldsymbol{q}_{max} - \boldsymbol{q}\right) && \text{(123b)} \\
\mu\left(\dot{\boldsymbol{q}}_{min} - \dot{\boldsymbol{q}}\right) \quad &\leq \boldsymbol{\tau} \leq \quad \mu\left(\dot{\boldsymbol{q}}_{max} - \dot{\boldsymbol{q}}\right) && \text{(123c)}
\end{aligned}
$$

where $\lambda, \mu \in \mathbb{R}$ are convergence rates and where $\boldsymbol{q}_{min}$, $\boldsymbol{q}_{max}$ $\dot{\boldsymbol{q}}_{min}$ and $\dot{\boldsymbol{q}}_{max} \in \mathbb{R}^n$ are lower and upper joint position and velocity limits vectors for the robot's upper kinematic chains. Both convergence rates λ and μ were chosen heuristically. The QP for the constrained multi-contact compliance problem can eventually be formulated as:

$$
\begin{aligned}
\boldsymbol{\tau}^{\star} = \arg\min_{\boldsymbol{\tau}} \qquad & \sum_{i=1}^{n} {}^{0}\ddot{\boldsymbol{x}}_{r_{\zeta_i}}^{\top} \boldsymbol{\Psi}_i \, {}^{0}\ddot{\boldsymbol{x}}_{r_{\zeta_i}} + \varepsilon \boldsymbol{\tau}^{\top} \boldsymbol{\tau} \\
\text{s.t.} \qquad & -\boldsymbol{\tau}_u \leq \boldsymbol{\tau} \leq \boldsymbol{\tau}_u \\
& \lambda\left(\boldsymbol{q}_{min} - \boldsymbol{q}\right) \leq \boldsymbol{\tau} \leq \lambda\left(\boldsymbol{q}_{max} - \boldsymbol{q}\right) \\
& \mu\left(\dot{\boldsymbol{q}}_{min} - \dot{\boldsymbol{q}}\right) \leq \boldsymbol{\tau} \leq \mu\left(\dot{\boldsymbol{q}}_{max} - \dot{\boldsymbol{q}}\right)
\end{aligned} \tag{124}
$$

When one of the robot limbs approaches a mechanical limit, the torque generated by the external forces onto the considered joints is automatically bounded. The QP-solver then iteratively searches for the control torque that best matches the objective function while complying with these new constraints. In our approach, if the solver fails to find such a solution, the generalized forces applied to the considered limb will be directly propagated to a parent kinematic chain by setting $\boldsymbol{\tau}^{\star} = \mathbf{0}_{n\times 1}$ in eq. 3.4.1. It is important to notice that when interaction occurs between the robot and its environment, the rigid links that are involved in the interaction process have strict priority over those which are not in contact with the environment. In practice, this is achieved by setting the acceleration residual ${}^{0}\ddot{\boldsymbol{x}}_{r_{\zeta_i}}^{\top}$ of the considered links to zero in eq. (121) and (124).

3.4. Contact Residual Computation and Propagation

The motion components that cannot be properly executed by the upper limbs of a robot – referred to as *contact residuals* – can be identified and propagated onto its locomotor limbs, thereby increasing the compliance range. This section addresses the issues of contact residual computation and propagation.

3.4.1. Computation of the Contact Residuals

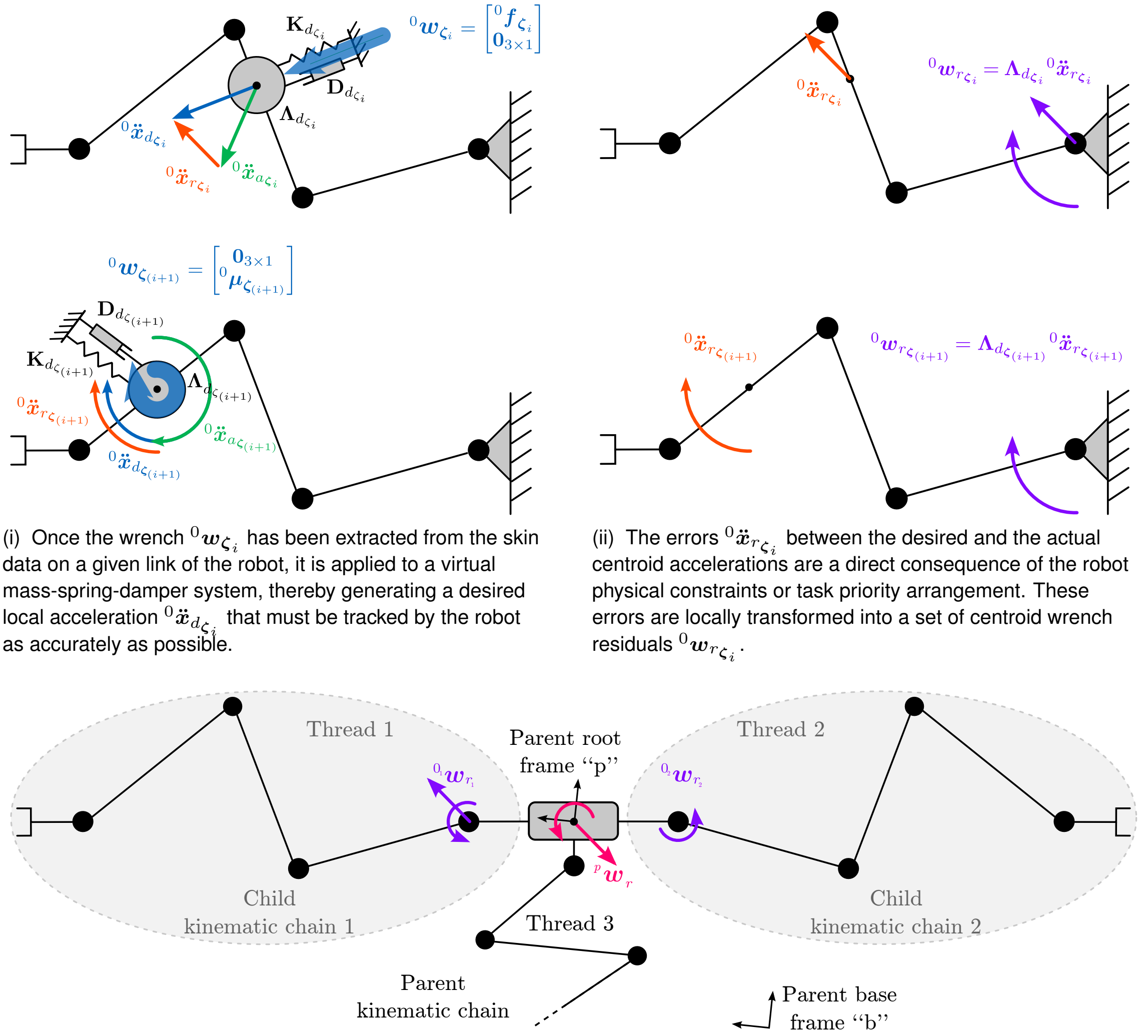
When an external wrench ${}^{0}\boldsymbol{w}_{\zeta_i}$ applied to a given robot limb, cannot produce the Cartesian acceleration ${}^{0}\ddot{\boldsymbol{x}}_{d_{\zeta_i}}$ required to match the desired contact impedance at the corresponding force centroid $\boldsymbol{\zeta}_i$, a local acceleration residual ${}^{0}\ddot{\boldsymbol{x}}_{r_{\zeta_i}}$ is generated. This residual can be computed as the difference between the desired acceleration ${}^{0}\ddot{\boldsymbol{x}}_{d_{\zeta_i}}$ of the mass-spring-damper system submitted to the centroid wrench ${}^{0}\boldsymbol{w}_{\zeta_i}$, and the acceleration ${}^{0}\ddot{\boldsymbol{x}}_{a_{\zeta_i}}$ the robot limb can locally generate without violating its physical limitations:

$$\begin{aligned} {}^{0}\ddot{\boldsymbol{x}}_{r_{\zeta_i}} &= {}^{0}\ddot{\boldsymbol{x}}_{d_{\zeta_i}} - {}^{0}\ddot{\boldsymbol{x}}_{a_{\zeta_i}} \\ &= {}^{0}\ddot{\boldsymbol{x}}_{d_{\zeta_i}} - \dot{\boldsymbol{J}}_{\zeta_i}\dot{\boldsymbol{q}} - \boldsymbol{J}_{\zeta_i}\boldsymbol{M}^{-1}\left[\boldsymbol{\tau}^{\star} - \boldsymbol{h}\right] \end{aligned} \tag{125}$$

One of the main challenges is here to correctly propagate these components onto the parent limbs in order to generate suitable reactive behaviors. One way to proceed would be to consider the residuals as local measures of the acceleration of a rigid body. Following the approach developed in [218], it would then be possible to express the total child limb residual acceleration with respect to a parent frame, as an optimization problem. However under these conditions, a proper solution could only be obtained provided that a set of – at least – four distinct acceleration residuals is available. If this condition is not verified, the problem may have an infinite number of solutions. To solve these issues, we decided to formulate the residual propagation problem at the force level and thus to convert the different ${}^{0}\ddot{\boldsymbol{x}}_{r_{\zeta_i}}$ into a set of wrench residuals ${}^{0}\boldsymbol{w}_{r_{\zeta_i}}$ using the desired dynamics $\boldsymbol{\Lambda}_{d_{\zeta_i}}$ of the considered link centroid $\boldsymbol{\zeta}_i$:

$${}^{0}\boldsymbol{w}_{r_{\zeta_i}} = \begin{bmatrix} {}^{0}\boldsymbol{f}_{r_{\zeta_i}} \\ {}^{0}\boldsymbol{\mu}_{r_{\zeta_i}} \end{bmatrix} = \boldsymbol{\Lambda}_{d_{\zeta_i}} {}^{0}\ddot{\boldsymbol{x}}_{r_{\zeta_i}} \tag{126}$$

These residuals contain the components of the centroid wrenches that were not adequately compensated on the considered limb. They have to be merged into a single limb wrench


$\mathbf{K}_{d_{\zeta_i}}$
${}^0\boldsymbol{w}_{\zeta_i} = \begin{bmatrix} {}^0\boldsymbol{f}_{\zeta_i} \\ \mathbf{0}_{3\times1} \end{bmatrix}$
$\mathbf{D}_{d_{\zeta_i}}$
$\boldsymbol{\Lambda}_{d_{\zeta_i}}$
${}^0\ddot{\boldsymbol{x}}_{d_{\zeta_i}}$
${}^0\ddot{\boldsymbol{x}}_{r_{\zeta_i}}$
${}^0\ddot{\boldsymbol{x}}_{a_{\zeta_i}}$
${}^0\ddot{\boldsymbol{x}}_{r_{\zeta_i}}$
${}^0\boldsymbol{w}_{r_{\zeta_i}} = \boldsymbol{\Lambda}_{d_{\zeta_i}}\,{}^0\ddot{\boldsymbol{x}}_{r_{\zeta_i}}$
${}^0\boldsymbol{w}_{\zeta_{(i+1)}} = \begin{bmatrix} \mathbf{0}_{3\times1} \\ {}^0\boldsymbol{\mu}_{\zeta_{(i+1)}} \end{bmatrix}$
$\mathbf{D}_{d_{\zeta_{(i+1)}}}$
$\mathbf{K}_{d_{\zeta_{(i+1)}}}$
$\boldsymbol{\Lambda}_{d_{\zeta_{(i+1)}}}$
${}^0\ddot{\boldsymbol{x}}_{r_{\zeta_{(i+1)}}}$
${}^0\ddot{\boldsymbol{x}}_{a_{\zeta_{(i+1)}}}$
${}^0\ddot{\boldsymbol{x}}_{d_{\zeta_{(i+1)}}}$
${}^0\ddot{\boldsymbol{x}}_{r_{\zeta_{(i+1)}}}$
${}^0\boldsymbol{w}_{r_{\zeta_{(i+1)}}} = \boldsymbol{\Lambda}_{d_{\zeta_{(i+1)}}}\,{}^0\ddot{\boldsymbol{x}}_{r_{\zeta_{(i+1)}}}$
Thread 1
Parent root
frame “p”
Thread 2
${}^{0_i}\boldsymbol{w}_{r_i}$
${}^{0_i}\boldsymbol{w}_{r_i}$
${}^p\boldsymbol{w}_r$
Child
kinematic chain 1
Thread 3
Child
kinematic chain 2
Parent
kinematic chain
Parent base
frame “b”

(i) Once the wrench ${}^0\boldsymbol{w}_{\zeta_i}$ has been extracted from the skin data on a given link of the robot, it is applied to a virtual mass-spring-damper system, thereby generating a desired local acceleration ${}^0\ddot{\boldsymbol{x}}_{d_{\zeta_i}}$ that must be tracked by the robot as accurately as possible.

(ii) The errors ${}^0\ddot{\boldsymbol{x}}_{r_{\zeta_i}}$ between the desired and the actual centroid accelerations are a direct consequence of the robot physical constraints or task priority arrangement. These errors are locally transformed into a set of centroid wrench residuals ${}^0\boldsymbol{w}_{r_{\zeta_i}}$.

(iii) The wrench residuals are then summed up link-wise to obtain the limb residual wrench ${}^{0_i}\boldsymbol{w}_{r_i}$. The resulting wrench residual ${}^p\boldsymbol{w}_{r_i}$ of each arm can eventually be propagated on the parent kinematic chain (in our case the robot base), in the same manner as an external wrench applied to “p”.

Figure 13 Illustration of tactile-based multi-contact compliance with generalized force propagation.

residual ${}^0\boldsymbol{w}_r$ applied to the parent-child junction:

$$ {}^0\boldsymbol{w}_r = \sum_{i=1}^{n} \begin{bmatrix} {}^0\boldsymbol{f}_{r_{\zeta_i}} \\ {}^0\boldsymbol{\mu}_{r_{\zeta_i}} + {}^0\boldsymbol{x}_{\zeta_i}(1:3) \times {}^0\boldsymbol{f}_{r_{\zeta_i}} \end{bmatrix} \tag{127} $$

3.4.2. Propagation of the Limb Total Wrench Residual

The next step is to propagate ${}^0\boldsymbol{w}_r$ onto the considered parent limb and therefore to express it on the corresponding basis. This can be achieved using the adjoint transform $\boldsymbol{Ad}_{{}^i\boldsymbol{H}_j} = \begin{bmatrix} {}^i\boldsymbol{R}_j & \mathbf{0}_{3\times3} \\ \left[{}^i\boldsymbol{x}_j(1{:}3)\right]_\times {}^i\boldsymbol{R}_j & {}^i\boldsymbol{R}_j \end{bmatrix}$, where ${}^i\boldsymbol{R}_j \in \boldsymbol{SO}(3)$ is the rotation matrix of frame j with respect to frame i, and $\left[{}^i\boldsymbol{x}_j(1:3)\right]_\times \in \mathbb{R}^{3\times3}$ is the skew operator associated with the first three components of the position vector ${}^i\boldsymbol{x}_j$: $\forall \boldsymbol{u} \in \mathbb{R}^3 : \left[{}^i\boldsymbol{x}_j(1:3)\right]_\times \boldsymbol{u} = {}^i\boldsymbol{x}_j(1:3) \times \boldsymbol{u}$. Since a given parent limb may have several child limbs sharing the same root frame (e.g. arms mounted on a rigid torso), the expression of the residual wrench applied to this frame must therefore be the sum of these limbs' contributions:

$$ {}^{p}\boldsymbol{w}_{r} = \sum_{c=1}^{\kappa} \boldsymbol{Ad}_{{}^{p}\boldsymbol{H}_{0_c}} {}^{0_c}\boldsymbol{w}_{r_c} \tag{128} $$

where κ is the number of child limbs attached to the same parent link (c.f. Fig.13iii). There may of course be several root links within a considered parent limb. The residual wrench ${}^{p}\boldsymbol{w}_{r}$ will therefore have to be calculated for each of these links before being classically re-injected into the parent limb controller as:

$$ \boldsymbol{\tau}^{\star} = \boldsymbol{M}_{p}(\boldsymbol{q})\ddot{\boldsymbol{q}}_{d} + \boldsymbol{h}_{p}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \tag{129a} $$

$$ \ddot{\boldsymbol{q}}_{d} = \boldsymbol{J}_{aug}^{\#}\left({}^{p}\ddot{\boldsymbol{x}}_{d_{aug}} - \dot{\boldsymbol{J}}_{aug}\dot{\boldsymbol{q}}\right) + \boldsymbol{N}_{aug}\boldsymbol{J}_{r_{aug}}^{\#}\left(\underbrace{\begin{bmatrix} \boldsymbol{\Lambda}_{d_1}^{-1}\ {}^{p}\boldsymbol{w}_{r_1} \\ \vdots \\ \boldsymbol{\Lambda}_{d_\chi}^{-1}\ {}^{p}\boldsymbol{w}_{r_\chi} \end{bmatrix}}_{\text{child limbs residual}} - \dot{\boldsymbol{J}}_{r_{aug}}\dot{\boldsymbol{q}}\right) \tag{129b} $$

where as previously $\boldsymbol{J}_{aug} = \begin{bmatrix} \boldsymbol{J}_{\boldsymbol{\zeta}_1}^{\top} & \cdots & \boldsymbol{J}_{\boldsymbol{\zeta}_\rho}^{\top} \end{bmatrix}^{\top} \in \mathbb{R}^{\rho m \times n}$ concatenates the $\rho \in [1, n]$ Jacobian matrices of the reference interaction points (this time in the parent limb) and ${}^{p}\ddot{\boldsymbol{x}}_{d_{aug}} = \begin{bmatrix} {}^{p}\ddot{\boldsymbol{x}}_{d_1}{}^{\top} & \cdots & {}^{p}\ddot{\boldsymbol{x}}_{d_\rho}{}^{\top} \end{bmatrix}^{\top} \in \mathbb{R}^{\rho m}$ contains the corresponding desired accelerations of the parent centroids. In this case, n denotes the dimension of the parent joint space. The residual is propagated in the nullspace of the parent limb interaction task, using the corresponding projector $\boldsymbol{P}_{aug} = \boldsymbol{I}_{n \times n} - \boldsymbol{J}_{aug}^{\#}\boldsymbol{J}_{aug} \in \mathbb{R}^{n \times n}$. The augmented Jacobian $\boldsymbol{J}_{r_{aug}} = \begin{bmatrix} \boldsymbol{J}_{r_1}^{\top} & \cdots & \boldsymbol{J}_{r_\chi}^{\top} \end{bmatrix}^{\top} \in \mathbb{R}^{\chi m \times n}$ concatenates the χ Jacobian matrices of the residuals, χ being the number of links – within the parent limb – that have at least one child limb with a non zero residual. In this case, $\boldsymbol{\Lambda}_{d_i} \in \mathbb{R}^{m \times m}$ denotes the desired inertia matrix of parent link i.

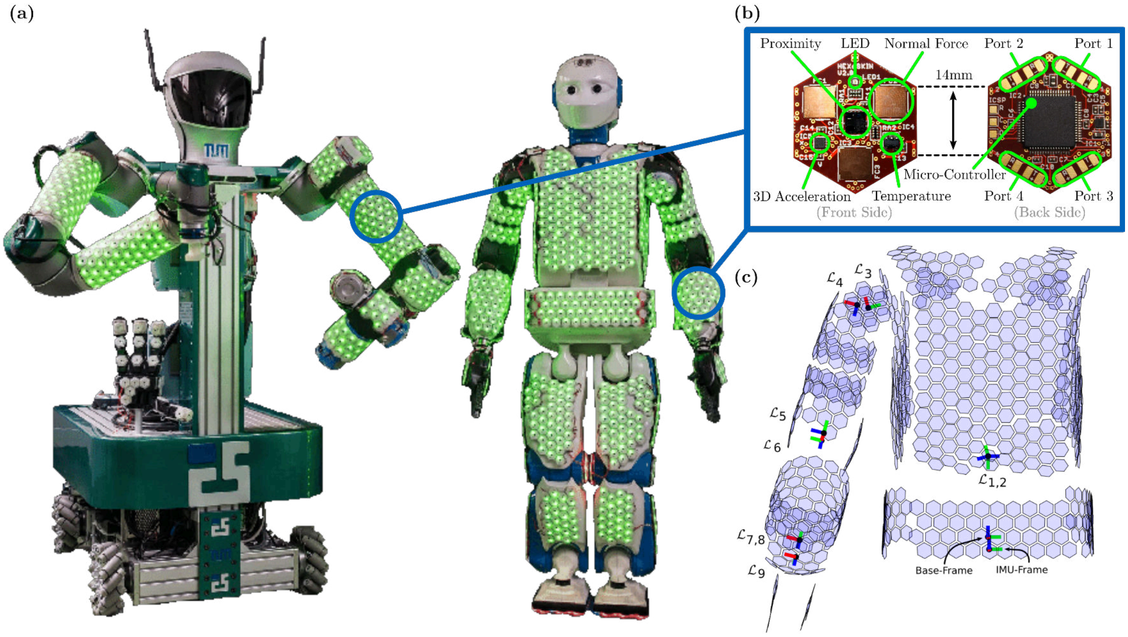


Figure 14 (a): Our experimental platforms, TOMM (*Tactile Omnidirectional Mobile Manipulator*) on the left and REEM-C (H1), on the right, are covered with multimodal artificial skin (Photo credit: A.Eckert/TUM.) **(b)**: The artificial skin mounted on our robot is a redundant network of identical measurement units or "cells" organized in patches. **(c)**: Partial shape reconstruction of the skin patches (here the torso and right arm) can be achieved after self-organization of the cells network, followed by a self-calibration step [202]. In this manner, it is possible to get the position and orientation of every skin cell with respect to a body-attached frame $\mathcal{L}_i$.

3.5. Experimental Validation

This section covers the details of the different experiments carried out to demonstrate the performance of the proposed method.

3.5.1. Description of the Robot Hardware and Control Interfaces

3.5.1.1 Description of the Robot Hardware and Control Interface

The robot platforms used during these experiments are presented in Fig. 14.a. Our first robot, named TOMM (*Tactile Omni-directional Mobile Manipulator*) [219] is a dual arm mobile manipulator, composed of two *Universal Robots UR5* industrial manipulators with 6 degrees of freedom each, covered with a multimodal artificial skin and mounted on a holonomic mobile base. TOMM's central unit is composed of two *intel Core i7* computers with 16Gb of RAM each. One of these CPU is fully dedicated to the control tasks. Our second robot is a REEM-C humanoid, H1, with 30 position controllable degrees of freedom (DoF). Its embedded computing unit also consists of two *intel Core i7* computers with respectively 8Gb and 16Gb of RAM, one of which is entirely dedicated to the real-time control tasks. Unlike TOMM, REEM-C is fully integrated to the ROS control [220] framework. On both TOMM and REEM-C, the embedded computers operate under a real-time patched Linux operating system and the complete control stack, including linear algebra and optimization libraries is coded in C++. TOMM's arms are velocity-controlled (*similarly to most industrial robots*) and provide a control interface capable of running at a frequency of 125Hz. The REEM-C control interface runs at

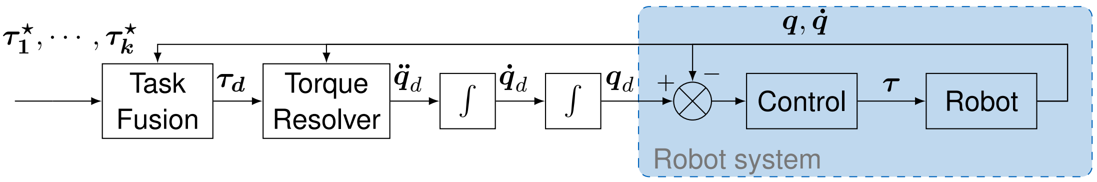


Figure 15 Block diagram of the torque resolver running on the robots, in a 500Hz real-time thread on TOMM and in a 1KHz real-time thread on REEM-C.

a steady frequency of 200Hz. On both robots, the different tasks are fused together at the torque level, using the recursive nullspace projection method presented in (90). As depicted in figure 15, resolved torque control is here made possible by means of an internal torque resolver loop, running on a dedicated thread at a frequency of 1kHz on REEM-C and 500Hz on TOMM [177]. A specific admittance controller – similar to [178] – was also developed for TOMM's mobile base, in order to make it force/torque-controllable, thereby allowing force propagation from the arms to the mobile base. The virtual dynamics of the mobile base is described by the following three-dimensional equation of motion:

$$\boldsymbol{\mu}_b \dot{\boldsymbol{v}}_b + \boldsymbol{D}_b \boldsymbol{v}_b = \boldsymbol{S}_b {}^p\boldsymbol{w}_r. \tag{130}$$

where $\dot{\boldsymbol{v}}_b = \boldsymbol{S}_b \ddot{\boldsymbol{x}} \in \mathbb{R}^3$ concatenates the first two linear components and the last angular component of a Cartesian acceleration and $\boldsymbol{S}_b = \begin{bmatrix} 1 & 0 & & 0 \\ 0 & 1 & \mathbf{0}_{3\times3} & 0 \\ 0 & 0 & & 1 \end{bmatrix}$ is the corresponding selection matrix. The parameters $\boldsymbol{\mu}_b = \mathrm{diag}(m, m, I) \in \mathbb{R}^{3\times3}$ and $\boldsymbol{D}_b = \mathrm{diag}(d_1, d_2, d_3) \in \mathbb{R}^{3\times3}$ represent the virtual inertia and damping of the admittance controller, where $m \in \mathbb{R}_+$ is the mobile base virtual mass, $I \in \mathbb{R}_+$ is its virtual moment of inertia and $d_1, d_2, d_3 \in \mathbb{R}_+$ are its viscous friction coefficients. During the experiments, the virtual damping values were set to $d_1 = d_2 = 40N.s.m^{-1}$ and $d_3 = 40N.m.s.rad^{-1}$. These values were chosen heuristically. When the wrench ${}^p\boldsymbol{w}_r$ is applied to the mobile base, it produces a reference Cartesian acceleration $\dot{\boldsymbol{v}}_b \in \mathbb{R}^3$. This acceleration is then integrated and sent as a bounded velocity reference to the mobile-base velocity controller each $8ms$[4].

3.5.1.2 Assumptions

Similar to the "floating base" framework of humanoid robots, and disregarding the issues of tilting and instability, we model TOMM's omnidirectional mobile base coupled to its low level velocity controller, as a 3-DoF robot made of two orthogonal prismatic joints moving in the horizontal plane and one revolute joint, rotating along the vertical axis. We moreover consider the mobile base and the upper limbs as separated systems. The arm-arm and arm-base dynamic couplings are especially not taken into account. Although this assumption is constraining, it nevertheless remains valid in our case since the torso is not articulated, the mass of the mobile base is one order of magnitude higher than that of the arms, that the omnidirectional wheels are not mounted on shock absorbers and that they also generate significant friction on the ground. Moreover, since the arms of the robot are industrial grade

[4] The internal velocity control loop of the mobile base operates at $1kHz$ on its own real-time thread.

manipulators, we assume that the disturbances generated by the mobile base accelerations are negligible. The fully coupled system is to be considered in future works.

3.5.1.3 Details of the QP formulation

We use the *qpOASES* C++ library [221] as QP solver since it provides useful regularization options as well as suitable timing tools, allowing implementation in real-time control loops. On a more formal point of view, *qpOASES* requires the problem to be formulated using box constraints, which is very well suited to our implementation:

$$\min_{\boldsymbol{z}} \quad \frac{1}{2}\boldsymbol{z}^\top \boldsymbol{H}\boldsymbol{z} + \boldsymbol{z}^\top \boldsymbol{b} \tag{131a}$$

$$\text{s.t.} \quad \boldsymbol{lb} \leq \boldsymbol{z} \leq \boldsymbol{ub} \tag{131b}$$

In this expression, the matrix $\boldsymbol{H} \in \mathbb{R}^{\vartheta\times\vartheta}$ denotes the Hessian, and the vectors $\boldsymbol{b}, \boldsymbol{z}, \boldsymbol{lb}, \boldsymbol{ub} \in \mathbb{R}^{\vartheta}$ respectively denotes the gradient, the optimization variable, and its corresponding lower and upper bounds. Here $\vartheta \in [0, 6]$ is the dimension of the problem. Note that in this case, equality constraints can still be handled by simply putting the same expression on both sides of the inequality in eq. (131b). In our case[5] the QP (124) can be formulated as:

$$\boldsymbol{z} = \boldsymbol{S}\boldsymbol{\tau} \tag{132a}$$

$$\boldsymbol{H} = \boldsymbol{S}\left[\sum_{i=1}^{n}\left(\boldsymbol{J}_{\zeta_i}\boldsymbol{M}^{-1}\right)^\top \boldsymbol{\Psi}_i \boldsymbol{J}_{\zeta_i}\boldsymbol{M}^{-1}\right]\boldsymbol{S}^\top \tag{132b}$$

$$\boldsymbol{g} = -\boldsymbol{S}\sum_{i=1}^{n}\left(\boldsymbol{J}_{\zeta_i}\boldsymbol{M}^{-1}\right)^\top \boldsymbol{\Psi}_i\left[{}^{0}\ddot{\boldsymbol{x}}_{d_{\zeta_i}} - \dot{\boldsymbol{J}}_{\zeta_i}\dot{\boldsymbol{q}} + \boldsymbol{J}_{\zeta_i}\boldsymbol{M}^{-1}\boldsymbol{h}\right] \tag{132c}$$

$$\boldsymbol{ub} = \boldsymbol{S}\min(\boldsymbol{\tau}_{max}, \lambda\left(\boldsymbol{q}_{max} - \boldsymbol{q}\right), \mu\left(\dot{\boldsymbol{q}}_{max} - \dot{\boldsymbol{q}}\right)) \tag{132d}$$

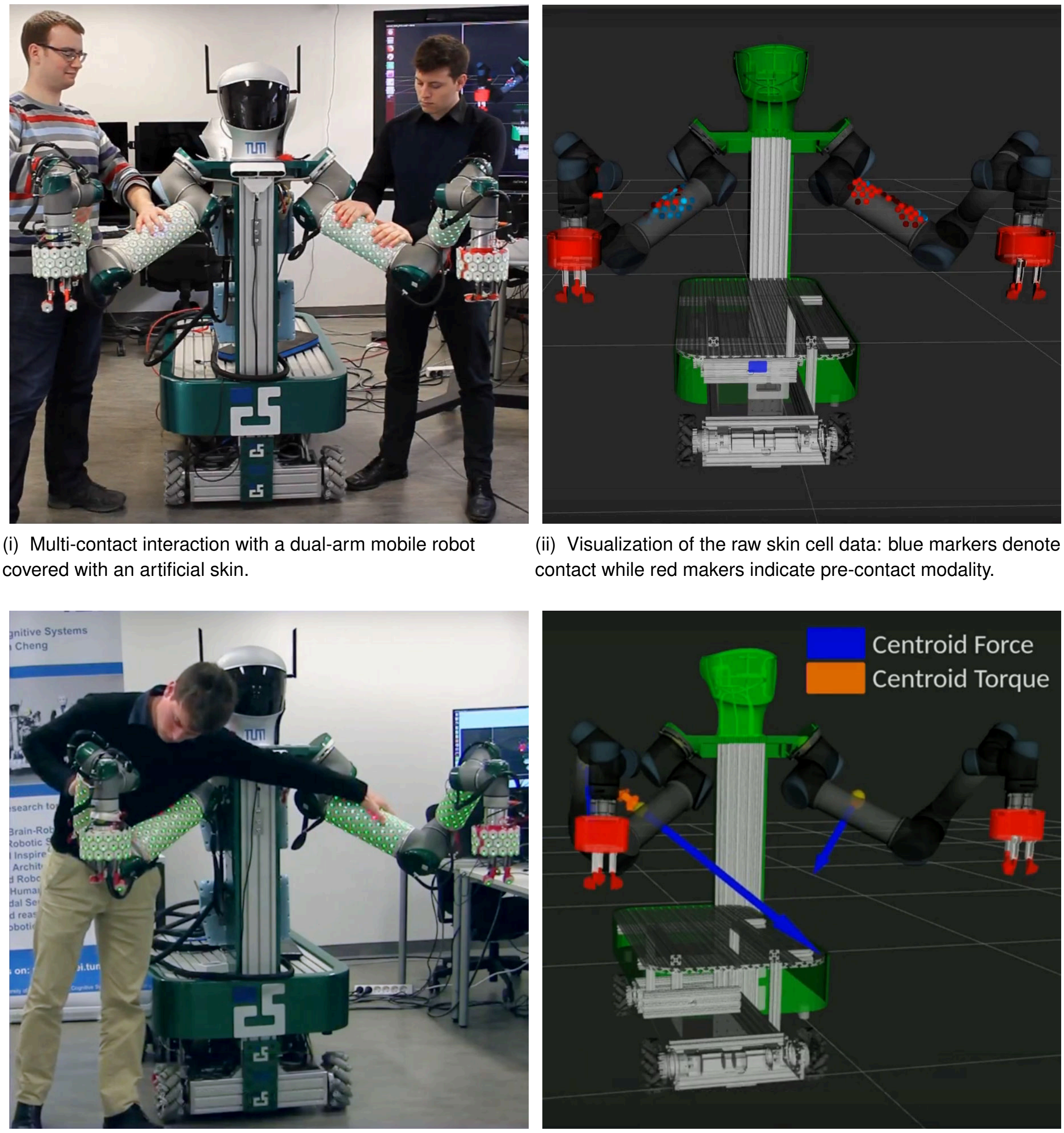
$$\boldsymbol{lb} = \boldsymbol{S}\max(-\boldsymbol{\tau}_{max}, \lambda\left(\boldsymbol{q}_{min} - \boldsymbol{q}\right), \mu\left(\dot{\boldsymbol{q}}_{min} - \dot{\boldsymbol{q}}\right)) \tag{132e}$$

where the $\min$ and $\max$ operators are defined element-wise and where $\boldsymbol{S} = \left[\mathbf{1}_{\vartheta\times\vartheta}, \mathbf{0}_{\vartheta\times(n-\vartheta)}\right] \in \mathbb{R}^{\vartheta\times n}$ is a selection matrix allowing to discard the influence of unused joints within the QP, where $\mathbf{1}_{\vartheta\times\vartheta}$ is the $\vartheta \times \vartheta$ identity matrix. For a given kinematic chain, the dimension ϑ of the QP corresponds to the index of the highest-order joint involved in the interaction process. For instance, a single-contact interaction with the fourth rigid link of the robot (moving with the third joint), will result in a three-dimensional problem.

3.5.2. Description of the Artificial Skin Setup

On TOMM, a set of 4 distinct Tactile Section Units (TSUs) is broadcasting the data from 14 skin patches, for a total of *644 skin cells* covering the two arms. Depending on how "far" a skin cell is from the TSU – *i.e. how many skin cells have to be crossed before reaching the TSU* – it takes about $1ms$ to propagate the information to the main CPU. A new command is then issued by the controller every $8ms$ using the last information provided by the skin (note that the control frequency is here determined by the real-time interface used to communicate with

[5] For the sake of compactness, the $\boldsymbol{q}, \dot{\boldsymbol{q}}$ dependencies will be removed in the following equations.



(i) Multi-contact interaction with a dual-arm mobile robot covered with an artificial skin.

(ii) Visualization of the raw skin cell data: blue markers denote contact while red makers indicate pre-contact modality.

(iii) Multi-contact interaction with a dual-arm mobile robot covered with an artificial skin.

(iv) Link wrenches computed at the contact centroid for data reduction purposes.

Figure 16 Skin raw contact forces and centroidal wrenches

each UR5 robot arm)[6]. The skin setup of the REEM-C robot is more complex and consists of a total of 1260 cells, divided into 47 patches covering the whole robot body, and connected to the main computer with a Gigabit Ethernet interface, through a set of TSUs. To be fully integrated into a real-time control loop, it is essential that the skin data is processed efficiently, to avoid excessive overhead. One method to achieve this is not to formulate the tactile control problem at the point of contact (c.f. Fig. 16i and Fig. 16ii) but rather at the contact centroid. This allows to keep the essential information related to the interaction while maintaining a low computational effort (c.f. Fig. 16iii and Fig. 16iv).

[6] It is here worth comparing this result to human tactile reflexes, such as the patellar reflex, which is issued in about $29.6 \pm 6.0ms$ [222].

3.5.3. Description of the Experiments

3.5.3.1 Summary

A set of six different experiments was performed to demonstrate the main features of the proposed approach. The first experiment aims at illustrating the ability of our control algorithm to comply with a set of joint position constraints. The second experiment aims at highlighting the possibility of generating compliant reactive behaviors with an adjustable dynamics, in reaction to multi-contact physical interactions under torque constraints[7]. The third and fourth experiments study the behavior of the proposed control algorithm in the specific cases of fully propagated and nullspace interactions respectively. The fifth experiment compares the computational efficiency of the proposed method with that of a previous approach [204] formulating the compliance problem at the cell level rather than at the centroid level. Finally the sixth experiment proves that our method can also run on floating base robots and provide enhanced compliance by generating suitable walking motions in reaction to whole-body tactile physical interaction.

3.5.3.2 Experiment protocols

The experiments follow the same protocol: forces are applied to different locations of the robots. To have an effect on the robot, it is essential that these forces are applied to a surface which is covered by the artificial skin. In the current configuration, any force applied on an uncovered surface of the robot will *not* be detected. These forces are then processed according to eq. (120) in order to compute the position ${}^{0}\boldsymbol{x}_{\zeta_i}$ of the link force-centroid and the associated interaction wrench ${}^{0}\boldsymbol{w}_{\zeta_i}$. The next step is to generate a command $\boldsymbol{\tau}^{\star}$ that will satisfy as much as possible the desired reactive behaviors – defined by the mass-spring-damping parameters assigned to each centroid – while at the same time complying with the robot physical constraints. This is achieved by solving the QP of eq. (124). The Cartesian acceleration residuals are then computed for each arm according to eq. (3.4.1).

Once transformed into wrench residuals by injecting them into eq. (126), they are propagated onto the mobile base using eq. (127), (128) and eq. (130). It is important to note that single and dual arm interactions are treated exactly in the same manner since each arm has its own controller instance, running in a specific thread, as shown in Fig.13iii. In the case of dual arm interaction, the wrench residuals computed on the base frame of each arm, are simply summed up on the parent root frame (denoted as “p” on Fig.13iii) according to eq. (128). The resulting wrench is then propagated to the mobile base.

[7] It is important to note that although this experiment specifically considers the case of high stiffness reactive dynamics, a wide variety of different dynamic responses can also be generated. Some of these behaviors can be observed within the experiments 1 (zero stiffness) and 3-4 (medium stiffness) for which different values of the contact mass-spring-damper constants were investigated as well as different mobile base dynamic parameters.

3.5.4. Results and Discussion

3.5.4.1 Experiment 1. Robot joint limit avoidance

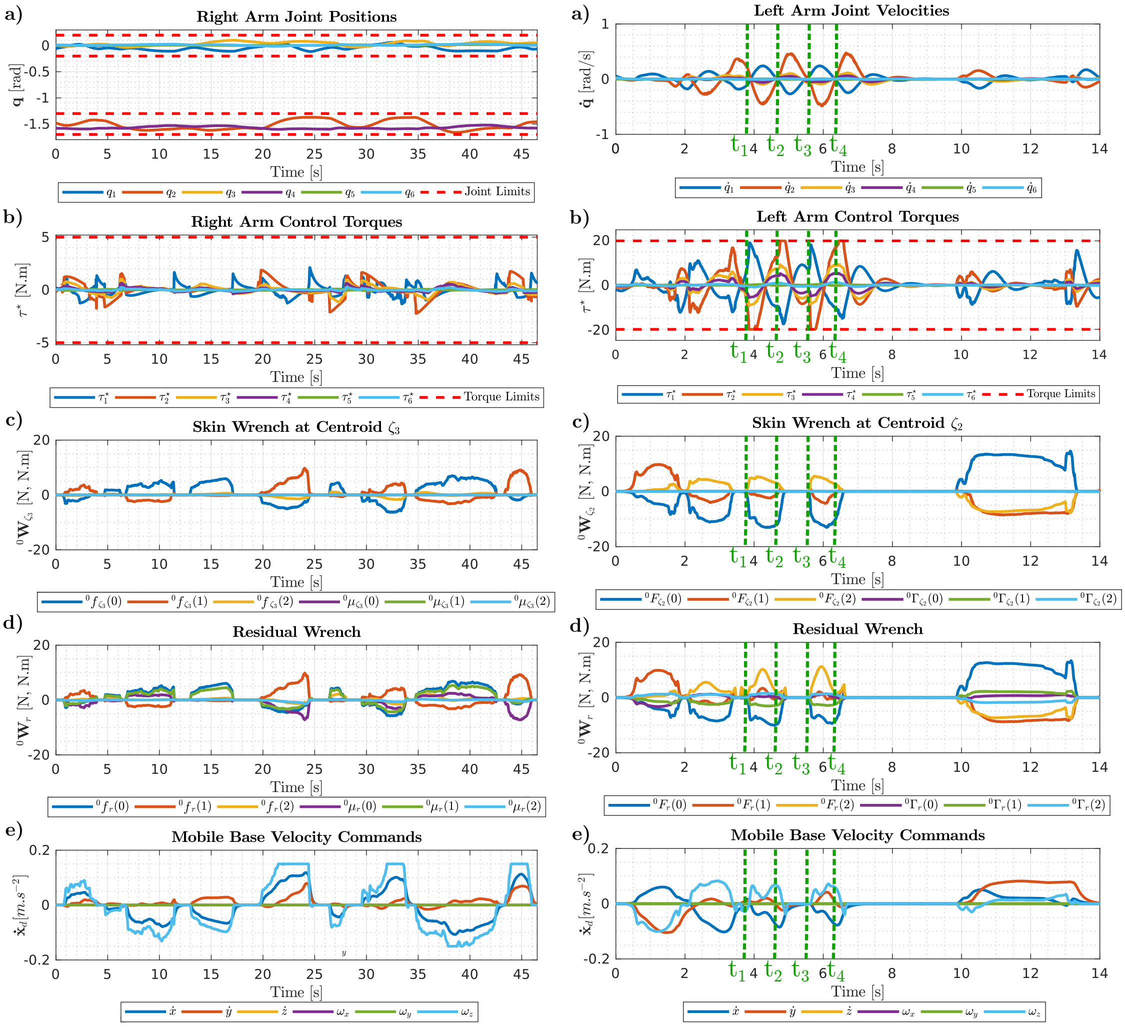
The objective of this first experiment is to demonstrate that the proposed QP formulation of the compliance control law allows a convenient handling of the robot physical constraints and in particular of joint limits. The experiment itself consists in interacting with the robot in different contact points. Moving the right arm of TOMM to its home position ($\boldsymbol{q} = [0, \pi/2, 0, \pi/2, 0, 0]^\top$), we define the following constrains:

- Torque constraints: $-\boldsymbol{\tau}_{max} \leq \boldsymbol{\tau} \leq \boldsymbol{\tau}_{max}$ where $\boldsymbol{\tau}_u = [5, 5, 5, 5, 5, 5]^\top\,[\mathrm{N.\,m}]$
- Joint limits: $\lambda\left(\boldsymbol{q}_{min} - \boldsymbol{q}\right) \leq \boldsymbol{\tau}^\star \leq \lambda\left(\boldsymbol{q}_{max} - \boldsymbol{q}\right)$ where the convergence rates are set to $\lambda = \mu = 7$ and where $\boldsymbol{q}_{min} = \left[-0.2, \frac{\pi}{2} - 0.2, -0.2, \frac{\pi}{2} - 0.2, -0.2, -0.2\right]^\top$ $[\mathrm{rad}]$ and $\boldsymbol{q}_{max} = \left[0.2, \frac{\pi}{2} + 0.2, 0.2, \frac{\pi}{2} + 0.2, 0.2, 0.2\right]^\top$ $[\mathrm{rad}]$.

In this experiment, the desired centroid dynamic parameters are as follows: $\boldsymbol{\Lambda}_{d_{\zeta_3}} = 2.5 \cdot \boldsymbol{I}_{6\times6}$, $\boldsymbol{D}_{d_{\zeta_3}} = 3 \cdot \boldsymbol{I}_{6\times6}$ and $\boldsymbol{K}_{d_{\zeta_3}} = \boldsymbol{0}_{6\times6}$. The mobile base virtual mass is set to $m = 10kg$ and its virtual inertia to $I = 1.0kg.m^2$. The results, shown in Fig. 17i depict that the previously defined constraints are respected: the considered interaction wrench ${}^0\boldsymbol{w}_{\zeta_3}$ located in a centroid ζ_3 (Fig. 17i.b) produces bounded position (Fig. 17i.a) response. As shown in Figs. 17i.c-d the wrench components that would have led to a violation of these constraints are propagated to the mobile base, thereby generating suitable reactive motions. In practice, since the bounds are tight, most of the applied wrench is directly propagated to the mobile base. In the following experiments, we set $\lambda = \mu = 100$ as the robot is moving far from its joint limits.

3.5.4.2 Experiment 2. Contact impedance control with torque limits

The following experiment is focusing on the generation of compliant reactive motions following a desired dynamics, in response to physical interactions between the robot and an external operator. A set of external forces is applied to the second link of TOMM's left arm. This results in an interaction wrench ${}^0\boldsymbol{w}_{\zeta_2}$ located in a centroid $\boldsymbol{\zeta}_2$ on the second link of the robot (c.f. Fig. 17ii.c). The QP solver generates a set of control torques (c.f. Fig. 17ii.b) that aims to provide the considered centroid with the desired reactive compliant behavior while respecting a set of predefined robot constraints. The desired dynamic parameters of the centroid are the following: $\boldsymbol{\Lambda}_{d_{\zeta_3}} = 2.5 \cdot \boldsymbol{I}_{6\times6}$, $\boldsymbol{D}_{d_{\zeta_3}} = 3 \cdot \boldsymbol{I}_{6\times6}$ and $\boldsymbol{K}_{d_{\zeta_3}} = 50 \cdot \boldsymbol{I}_{6\times6}$. The mobile base virtual mass is set to $m = 25kg$ and its virtual inertia to $I = 1.0kg.m^2$. We notice in Fig. 17ii.a that the joint velocities converge to zero, (e.g. for $10 \leq t \leq 12$) while the force is still applied to the robot arm (c.f. Fig. 17ii.c). This is the expected behavior from a mass-spring damper system with a high stiffness value. Notice on Fig.17ii.b, the torque saturations occurring at t_1, t_2, t_3 and t_4. These saturations are the result of the torque constraints defined in the QP. In our case, the torque limits are set to $\pm 20N.m$ for each joint. As the torque reaches its limit, the considered robot limb cannot entirely propagate the external forces. The resulting wrench residual (c.f. Fig. 18i.d) must therefore be propagated to the mobile base as shown in Fig.18i.e.



(i) Robot joint limit avoidance

(ii) Contact impedance control

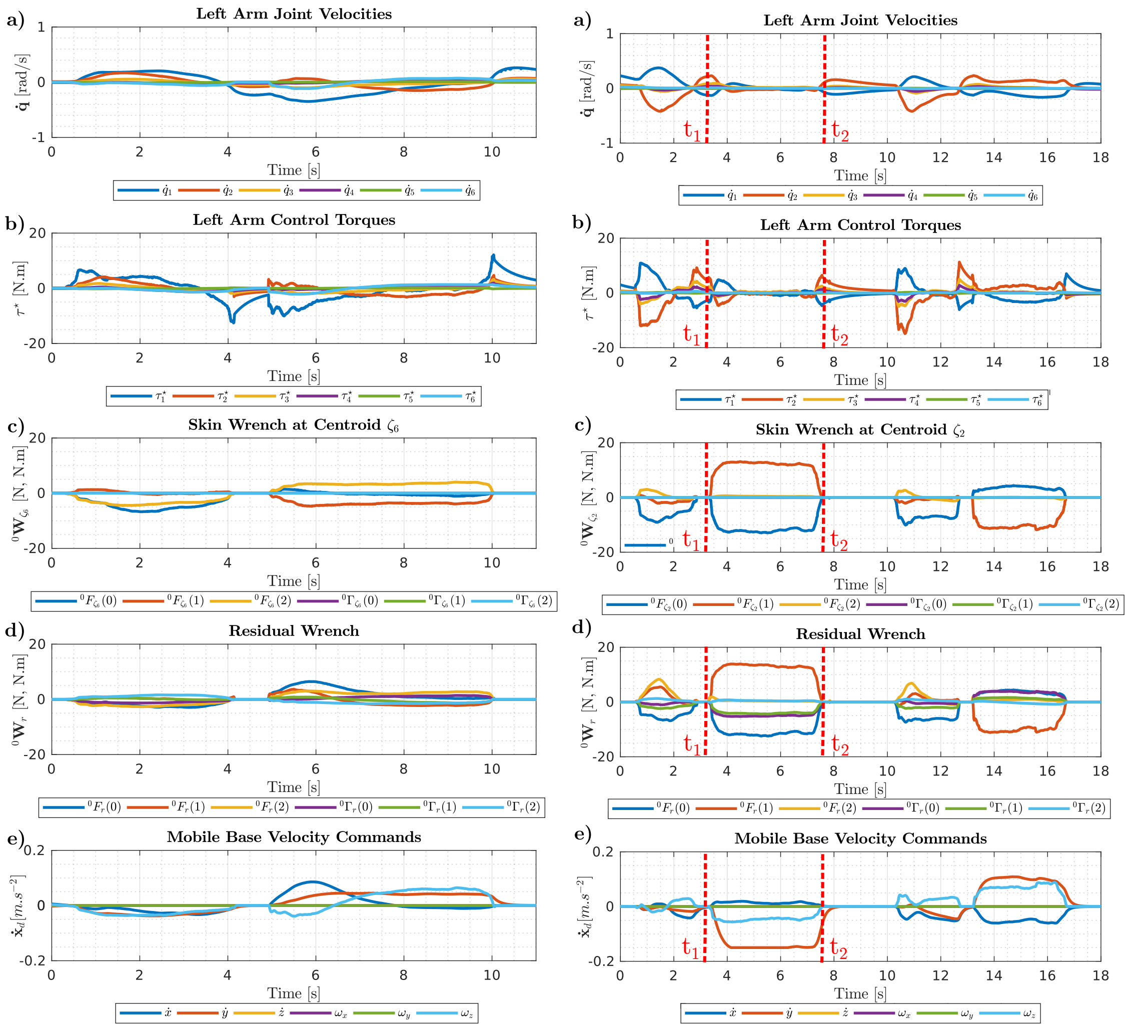
Figure 17 Joint constraint experiment.

3.5.4.3 Experiment 3. Fully propagated interactions

In this experiment, a set of forces is applied by an external operator, to the robot end-effector, thereby resulting in a wrench ${}^{0}\boldsymbol{w}_{\zeta_6}$ located in the centroid $\boldsymbol{\zeta}_6$ (c.f. Fig. 18i.c). The desired centroid dynamic parameters are set to the following values: $\boldsymbol{\Lambda}_{d_{\zeta_3}} = 2.5 \cdot \boldsymbol{I}_{6\times6}$, $\boldsymbol{D}_{d_{\zeta_3}} = 3 \cdot \boldsymbol{I}_{6\times6}$ and $\boldsymbol{K}_{d_{\zeta_3}} = 5 \cdot \boldsymbol{I}_{6\times6}$. The mobile base virtual mass is now set to $m = 20kg$ and its virtual inertia to $I = 1.0kg.m^2$ to make it easier to move.

The objective of this experiment is to demonstrate that a force which is properly compensated by a given limb[8], does not generate any significant residual onto its parent (in our case, on the mobile base). The experimental results, depicted in Fig. 18i, show that this is indeed the case: the residual wrench – resp. mobile base motions – are negligible on the considered time frame, as shown in Fig.18i.d-e.

[8] Hence the choice of the end-effector for this experiment.



(i) Fully propagated interactions

(ii) Nullspace interactions

Figure 18 Fully propagated vs. nullspace interactions.

3.5.4.4 Experiment 4. Nullspace interactions

In this experiment, a force is applied close to the nullspace of the second link, as depicted in Fig. 20. The desired dynamic parameters are the same as those of the previous experiment: $\boldsymbol{\Lambda}_{d_{\zeta_3}} = 2.5 \cdot \boldsymbol{I}_{6\times6}$, $\boldsymbol{D}_{d_{\zeta_3}} = 3 \cdot \boldsymbol{I}_{6\times6}$ and $\boldsymbol{K}_{d_{\zeta_3}} = 5 \cdot \boldsymbol{I}_{6\times6}$. The mobile base virtual mass is here also set to $m = 20kg$ and its virtual inertia to $I = 1.0kg.m^2$. On a theoretical point of view, since the considered limb cannot generate any compliant reactive movement in the direction of the applied force, this force should therefore be entirely propagated to the mobile base.

The experiment results show this behavior for $t_1 \leq t \leq t_2$. In this time interval, we observe in Fig.18ii.c-e that the applied wrench is entirely transmitted to the mobile base, generating suitable reactive motion, while the arm velocities and torques remain close to zero[9] (c.f. Fig.18ii.a-b) since the arm cannot generate suitable accelerations in the direction of the applied force. Notice on Fig.18ii.e that for $t \geq t_2$, the mobile base velocity command is not

[9] Joint velocities and torques are nonetheless not null, since the external forces are not applied “exactly” to the nullspace of the robot.

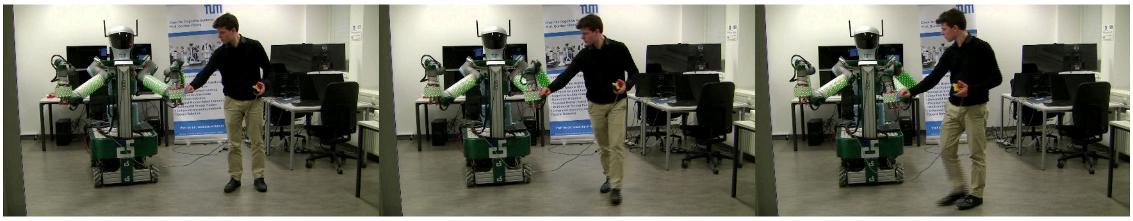


Figure 19 Example of a fully propagated interaction

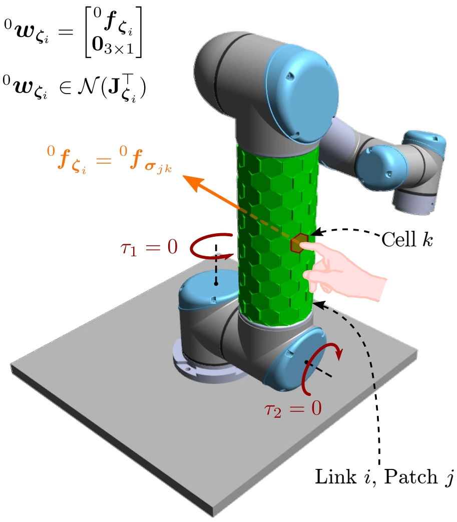

$^{0}\boldsymbol{w}_{\zeta_i} = \begin{bmatrix} {}^{0}\boldsymbol{f}_{\zeta_i} \\ \mathbf{0}_{3\times 1} \end{bmatrix}$
$^{0}\boldsymbol{w}_{\zeta_i} \in \mathcal{N}(\mathbf{J}_{\zeta_i}^{\top})$
$^{0}\boldsymbol{f}_{\zeta_i} = {}^{0}\boldsymbol{f}_{\sigma_{jk}}$
$\tau_1 = 0$
Cell k
$\tau_2 = 0$
Link i, Patch j

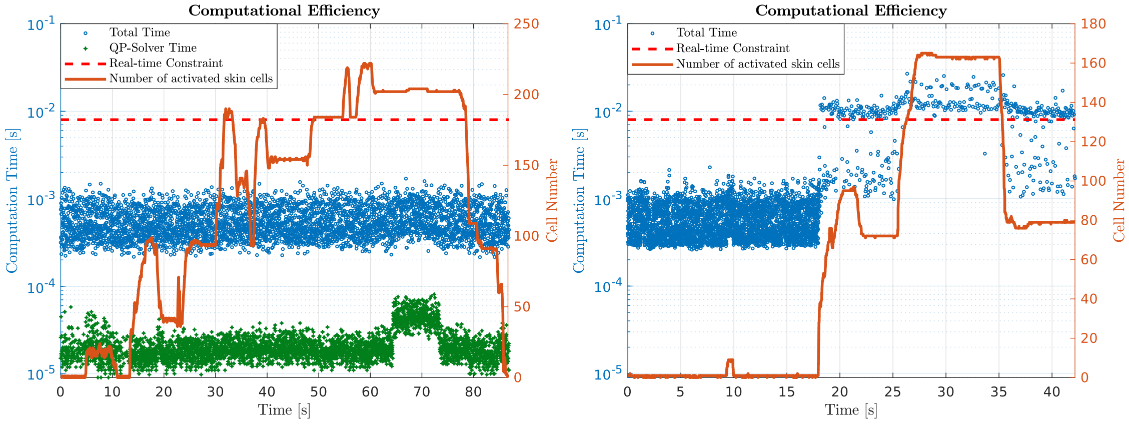
Figure 20 Single-contact nullspace interaction with the second rigid link of a UR5 robot. Since the wrench $^{0}\boldsymbol{w}_{\zeta_i}$ lies in the nullspace $\mathcal{N}(\boldsymbol{J}_{\zeta_i}^{\top})$ of the corresponding contact Jacobian transpose, it cannot produce any torque on the robot kinematic chain, and must therefore be propagated to a parent limb.

immediately set to zero, even if the external wrench is no longer applied to the robot. This is a consequence of the mobile base virtual dynamics.

3.5.4.5 Experiment 5. Computational Efficiency

During this experiment, we focus on the evolution of our controller's computational complexity as a function of the number of activated skin cells. For both practical and safety reasons, the robot must remain motionless during the whole process. Therefore although the control torque $\boldsymbol{\tau}^{\star}$ is still computed for timing evaluation purpose, we rather control the real robot with a simple gravity compensation controller $\boldsymbol{\tau} = \boldsymbol{g}(\boldsymbol{q})$, resulting in $\ddot{\boldsymbol{q}} = \mathbf{0}_{n\times 1}$. We then proceed by gradually covering the robot's artificial skin with a set of opaque screens, in order to activate as many proximity sensors as possible. Since the skin's proximity feedback is automatically mapped to a virtual force [177], it is therefore seamlessly integrated into our controller.

The experiment results are shown in Fig. 21i and Fig. 21ii. We can observe that the global execution time of the proposed algorithm – in Fig. 21i – fully comply with the real-time constraint of our control loop ($\delta t = 8ms$ in dashed red). The results moreover suggest that the influence of the number of activated cells on the computation time of the loop is negligible. Unlike our previous work, where a QP had to be solved for every activated skin cell, the



(i) Proposed method

(ii) Previous framework [204]

Figure 21 Computational Efficiency. In green the time t_{qp} required to solve the QP (on a semi-log scale). In blue on the same scale, the time t_{tot} necessary to solve the entire control problem (i.e. acquire skin data, and solve the QP). The real-time loop constraint $t_{loop} = 8ms$ is showed in dashed red. Finally the number of activated cells, sending force and position data is showed in Orange.

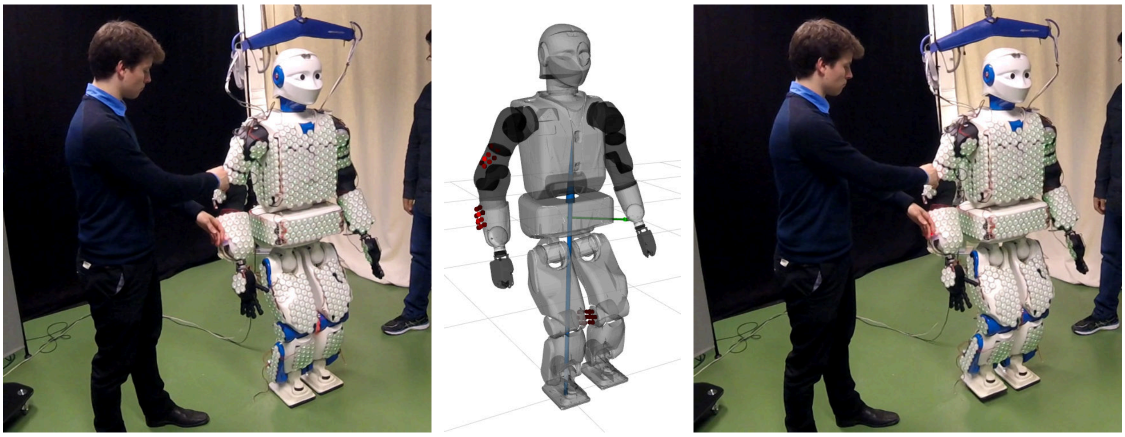
proposed control method now only requires a single QP to be solved *for each limb and for each time iteration of the control loop*. In the worst case, for TOMM, it is therefore only two QP that will have to be solved at each control iteration[10]. Finally, since each robot limb is associated with a specific thread, these QP can be solved in parallel which further increases the computational efficiency.

3.5.4.6 Experiment 6. Walk With Me: Force Propagation on a Floating-Base Robot

On a floating base robot, the interaction force applied to the robot upper limbs tends to change the pressure distribution at the contact surface between the robot feet and the ground. The ZMP will hence be shifted in the direction of the effort. Since most humanoid robot stabilizers are tracking the ZMP motions to issue reactive ankle torque or CoM shifts, the robot will thus exhibit some degree of compliance at the floating base level. Nevertheless, such setup rapidly shows limitations due to the stability constraints of humanoids, and compliance will eventually prove unachievable as the ZMP reaches the vicinity of the support polygon. In this context, a relevant strategy to increase the compliance range is to make the robot walk in the direction of the perturbation. The main problem is then not to determine in which direction to walk but rather to know when to generate the walking motion. In general, the stronger the external disturbance, the faster the robot will need to execute a reactive walking motion. This implies not only having sufficiently powerful actuators, but also being able to absorb the resulting impacts between the robot's feet and the ground. We here propose to use the artificial skin as an interface to generate suitable reactive steps on a floating base robot.

During this experiment, the upper body of the robot remains rigid. Both force and skin proximity modalities are activated. The measured interactions are directly propagated to the legs of the robot and sent to the gait controller as a set of new commands. The use of an artificial

[10] The mobile base does not requires the resolution of any QP as it does not have any parent limb.



(i) The robot is being pushed. The applied force is sensed by the artificial skin.

(ii) The force is propagated onto the floating base.

(iii) The resulting walking motions extend the compliance range.

Figure 22 The walk with me experiment. The blue arrow denotes the ground reaction force, the green arrow is the propagated wrench and the red dots are the activated skin cells.

skin allows to decouple the full-body physical interactions occurring between a robot and its environment from the disturbances arising from joint control errors as well as from ground imperfections. The resulting behavior can be observed in Figure 22. A point that should be raised here is the latency induced by the step generation process. This latency, highlighted in green on Figure 22, is estimated to be approximately one second, during which the floating base compliance will only be the result of the stabilizer. It is also worth noting that the hip oscillations induced by the static walking algorithm, and visible in Figure 23, may produce unexpected behavior for the external user in case interaction takes place on the coronal place, as the robot first moves toward the user who pushes it.

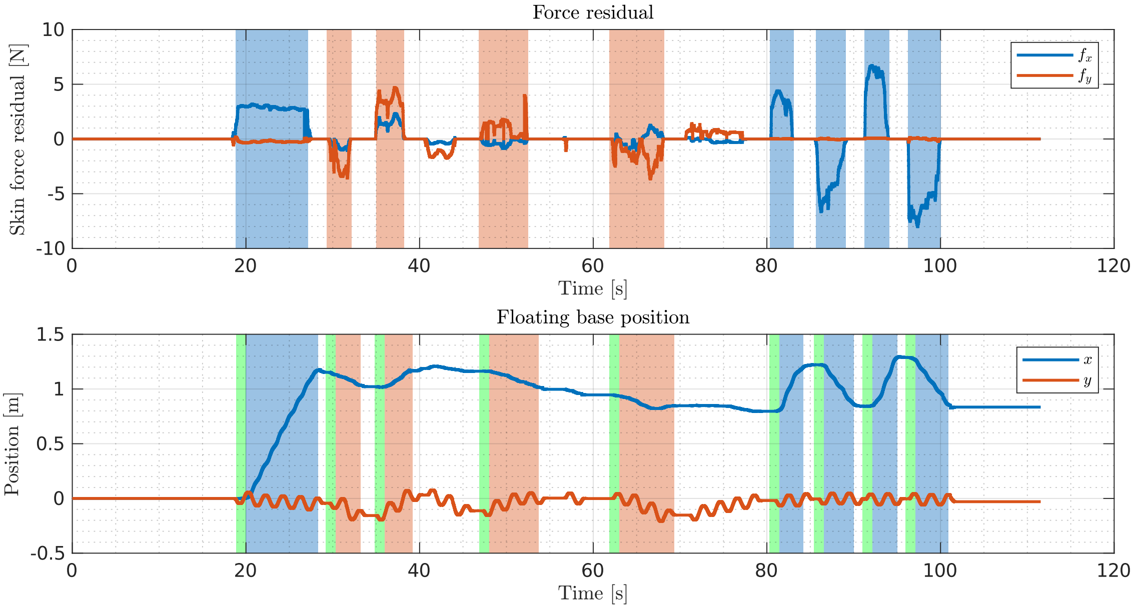


Figure 23 Force propagation on a floating base robot. The blue area denote interactions along the x direction, the orange areas interactions along the y direction and the green area the propagation latency (estimated from the raw data).

3.6. Summary

In this work, we address the issue of tactile-based whole-body compliance with force propagation for mobile and floating-base robots. Using the tactile feedback of an artificial skin, we propose a new control paradigm, based on the reactive Cartesian acceleration error between each of the robot's rigid links, and a corresponding set of virtual mass-spring-damper systems, moving with predefined compliant dynamics. The use of Quadratic Programming makes it possible to determine a set of proper control torques, allowing to track as closely as possible the desired reactive behavior while intrinsically complying with the robot physical constraints. In addition to its intuitiveness, formulating the controller in Cartesian space, allows us to conveniently capture the dynamic residuals between the desired and the actual reactive motions of each limb, and to propagate them onto a parent kinematic chain. In this manner, the robot remains compliant under external tactile perturbations, even when the arms activate kinematic or dynamic constraints. Our artificial skin here plays a major role as it provides information on the magnitude, direction and localization of tactile interactions. To the best of our knowledge, the integration of such distributed multimodal tactile feedback to a compliant robot controller was never done before. The experiments performed on a dual arm mobile manipulator and on a humanoid robot demonstrate the reliability and efficiency of our method as well as its applicability to real-life human-robot interaction scenarios. We provide a video[11] to illustrate the robot behaviors described in this work.

[11] https://youtu.be/teKbiiSN2K8

Chapter 4

Distributed Inertial Feedback for Online Estimation of High-Order Motion Derivatives

This chapter presents the contributions of this thesis in the field of first and second order motion derivatives estimation in robotics, based on distributed inertial feedback. In order to counteract the noise amplification and phase shift phenomena induced by numerical derivation and digital filtering respectively, we here investigate the possibility of fusing the joint position measurements acquired by the encoders of a floating base robot, with a set of appropriate inertial data. In our case, these data consist of acceleration and angular velocity measurements from the robot's main inertial measurement unit, combined with redundant linear acceleration measurements obtained from a multimodal artificial skin covering the robot. The issues of robustness and scalability associated with such large-scale distributed inertial measurements are addressed and discussed, following a sensor selection scheme. This chapter is structured as follows: Section 4.1 introduces the issue of distributed inertial feedback, considering the robustness and data reduction problematics. Section 4.2 provides a formal description of the redundant distributed inertial measurement problem. Section 4.3 describes the proposed approach for estimating derivative-free first and second order kinematics on floating-base robots. Section 4.4 describes this thesis' novel approach to robustness and data reduction through online sensor selection on a robot covered with artificial skin. The experiments, as well as their results, are presented and discussed in Section 4.5. Finally, Section 4.6 gives a brief outlook over our contributions and results.

4.1. Derivative-Free Estimation of High-Order Motion Derivatives using Distributed Inertial Feedback

As robots are about to be democratized in multiple sectors of society, the associated requirements in terms of performance and reactivity are pushing for the consideration of high-order motion derivatives, whether for the development of more reactive control strategies or for inertial parameter identification purposes [223, 224]. On a robot, second-order motion derivatives are usually computed by double time-derivation of noisy encoder data. However, the resulting signal often proves unsuitable for use in high-gain control loops. This is mainly due to the significant noise-amplification effect induced by numerical differentiation, as well as the inevitable signal lag caused by digital filtering [225]. A promising alternative is to estimate second-order kinematics by rigidly attaching a set of inertial measurement units (IMU) to each link of a robot. The idea of using distributed IMU sensors mounted in different points of a rigid body to estimate its motion derivatives is not intrinsically new. This approach has already been the subject of extensive research since the 1960s [226]. As a matter of fact, spatially separated

IMUs naturally provide information on both the linear and angular motion components of a body, without requiring numerical differentiation. Although initially limited by the prohibitive cost of inertial sensors, this approach is now receiving sustained attention due to the large-scale availability of very low-cost sensors, referred to as micro-electro-mechanical systems (MEMS) [7].

4.1.1. Distributed Inertial Feedback: Concept and Existing Applications

Multiple promising applications based on distributed MEMS have recently been proposed in the fields of navigation [227, 228, 229], biomechanics [230, 231, 232, 233], and robotics [234, 235, 236], in particular for joint angle [237, 238, 239, 240, 241], joint velocity [242, 243, 218, 244, 245] and joint acceleration estimation purposes [246, 247, 248, 249]. The classic approach often referred to as *accelerometer-only* or *gyroscope-free* IMU (AO-IMU resp. GF-IMU) is based solely on the measurement of multiple acceleration sensors, mounted on a given rigid body. In [227], Schuler et al. demonstrated that a set of nine single-axis accelerometers arranged in a precise manner, is theoretically sufficient to estimate both the linear and the angular acceleration components of a rigid body. With such a configuration, the body's angular velocity is usually computed by time integration of the previously estimated angular acceleration signal, thereby resulting in a potentially unbounded bias. In [250], Schopp et al. proposed to solve this issue by using nonlinear Kalman filtering techniques. Another alternative proposed in particular in [251, 218, 228], makes use of kinematically redundant sensor arrays, with at least twelve single-axis accelerometers spanning a 3D space. Such a configuration allows to directly resolve the acceleration and squared angular velocity of a rigid body, by using linear least-squares techniques. Although this approach suffers from a sign ambiguity on the angular rate motion component, the sign can be estimated using either numerical integration or more reliably with dedicated sensor fusion techniques, as proposed in [252, 253, 251, 254, 255]. In most approaches, the results suggest that the performance of such accelerometers arrays is highly dependent on the relative geometric arrangement of the sensors as well as the nature of the measured motions: typically the larger the angular velocity and the volume of the sensor array, the better the precision [218]. It must be highlighted that AO-IMUs has often been criticized for their poor performance in the low angular velocity domain [218, 256, 257]. Consequently, while this technique can be successfully used in a range of specific applications involving high angular velocities – such as high maneuverability aircraft or artillery shells – its potential applications in robotics turn our to be rather limited.

In order to solve this issue, composite arrays made of accelerometers and gyroscopes can be used as suggested in [258, 229, 231, 232]. In [231], Skog et al. used a highly redundant network of 32 triple-axis accelerometers and 32 triple-axis gyroscopes. The proposed method is formulated as a two-step optimization problem, extracting the Maximum Likelihood angular velocity estimate out of the gyroscope readings before merging it with the data collected by the accelerometer array. In [232], Wahlstrom et al. improved the concept by introducing a motion model within the computations. These methods are appealing in several respects. First of all they do no longer require complex 3D sensor geometries. Furthermore, they enable good sensitivity at low angular velocities while at the same time providing robustness to gyroscopes saturation at higher angular rates [231]. In practice, sensor redundancy is also recognized as an essential feature, allowing to significantly reduce the measurement noise [259], and providing enhanced robustness to possible sensor failures [260, 261]. This is especially relevant for applications based on low-cost consumer-grade MEMS, as such devices generally exhibit significant noise levels and biases [262]. In robotics, most approaches using distributed inertial feedback focus on the issues of joint position and velocity estimation [240, 238, 239, 241]. The main idea is to overcome the lack or low quality of joint encoders, in particular on hydraulic manipulators. Only a few works actually deal with the estimation of second-order motion derivatives [246, 247, 236, 248, 249]. In these approaches, the joint acceleration estimation problem is formulated in a recursive manner, by exploiting the robot kinematic structure. In [246], Vihonen et al. use a set of six MEMS accelerometers and one triple-axis MEMS gyroscope, rigidly attached to every link of a 3-DoF planar robot arm, to estimate joint velocities and accelerations. In [247], Vihonen et al. further use geometrical considerations in order to further reduce the number of required sensors, with promising results on a heavy duty robot. In [236], Rotella et al. mounted a navigation-grade IMU on each lower body link of a humanoid robot, and estimated the joint positions and velocities by fusing the commanded joint accelerations with the measured link angular velocities and encoder data. The joint acceleration estimation process was performed independently, using a similar approach to [246]. Finally, in [248, 249], Mclean et al. proposed a cascaded sigma-point Kalman filter architecture with a constant jerk model, to fuse inertial data from distributed MEMS sensors, with encoder measurement on a 2 DoF pendulum. Multiple sensor configurations were evaluated. The results suggest that accelerometers redundancy has a positive effect on accuracy and that the use of a gyroscope considerably improves the estimate accuracy at low angular rates, which complies with the observations made in [231], although it is worth mentioning that Mclean et al. only noted marginal improvement when using redundant gyroscope configurations.

4.1.2. Robustness and Data Reduction through Sensor Selection

One of the main obstacles to the successful deployment of large-scale whole-body distributed inertial feedback in robotics is the considerable amount of data that can potentially be generated by such a system. This is especially true in the context of inertial sensing, as the required sampling rates tend to be high (typically 10^2-10^3 Hz). To produce the intended effect on the robot, data must be transmitted and processed in real-time, at the control loop frequency. This is a complex problem, which rapidly becomes intractable when the generated amount of data exceeds the system's bandwidth. Evolutionary processes in nature converged towards a remarkably elegant and effective solution to this issue in the context of tactile perception. By contextualizing the process of data generation, the sensory receptors of the natural skin mainly transmit information that suggests a change of state (e.g., variations in temperature or force) [263]. Accordingly, multiple neuromorphic event-driven approaches to distributed sensing were recently developed with promising results [264, 265, 266], especially for applications involving massively redundant arrays of multimodal sensors, such as artificial robot skins [267, 268, 269, 6]. Although such approaches could potentially be applied to raw inertial data, suitable implementation of the event generation engine usually require a dedicated sensor hardware. In the more general case where distributed sensors can only operate in continuous mode, the issue of data flow limitation can still be adequately addressed, provided that it is treated as a sensor-selection problem, namely finding for each robot link i equipped with a set of N_i inertial sensors, a subset of M_i sensing elements that provide the best compromise in terms of the network load and of an application-relevant metric. Once identified, the selected sensors are set to broadcast measurement at the maximum frequency while other sensors are deactivated or run at a lower update rate. Sensor-selection is a widely studied topic, with multiple contributions, ranging from energy optimization in redundant WiFi networks [270] to distributed state estimation and tracking [271, 272]. Sensor selection has many potential uses in robotics, especially for redundant inertial sensors. As a matter of fact, most state-of-the-art distributed inertial measurement techniques are based on the assumption that the sensors are rigidly coupled to the robot link to which they are mounted. However this assumption is not always verified in practice, considering that the sensors that come into direct contact with the external environment are likely to undergo local deformations and impacts, which in turn induces inconsistencies into the inertial measurements collected by the system (c.f. Fig. 24.c). In this context the ability to reorganize the sensor network online, eliminating corrupted elements, i.e., sensors in error states or sensors reporting notoriously erroneous data, would prove invaluable. However, for this to be achievable, this re-organization step must be done as fast as possible. Sensor-selection is often considered as a NP-hard combinatorial optimization problem [273]. As a result, the approaches to solving the sensor-selection problems are essentially application-specific [274]. If the accuracy of the result is of prime importance, global optimization techniques will be considered, regardless of the required computational effort. This is for example the case in [275] where a branch-and-bound method is presented. However if the considered application is subject to strict constraints in terms of timing or computational power, as it is often the case in robotics, local sub-optimal heuristic approaches will be preferred as for example proposed in [276] based on convex optimization techniques or in [277, 278] using tailor-made greedy algorithms.

4.1.3. Proposed Approach and Contributions

This section introduces the proposed approach to distributed inertial sensor fusion, as well as the developed method for automatic selection of optimal sensor configurations.

4.1.3.1 Estimation of first and second order motion derivatives

We propose a new derivation-free estimation method for second-order kinematics on floating-base robots, relying on highly-redundant distributed acceleration feedback. The main contribution of this work is to provide estimates of the first- and second-order joint motion derivatives, with sufficiently low noise and phase shift to be potentially applicable in high gain control loops or online inertial parameter identification. We proceed by covering a humanoid robot

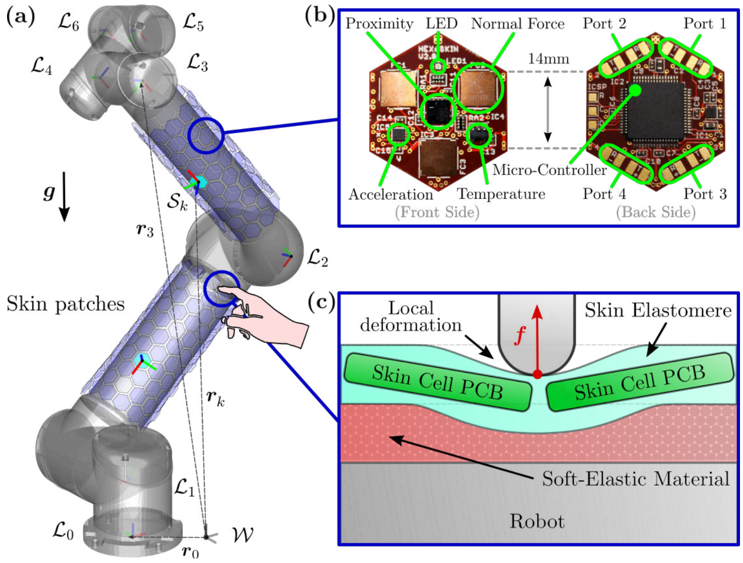


Figure 24 (a-b): One of our experimental platforms is a UR5 industrial robot from Universal robotics, covered with 253 artificial skin cells. **(c)**: The skin is mounted over a layer of soft-elastic material to better fit the shape of the robot and dampen high-frequency mechanical vibrations. During physical interaction with the external environment, this layer is likely to deform, thereby slightly changing the orientation of the sensors attached to it.

with a multimodal artificial skin [202, 203, 6]. The skin is a network of interconnected sensing units or "*cells*", organized in patches of different sizes (see Fig .14). It is, in particular capable of measuring three-dimensional accelerations, at the corresponding cells' mounting points. We propose to fuse the skin acceleration data, with that of the joint encoders and the central inertial measurement unit (IMU), mounted on the floating-base of our robot. Our approach is two-fold: the skin data is first processed locally, at the link level, using the approach of Skog et al. [231], to provide estimates of the linear and angular accelerations of each robot link in Cartesian space. We then fuse the obtained acceleration estimates with the data measured by the joint encoders and the main IMU, using a Sigma-point Kalman filter with a third-order motion process. We compensate for the lack of link-mounted gyroscopes – required in the approach of [231] – by using the velocity state estimate provided by the Kalman filter. The proposed approach is validated both in simulation and through experiments on a floating-base robot covered with artificial skin.

4.1.3.2 Online sensor selection for enhanced robustness and efficiency

Our second main contribution consists in a new online sensor selection algorithm, that allows to automatically determine suitable combinations of inertial sensors, for robot links acceleration estimation purposes. For convenience reasons, we here consider the case of a UR5 industrial robot, covered with a multimodal artificial skin. Although multiple approaches already tackled the problem of inertial sensor *placement*, in particular in the context of gyroscope-free navigation (see for instance [279, 280, 281, 282, 283] and the references therein), the issue of inertial sensor *selection* among larger redundant sets with fixed geometries is seldom addressed. Similarly to [276], our approach does not intend to find the globally optimal sensor configuration, but rather focuses on providing a "sufficiently good" sub-optimal configuration, fast enough to allow online implementation and adaptation on a robot. We formulate the sen-

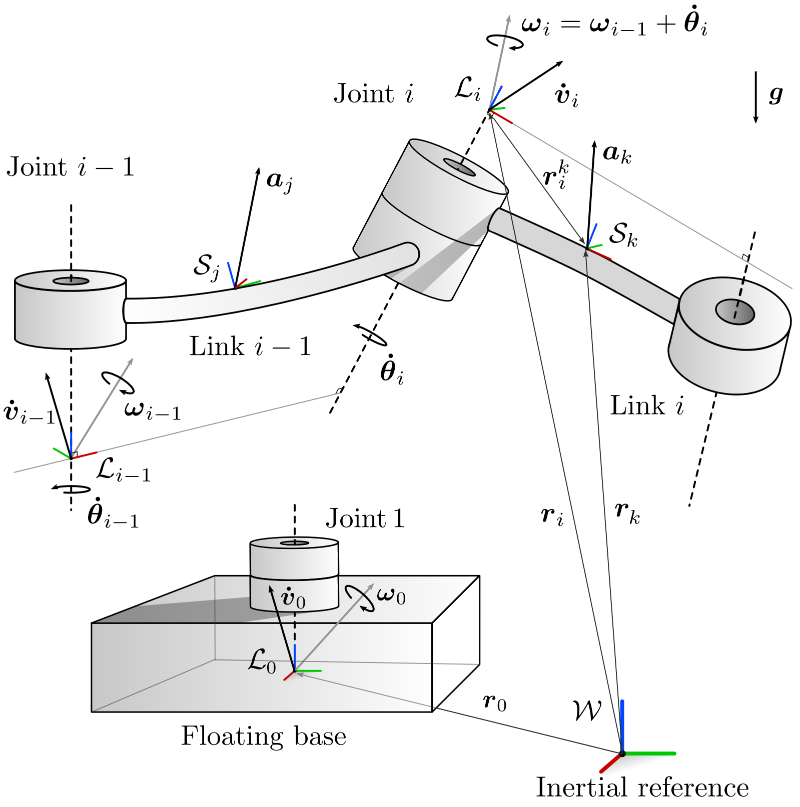


Figure 25 Illustration of the motions of a rigid robot with respect to an inertial frame $\mathcal{W}$. For each link i of the robot, we define a rigidly-attached body frame $\mathcal{L}_i$ and for each sensor k mounted on this link, we define the frame $\mathcal{S}_k$. The position of $\mathcal{L}_i$ and $\mathcal{S}_k$ relative to the inertial frame $\mathcal{W}$ are denoted by the three-dimensional vectors $\boldsymbol{r}_i$ and $\boldsymbol{r}_k$, respectively.

sor selection problem as a greedy algorithm, guided by a locally projected gradient heuristics. The adaptability of our approach stems from the fact that we can use the precontact information provided by the proximity sensors of the artificial skin in order to automatically trigger a re-computation of the sensor configuration, excluding the cells and neighbors that are likely to get into contact with the external environment. We validate our approach, both in simulation and through experiments on a UR5 industrial manipulator covered with a multimodal artificial skin.

4.2. Fusion of Distributed Inertial Measurements

This section presents the two main approaches found in scientific literature, to combine the different measurements from a set of inertial sensors mounted over a rigid body into a single measurement vector containing the linear and angular accelerations of this body.

4.2.1. Fundamentals of Distributed Inertial Feedback

Let us consider the movements of two contiguous revolute rigid robot links $i-1$ and i in an inertial reference frame $\mathcal{W}$, as depicted in Fig. 25. The links $i-1$ and i are equipped with N_{i-1} and N_i triple-axis acceleration sensors, respectively. Each sensor $k = \{1 \cdots N_i\}$ of link i measures a linear acceleration ${}^{\mathcal{S}_k}\boldsymbol{a}_k$ in a body-attached sensor frame $\mathcal{S}_k$. This acceleration is the result of two main components, namely the acceleration of gravity $\boldsymbol{g} = [0\ 0\ -9.81]^\top\, m.s^{-2}$ and the acceleration $\boldsymbol{a}_k \in \mathbb{R}^3$ of the sensor frame $\mathcal{S}_k$, induced by the relative movements of the robot joints:

$$ {}^{\mathcal{S}_k}\boldsymbol{a}_k = \boldsymbol{R}_{\mathcal{W}}^{\mathcal{S}_k}\left(\boldsymbol{a}_k + \boldsymbol{g}\right) + {}^{\mathcal{S}_k}\boldsymbol{b}_k + {}^{\mathcal{S}_k}\boldsymbol{n}_k \in \mathbb{R}^3. \tag{133} $$

In this expression, $\boldsymbol{R}_{\mathcal{W}}^{\mathcal{S}_k} \in \boldsymbol{SO}(3)$ denotes the rotation matrix relating the sensor frame $\mathcal{S}_k$ to the inertial frame $\mathcal{W}$, ${}^{\mathcal{S}_k}\boldsymbol{b}_k \in \mathbb{R}^3$ is the sensor bias vector and ${}^{\mathcal{S}_k}\boldsymbol{n}_k \sim \mathcal{N}(\boldsymbol{0}_{3\times1}, \boldsymbol{\sigma}_a)$ is the sensor noise vector. The acceleration $\boldsymbol{a}_k$ of frame $\mathcal{S}_k$ can be related to the Cartesian acceleration $\ddot{\boldsymbol{x}}_i = \left[\dot{\boldsymbol{v}}_i^\top,\ \dot{\boldsymbol{\omega}}_i^\top\right]^\top \in \mathbb{R}^6$ of a local coordinate frame $\mathcal{L}_i$, rigidly attached to link i (c.f. Fig. 25), using the following measurement equation:

$$ \underbrace{\boldsymbol{a}_k}_{\text{Sensor frame acceleration}} = \underbrace{\dot{\boldsymbol{v}}_i}_{\text{Link frame acceleration}} + \underbrace{\boldsymbol{\omega}_i \times \left(\boldsymbol{\omega}_i \times \boldsymbol{r}_i^k\right)}_{\text{Centrifugal acceleration}} + \underbrace{\dot{\boldsymbol{\omega}}_i \times \boldsymbol{r}_i^k}_{\text{Euler acceleration}}, \tag{134} $$

where $\boldsymbol{r}_i^k = \boldsymbol{r}_k - \boldsymbol{r}_i \in \mathbb{R}^3$ is the position vector of the sensor frame $\mathcal{S}_k$ relative to the joint frame $\mathcal{L}_i$, expressed in the coordinate system of $\mathcal{W}$, and $\forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^3, \boldsymbol{u} \times \boldsymbol{v}$ denotes the cross-product between vectors $\boldsymbol{u}$ and $\boldsymbol{v}$. Note that eq. (134) does not contain any Coriolis term as we assume that every sensor k is rigidly attached to the robot link i, and thus that the distance between $\mathcal{L}_i$ and $\mathcal{S}_k$ does not change.

4.2.2. Fusion of the Redundant Skin Acceleration Measurements

In order to avoid excessive computation load, the first step toward the estimation of second-order kinematics is to merge the noisy redundant acceleration measurements of the artificial skin cells into a set of linear and angular link acceleration signals. To this end, it is possible to rewrite eq. (134) in the form of a linear system, as for example proposed in [251, 218, 228]. Introducing the augmented state vector $\boldsymbol{\psi}_i = \left[\dot{\boldsymbol{v}}_i^\top\ \dot{\boldsymbol{\omega}}_i^\top\ \left[\omega_x^2\ \omega_y^2\ \omega_z^2\ \omega_x\omega_y\ \omega_y\omega_z\ \omega_z\omega_x\right]^\top\right]^\top \in \mathbb{R}^{12}$ and the associated observation matrix $\boldsymbol{G}_i^k = \left[\boldsymbol{I}_{3\times3}\ -\left[\boldsymbol{r}_i^k\right]_\times\ \left[\boldsymbol{r}_i^k\right]_\bullet\right] \in \mathbb{R}^{3\times12}$ where $\forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^3\ [\boldsymbol{u}]_\times \boldsymbol{v} = \boldsymbol{u} \times \boldsymbol{v}$, $[\boldsymbol{u}]_\bullet = \begin{bmatrix} 0 & -u_x & -u_x & u_y & 0 & u_z \\ -u_y & 0 & -u_y & u_x & u_z & 0 \\ -u_z & -u_z & 0 & 0 & u_y & u_x \end{bmatrix}$, it is possible to write:

$$ \boldsymbol{a}_k = \boldsymbol{G}_i^k \boldsymbol{\psi}_i. \tag{135} $$

Stacking the measurements of at least four acceleration sensors triads in an augmented acceleration vector $\boldsymbol{y} = \left[\boldsymbol{a}_1^\top \cdots \boldsymbol{a}_{N_i}^\top\right]^\top \in \mathbb{R}^{3N_i}$ and its corresponding augmented observation matrix $\boldsymbol{G}_i = [\boldsymbol{G}_i^{1\top} \cdots \boldsymbol{G}_i^{N_i\top}]^\top \in \mathbb{R}^{3N_i\times12}$, allows to inverse the previous expression, thereby resulting in the following acceleration estimate:

$$ \boldsymbol{\psi}_i = \boldsymbol{G}_i^{\#} \boldsymbol{y}, \tag{136} $$

where $\boldsymbol{G}_i^{\#} = (\boldsymbol{G}_i^\top \boldsymbol{G}_i)^{-1}\boldsymbol{G}_i^\top$ denotes the left pseudo-inverse of $\boldsymbol{G}_i$. Although relatively simple to implement, this method has several disadvantages, ranging from the sign ambiguity[1] on $\boldsymbol{\omega}_i$, to poor performances at low angular velocities, which is, of course, problematic for robotic applications. Moreover, the conditioning of $\boldsymbol{G}_i$ directly depends on the geometry of the sensor array, and the latter must span a 3D space in order to avoid rank deficiency. A workaround presented in [231], consists in reformulating eq. (134) as the sum of a linear and of a non-linear term, given fixed values of $\dot{\boldsymbol{v}}_i, \boldsymbol{\omega}_i$, and $\dot{\boldsymbol{\omega}}_i$:

$$\begin{aligned} \boldsymbol{a}_k &= \dot{\boldsymbol{v}}_i + \left([\boldsymbol{\omega}_i]_\times^2 + [\dot{\boldsymbol{\omega}}_i]_\times\right)\boldsymbol{r}_i^k && \text{(137a)} \\ &= \underbrace{\begin{bmatrix}\boldsymbol{I}_{3\times3} & -\left[\boldsymbol{r}_i^k\right]_\times\end{bmatrix}}_{\boldsymbol{H}_i^k}\underbrace{\begin{bmatrix}\dot{\boldsymbol{v}}_i \\ \dot{\boldsymbol{\omega}}_i\end{bmatrix}}_{\ddot{\boldsymbol{x}}_i} + \underbrace{[\boldsymbol{\omega}_i]_\times^2\,\boldsymbol{r}_i^k}_{\boldsymbol{\pi}_i^k(\boldsymbol{\omega}_i)}. && \text{(137b)} \end{aligned}$$

Using this expression and provided that a reliable information of $\boldsymbol{\omega}_i$ is available through other sources, such as body-attached gyroscopes, the linear and angular acceleration terms $\dot{\boldsymbol{v}}_i$ and $\dot{\boldsymbol{\omega}}_i$ of link i can be obtained using a weighted left pseudo-inverse:

$$\ddot{\boldsymbol{x}}_i = \begin{bmatrix}\dot{\boldsymbol{v}}_i \\ \dot{\boldsymbol{\omega}}_i\end{bmatrix} = \left(\boldsymbol{H}_i^\top \boldsymbol{\Sigma}_a^{-1}\boldsymbol{H}_i\right)^{-1}\boldsymbol{H}_i^\top \boldsymbol{\Sigma}_a^{-1}\left(\boldsymbol{y} - \boldsymbol{\pi}_i(\boldsymbol{\omega}_i)\right) \tag{138}$$

where $\boldsymbol{y} = \left[\boldsymbol{a}_1^\top \cdots \boldsymbol{a}_{N_i}^\top\right]^\top \in \mathbb{R}^{3N_i}$ is the augmented acceleration measurement vector, the terms $\boldsymbol{H}_i = \left[\boldsymbol{H}_i^{1\top} \cdots \boldsymbol{H}_i^{N_i\top}\right]^\top \in \mathbb{R}^{3N_i\times6}$ and $\boldsymbol{\pi}_i(\boldsymbol{\omega}_i) = \left[\boldsymbol{\pi}_i^1(\boldsymbol{\omega}_i)^\top \cdots \boldsymbol{\pi}_i^{N_i}(\boldsymbol{\omega}_i)^\top\right]^\top \in \mathbb{R}^{3N_i}$ respectively refer to the augmented observation matrix and centrifugal acceleration vector, and $\boldsymbol{\Sigma}_a \in \mathbb{R}^{3N_i\times3N_i}$ is the accelerometers' noise covariance matrix. This approach does not induce any sign ambiguity and, more important, complies with 2D sensor geometries, as pointed out in [231]. Nevertheless it is worth noting that its implementation is not straightforward in our case since no direct measurement of $\boldsymbol{\omega}_i$ is available.

4.3. Estimation of Second-order Robot Motion Derivatives using Redundant Distributed Inertial Feedback

This section presents the details of our approach to fusing redundant inertial sensor data, with encoders data, for joint velocity and acceleration estimation purposes.

4.3.1. Fusion of Inertial and Joint Sensor Data

In this section, we propose to fuse the n-dimensional joint position data $\boldsymbol{q} \in \mathbb{R}^n$ together with the Cartesian-space inertial measurements provided by the artificial skin and the floating-base IMU. Our objective is to estimate the first- and second-order joint-space motion derivatives $\dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}} \in \mathbb{R}^n$. The sensor fusion process is carried out using a Kalman filter. Relative link motions on humanoid robots are described using a set of kinematic equations, usu-

[1] The sign ambiguity on angular velocity arises from the fact that the components of $\boldsymbol{\omega}_i$ are in a quadratic form within $\boldsymbol{\psi}_i$. This is a direct consequence of the centrifugal acceleration term in eq. (134).

ally expressed with respect to a floating-base non-Galilean frame of reference denoted $\mathcal{L}_0$ (c.f. Fig. 25). In this context, mapping the joint velocities and accelerations to the measurements of a set of acceleration sensors placed on the robot body requires reliable estimates of the angular rate $\boldsymbol{\omega}_0$ as well as the linear and angular accelerations $\dot{\boldsymbol{v}}_0$ and $\dot{\boldsymbol{\omega}}_0$ of the floating-base with respect to an inertial frame[2] $\mathcal{W}$. We thus define the state vector $\boldsymbol{\Theta} = [{}^{\mathcal{L}_0}\ddot{\boldsymbol{v}}_0^\top \; {}^{\mathcal{L}_0}\dot{\boldsymbol{v}}_0^\top \; {}^{\mathcal{L}_0}\ddot{\boldsymbol{\omega}}_0^\top \; {}^{\mathcal{L}_0}\dot{\boldsymbol{\omega}}_0^\top \; {}^{\mathcal{L}_0}\boldsymbol{\omega}_0^\top \; \dddot{\boldsymbol{q}}^\top \; \ddot{\boldsymbol{q}}^\top \; \dot{\boldsymbol{q}}^\top \; \boldsymbol{q}^\top \;]^\top \in \mathbb{R}^{4 \cdot n+15}$ as a concatenation of the floating-base and joint-space motion derivatives, where ${}^{\mathcal{L}_0}\ddot{\boldsymbol{v}}_0, {}^{\mathcal{L}_0}\ddot{\boldsymbol{\omega}}_0 \in \mathbb{R}^3$ and $\dddot{\boldsymbol{q}} \in \mathbb{R}^n$ respectively denote the floating-base and joint-space jerk signals. Similar to [248, 249], we make use of a third order motion model in order to account for abrupt changes in acceleration that may arise from contact between the robot and its environment. Note that the floating base motion derivatives are still expressed relative to the inertial frame $\mathcal{W}$ but written in the coordinate system $\mathcal{L}_0$ of the floating-base, namely ${}^{\mathcal{L}_0}\dot{\boldsymbol{v}}_i = \boldsymbol{R}_{\mathcal{W}}^{\mathcal{L}_0}\dot{\boldsymbol{v}}_i$, ${}^{\mathcal{L}_0}\dot{\boldsymbol{\omega}}_i = \boldsymbol{R}_{\mathcal{W}}^{\mathcal{L}_0}\dot{\boldsymbol{\omega}}_i$, and ${}^{\mathcal{L}_0}\boldsymbol{\omega}_i = \boldsymbol{R}_{\mathcal{W}}^{\mathcal{L}_0}\boldsymbol{\omega}_i$, where $\boldsymbol{R}_{\mathcal{W}}^{\mathcal{L}_0} \in \boldsymbol{SO}(3)$ is the rotation matrix relating $\mathcal{L}_0$ to the inertial frame $\mathcal{W}$. This is justified by the fact that unlike $\boldsymbol{R}_{\mathcal{W}}^{\mathcal{L}_0}$, the homogeneous transforms between $\mathcal{L}_0$ and any frame $\mathcal{L}_i$ or $\mathcal{S}_k$ on the robot kinematic tree are available and accurate. Assuming that the time evolution of the jerk signal can be represented as a Gaussian noise with standard deviations $\boldsymbol{\sigma}_{fb}$ for the floating base and $\boldsymbol{\sigma}_{jt}$ for the kinematic tree, we define the following time and measurement update equations:

$$\boldsymbol{\Theta}_k^+ = \underbrace{\boldsymbol{\Theta}_k}_{\text{Current state vector estimate}} + \begin{bmatrix} \mathbf{0}_{3\times1} \\ {}^{\mathcal{L}_0}\ddot{\boldsymbol{v}}_0\Delta t \\ \mathbf{0}_{3\times1} \\ {}^{\mathcal{L}_0}\ddot{\boldsymbol{\omega}}_0\Delta t \\ {}^{\mathcal{L}_0}\dot{\boldsymbol{\omega}}_0\Delta t + {}^{\mathcal{L}_0}\ddot{\boldsymbol{\omega}}_0\dfrac{\Delta t^2}{2} \\ \mathbf{0}_{n\times1} \\ \dddot{\boldsymbol{q}}\Delta t \\ \ddot{\boldsymbol{q}}\Delta t + \dddot{\boldsymbol{q}}\dfrac{\Delta t^2}{2} \\ \dot{\boldsymbol{q}}\Delta t + \ddot{\boldsymbol{q}}\dfrac{\Delta t^2}{2} + \dddot{\boldsymbol{q}}\dfrac{\Delta t^3}{6} \end{bmatrix}_k + \boldsymbol{r}_k \tag{139}$$

$$\underbrace{{}^{\mathcal{L}_0}\Delta\boldsymbol{\zeta}_k}_{\text{Innovation vector}} = \underbrace{{}^{\mathcal{L}_0}\hat{\boldsymbol{\zeta}}_k}_{\text{Measurement vector}} - \underbrace{\big(\boldsymbol{\varphi}(\boldsymbol{\Theta}_k^+) + \boldsymbol{w}_k\big)}_{\text{Observation prediction}} \tag{140}$$

In this expression, $\boldsymbol{\varphi}(\boldsymbol{\Theta}) = \left[\boldsymbol{\varphi}_0^\top(\boldsymbol{\Theta}) \; \cdots \boldsymbol{\varphi}_n^\top(\boldsymbol{\Theta})\right]^\top$ denotes the nonlinear measurement model and ${}^{\mathcal{L}_0}\hat{\boldsymbol{\zeta}}_k$ the measurement vector. The noise terms $\boldsymbol{r}_k \sim \mathcal{N}(\mathbf{0}_{(4\cdot n+15)\times1}, \boldsymbol{\Sigma}_{\boldsymbol{vv}})$, and $\boldsymbol{w}_k \sim \mathcal{N}(\mathbf{0}_{m\times1}, \boldsymbol{\Sigma}_{\boldsymbol{ww}})$ correspond to the process and measurement noise vectors, and the square matrices $\boldsymbol{\Sigma}_{\boldsymbol{vv}} \in \mathbb{R}^{(4\cdot n+15)\times(4\cdot n+15)}$ and $\boldsymbol{\Sigma}_{\boldsymbol{ww}} \in \mathbb{R}^{m\times m}$ are the associated process and measurement noise covariance matrices, respectively. In practice, model non-linearities can be addressed in several different manners [76]. In an Extended Kalman Filter (EKF), the prior estimate is propagated through a first-order linearized version of the model. Although the resulting computational cost is low, the first order linearization also induces a bias in the

[2] This is usually the case on humanoid robots, most of which are equipped with an industrial-grade IMU mounted on the floating base.

posterior mean and covariance of the estimate. This bias can be non-negligible when the studied system is highly nonlinear. In a Sigma-Point Kalman Filter (SPKF), a set of deterministic samples of the prior estimate distribution are propagated through the true nonlinear model, thereby preserving the posterior mean and covariance, but at the price of higher computational cost. In this work, we use an Unscented Kalman filter (UKF), namely a variety of sigma-point Kalman filter. The detailed UKF processing pipeline is given in Appendix C. The reader is referred to [76] for a deeper discussion on Sigma-Point Kalman filters, their performance and potential applications.

4.3.2. Detailed Description of the Measurement Update

4.3.2.1 Observation prediction

On a floating-base robot such as a humanoid, the linear and angular accelerations $\dot{\boldsymbol{v}}_i$ and $\dot{\boldsymbol{\omega}}_i$ of every link frame $\mathcal{L}_i$, expressed in the inertial frame $\mathcal{W}$, can be related to the floating-base linear and angular motion derivatives $\dot{\boldsymbol{v}}_0, \dot{\boldsymbol{\omega}}_0, \boldsymbol{\omega}_0$, and to the joint state derivatives $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}} \in \mathbb{R}^n$, by noting that:

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} + \dot{\boldsymbol{\theta}}_i \tag{141a}$$

$$\dot{\boldsymbol{\omega}}_i = \dot{\boldsymbol{\omega}}_{i-1} + \ddot{\boldsymbol{\theta}}_i + \boldsymbol{\omega}_{i-1} \times \dot{\boldsymbol{\theta}}_i \tag{141b}$$

$$\dot{\boldsymbol{v}}_i = \dot{\boldsymbol{v}}_{i-1} + \boldsymbol{\omega}_{i-1} \times \left(\boldsymbol{\omega}_{i-1} \times \boldsymbol{r}_{i-1}^{i}\right) + \dot{\boldsymbol{\omega}}_{i-1} \times \boldsymbol{r}_{i-1}^{i} \tag{141c}$$

where $\dot{\boldsymbol{\theta}}_i = \boldsymbol{\omega}_{i-1}^{i} = \dot{q}_i \tilde{\boldsymbol{z}}_i$ and $\ddot{\boldsymbol{\theta}}_i = \dot{\boldsymbol{\omega}}_{i-1}^{i} = \ddot{q}_i \tilde{\boldsymbol{z}}_i$ are respectively the angular velocity and acceleration resulting from the action of joint i, expressed in $\mathcal{W}$, and where $\tilde{\boldsymbol{z}}_i$ denotes the motion axis of the i^{th} joint, also expressed in $\mathcal{W}$. Combining these expressions together leads to the following recursion [284]:

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_0 + \sum_{p=1}^{i} \dot{\boldsymbol{\theta}}_p \tag{142a}$$

$$\dot{\boldsymbol{\omega}}_i = \dot{\boldsymbol{\omega}}_0 + \sum_{p=1}^{i} \ddot{\boldsymbol{\theta}}_p + \sum_{p=1}^{i} \left[\left(\boldsymbol{\omega}_0 + \sum_{q=1}^{p-1} \dot{\boldsymbol{\theta}}_q\right) \times \dot{\boldsymbol{\theta}}_p\right] \tag{142b}$$

$$\dot{\boldsymbol{v}}_i = \dot{\boldsymbol{v}}_0 + \sum_{p=0}^{i-1} \left[\boldsymbol{\omega}_p \times \left(\boldsymbol{\omega}_p \times \boldsymbol{r}_p^{p+1}\right) + \dot{\boldsymbol{\omega}}_p \times \boldsymbol{r}_p^{p+1}\right] \tag{142c}$$

The observation prediction of the measurement update step eq. (140) is obtained by mapping eq. (142) into the floating base frame $\mathcal{L}_0$, namely:

$$\varphi(\boldsymbol{\Theta}_k^+) = \left[{}^{\mathcal{L}_0}\dot{\boldsymbol{v}}_{\boldsymbol{0}}^\top(\boldsymbol{\Theta}_k^+), {}^{\mathcal{L}_0}\dot{\boldsymbol{\omega}}_{\boldsymbol{0}}^\top(\boldsymbol{\Theta}_k^+), {}^{\mathcal{L}_0}\boldsymbol{\omega}_{\boldsymbol{0}}^\top(\boldsymbol{\Theta}_k^+), {}^{\mathcal{L}_0}\dot{\boldsymbol{v}}_{\boldsymbol{n}}^\top(\boldsymbol{\Theta}_k^+), {}^{\mathcal{L}_0}\dot{\boldsymbol{\omega}}_{\boldsymbol{n}}^\top(\boldsymbol{\Theta}_k^+), \boldsymbol{q}^\top\right]^\top \tag{143}$$

4.3.2.2 Measurement vector

The skin acceleration data is processed locally, in a link-wise manner, using the Maximum Likelihood acceleration observer proposed by Skog et al. [231], to provide estimates ${}^{\mathcal{L}_0}\hat{\dot{\boldsymbol{v}}}_i$

and ${}^{\mathcal{L}_0}\hat{\dot{\boldsymbol{\omega}}}_i$ of the linear and angular accelerations of each robot link in Cartesian space. As mentioned earlier, this observer requires input information about the link's angular velocity of ${}^{\mathcal{L}_0}\boldsymbol{\omega}_i$. In [231], this information is provided by a set of gyroscope sensors. Here, we propose to use the estimate of the Kalman filter to build our observation. The equation eq. (138) of the observer then becomes:

$$\begin{bmatrix} {}^{\mathcal{L}_0}\hat{\dot{\boldsymbol{v}}}_i \\ {}^{\mathcal{L}_0}\hat{\dot{\boldsymbol{\omega}}}_i \end{bmatrix}_k = \left(\boldsymbol{H}_i^\top \boldsymbol{\Sigma}_a^{-1} \boldsymbol{H}_i\right)^{-1} \boldsymbol{H}_i^\top \boldsymbol{\Sigma}_a^{-1} \left({}^{\mathcal{L}_0}\hat{\boldsymbol{y}}_k - \boldsymbol{\pi}_i(\boldsymbol{\Theta}_k)\right) \tag{144}$$

where ${}^{\mathcal{L}_0}\hat{\boldsymbol{y}}$ denotes the acceleration vectors measured by the skin and the floating-base IMU, expressed in the floating base coordinate frame $\mathcal{L}_0$. Hence the expression of the measurement vector ${}^{\mathcal{L}_0}\hat{\boldsymbol{\zeta}}_k$ of eq. (140) can be written as:

$${}^{\mathcal{L}_0}\hat{\boldsymbol{\zeta}}_k = \left[{}^{\mathcal{L}_0}\hat{\dot{\boldsymbol{v}}}_{\boldsymbol{0}}^\top, {}^{\mathcal{L}_0}\hat{\dot{\boldsymbol{\omega}}}_{\boldsymbol{0}}^\top, {}^{\mathcal{L}_0}\hat{\boldsymbol{\omega}}_{\boldsymbol{0}}^\top, \cdots {}^{\mathcal{L}_0}\hat{\dot{\boldsymbol{v}}}_{\boldsymbol{n}}^\top, {}^{\mathcal{L}_0}\hat{\dot{\boldsymbol{\omega}}}_{\boldsymbol{n}}^\top, \hat{\boldsymbol{q}}^\top\right]_k^\top \tag{145}$$

where $\hat{[\cdot]}$ denotes a measured quantity. Note that in this expression, ${}^{\mathcal{L}_0}\hat{\boldsymbol{\omega}}_{\boldsymbol{0}}^\top$ is directly measured by the gyroscope sensor of the floating-base IMU.

4.4. Online Sensor Selection Algorithm for Data Reduction and Enhanced Robustness

This section discusses the details of our approach to online inertial sensor selection.

4.4.1. Distributed Inertial Sensor Selection: Problem Formulation

4.4.1.1 Formulation as a Combinatorial Optimization Problem

The task of selecting M_i distinct sensing elements among a larger finite set of N_i sensors can be formulated as a combinatorial optimization problem, of the form:

$$\boldsymbol{u}_i^\star = \arg\min_{\boldsymbol{u}_i} f(\boldsymbol{u}_i), \quad \text{s.t.} \quad \boldsymbol{1}_{N_i}^\top \boldsymbol{u}_i = M_i \tag{146}$$

where the optimization variable $\boldsymbol{u}_i \in \{0,1\}^{N_i}$ is a Boolean sensor selection vector and $f(\boldsymbol{u}_i) : \{0,1\}^{N_i} \rightarrow \mathbb{R}_+$ is an application-dependent value function. Since the fusion of spatially-separated inertial measurements can be considered as an inverse problem, the result is, therefore, intrinsically dependent on the conditioning of the corresponding observation matrix (i.e. $\boldsymbol{G}_i$ in eq. (138) or $\boldsymbol{H}_i$ in eq. (135)). The conditioning is a function of both the number of sensors and the geometric arrangement of the sensor array. Therefore it can be used as a value function and optimized by modifying the selection vector or the sensor placement in space. Previous work in the context of gyro-free navigation [285] demonstrated that the best results in terms of sensor placement were obtained when the acceleration sensors were located on the vertices of a platonic solid, as in this case, the observation matrix turns out to be isotropic, i.e., with a unitary condition number. Another relevant figure of merit in the context of our application, and more generally for consumer MEMS-based distributed inertial sensor networks, is the measurement noise mitigation capability. This can be interpreted as minimizing the volume of the confidence ellipsoid of the acceleration estimate [286, 287, 288], which is actually proportional to the square-root of the determinant of the covariance matrix $\boldsymbol{\Sigma}_i$. This thesis elaborates on a compromise between these two metrics and defined the value function $f(\boldsymbol{u}_i) : \{0,1\}^{N_i} \to \mathbb{R}_+$ of the robot link i as a weighted sum of these two terms, plus a penalty on the number of sensors namely:

$$f(\boldsymbol{u}_i) = k_1 \cdot \operatorname{cond}(\boldsymbol{\Sigma}_i^{-1}(\boldsymbol{u}_i)) + k_2 \cdot \sqrt{\det(\boldsymbol{\Sigma}_i(\boldsymbol{u}_i))} + k_3 \cdot \boldsymbol{u}_i^\top \boldsymbol{u}_i \tag{147}$$

where $k_1, k_2, k_3 \in \mathbb{R}_+$ are heuristically determined tuning coefficients. Although solving such a problem is in theory feasible by brute-forcing every possible sensor combination, this proves intractable in practice since this number grows with the binomial coefficient of M_i and N_i. As a matter of fact, this problem can be solved using dedicated combinatorial or mixed-integer solvers, and our first approach was to resolve this optimization problem directly, using a genetic algorithm. Although this method converges, its practical implementation turns out to be incompatible with the requirements of an online implementation in a real-time control loop, the convergence being much too slow. The following sections proposes a set of simplification allowing to eventually address this issue and propose an algorithm capable of running online, on a robot.

4.4.1.2 Simplification for Online Implementation

It is worth noticing that eq. (147) can be reformulated as a function $f(\boldsymbol{z}_i) : \mathbb{R}^{3\times M_i} \to \mathbb{R}_+$ of a continuous Cartesian space variable $\boldsymbol{z}_i \in \mathbb{R}^{3\times M_i}$ in place of the binary sensor selection vector $\boldsymbol{u}_i$ provided that the number M_i of sensors is predefined:

$$f(\boldsymbol{z}_i) = k_1 \cdot \operatorname{cond}(\boldsymbol{\Sigma}_i^{-1}(\boldsymbol{z}_i)) + k_2 \cdot \sqrt{\det(\boldsymbol{\Sigma}_i(\boldsymbol{z}_i))} \tag{148}$$

This in turns allows reformulating eq. (146) as a smooth optimization problem, that can be solved efficiently using standard line-search methods. In effect, the variable $\boldsymbol{z}_i$ concatenates the 3D positions of the set of M_i selected sensors relative to a body-attached reference frame. Although in essence the geometric arrangement of the skin sensors is predefined and fixed[3], this formulation allows the sensitivities of the cost function to be determined in terms of Cartesian variables, which in turn can be associated with cell switching events using a dedicated heuristic.

4.4.2. Projected Gradient Heuristics

In this thesis, the proposed heuristic to minimize eq. (148) is formulated as a projected gradient descent. The main idea is to limit the number of sensor combinations that have to be explored by locally analyzing the sensitivities of the cost function in terms of 3-dimensional Cartesian coordinates and projected onto the set of feasible solutions (i.e. the skin patch).

[3] The geometry of the skin patch is not supposed undergo drastic changes during the experiment.

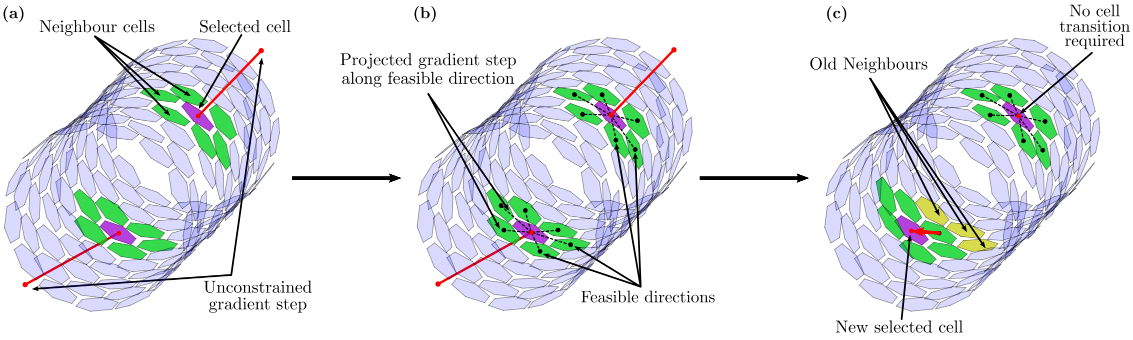

(a)
Neighbour cells
Selected cell
Unconstrained gradient step
(b)
Projected gradient step along feasible direction
Feasible directions
(c)
No cell transition required
Old Neighbours
New selected cell

Figure 26 One iteration of the proposed algorithm on two cells of the upper-arm patch. The idle cells are depicted in blue, while the selected cells broadcasting information at the maximum frequency, and their neighbors are depicted in purple and green respectively. The unconstrained gradient step is represented as a red segment and its projection along feasible directions of the problem is depicted in magenta. **(a)**: First the neighbors of each selected cells are extracted and a classic gradient step of the unconstrained problem is computed. **(b)**: The unconstrained gradient step is then projected along each feasible direction. The highest norm projection is then selected. **(c)**: If the norm of this gradient projection is greater than a heuristically defined threshold ζ, then a cell-switch event is triggered with the corresponding neighbor cell along the selected direction.

From an initial random sensor distribution of M_i elements on the robot link i, we proceed by first computing the gradient $\nabla_{\boldsymbol{z}_i} f(\boldsymbol{z}_i)$ of the value function $f(\boldsymbol{z}_i)$ in terms of the selected sensors' Cartesian coordinate vectors $\boldsymbol{z}_i \in \mathbb{R}^{3 \cdot M_i}$, relative to the link frame $\mathcal{L}_i$. The following gradient step is then derived for each selected sensor k, regardless of the geometrical constraints of the patches (c.f. Fig. 26.a):

$$\hat{\boldsymbol{z}}_i^k = \boldsymbol{z}_i^k - \eta \nabla_{\boldsymbol{z}_i^k} f(\boldsymbol{z}) \tag{149}$$

where $\eta \in \mathbb{R}_+$ is a tuning parameter. This unconstrained gradient step is then projected onto the constrained Cartesian subspace $\mathcal{E}_i^k$ of the problem. We define the subspace $\mathcal{E}_i^k$ in a cell-wise manner, as the set of unit-length segments directed along the cell-to-neighbor vectors $\boldsymbol{z}_i^{k,n_k} = \boldsymbol{z}_i^{n_k} - \boldsymbol{z}_i^k$ as depicted in Fig. 26.b. For each selected cell, the neighbor $n_k^\star$ of cell k with the maximal projection value will be labeled as *candidate*, to be considered for the next algorithm iteration:

$$n_k^\star = \underset{n_k}{\arg\max}(\mathrm{Proj}_{\mathcal{E}_i^k(n_k)}(\hat{\boldsymbol{z}}_i^k - \boldsymbol{z}_i^k)) \tag{150}$$

The different candidates are then filtered such that:

1. only the candidates whose projected norm is greater than a predefined threshold $\rho \in \mathbb{R}_+$ are declared valid and used (c.f. Fig. 26.c),
2. the candidates whose index belong to the set of "unusable cells", namely the sensors that are identified as malfunctioning, undergoing physical interaction[4], or that are already selected in the current iteration are automatically discarded. The vector containing these indexes is automatically updated at each control period.

At the end of these two filtering steps, the remaining cell candidates define the new selected sensor set. The whole process is repeated until convergence, when the vector of filtered

[4] This work identifies unusable cells by exploiting the multi-modality of the skin, making use of the proximity and force sensors to determine precontact and contact conditions.

candidates turns out to be empty.

4.5. Experimental Validation

This section covers the details of the simulations and experiments carried out to demonstrate the performance of the different methods proposed in this chapter.

4.5.1. Description of the Motion Estimation Experiment Setup and Protocol

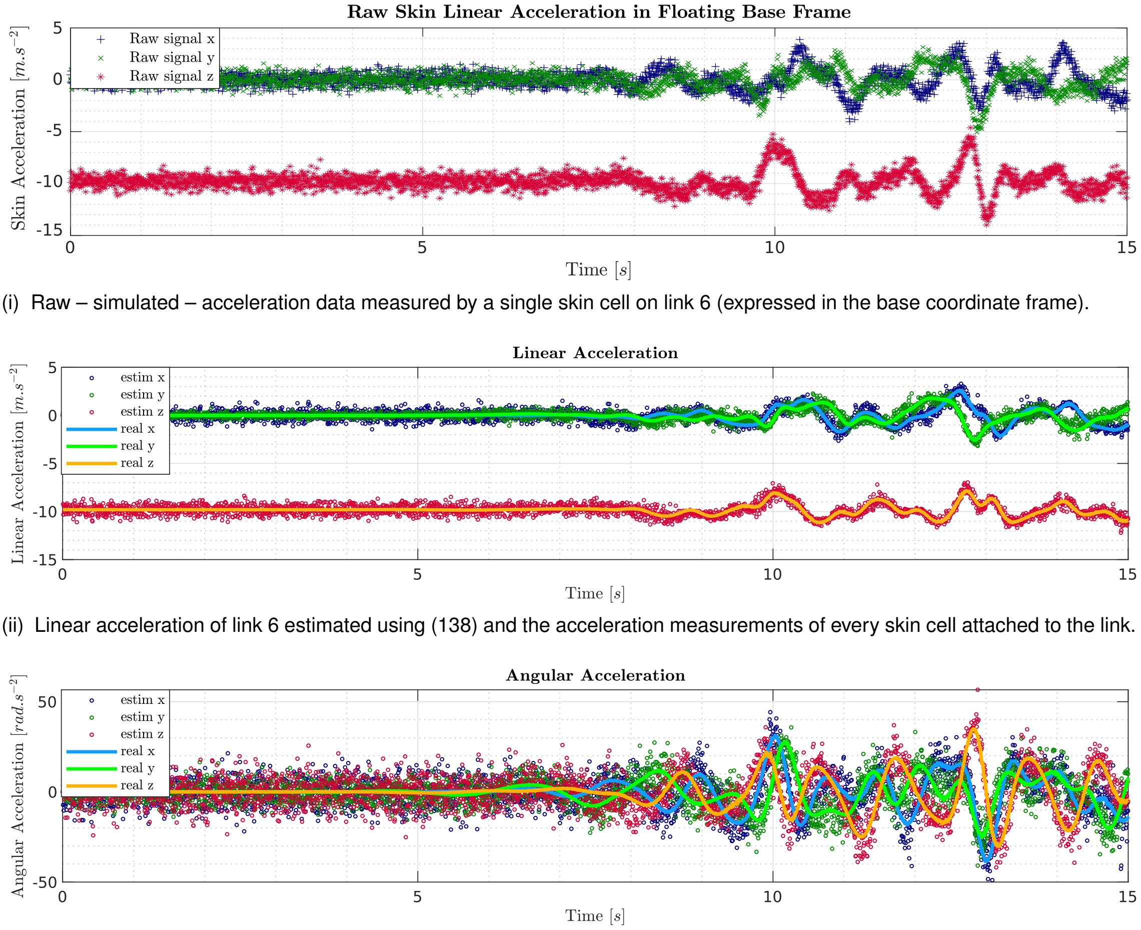
During this work, we executed a set of two distinct experiments to validate our estimation approach, first in simulation and then on a real humanoid robot. The experimental setup is presented in Fig. 28. Our robot is a 30-DoF REEM-C humanoid covered with 1260 cells of multimodal artificial skin [6]. The main inertial measurement unit, located in the floating base, is an industrial-grade IMU (MTi-30-AHRS). During these experiments, the skin cells are supposed to be broadcasting acceleration measurement data at a steady frequency of 250 Hz. This of course raises multiple issues considering the significant amount of information that has to be transmitted on the skin network and processed in real-time by the robot's main computer[5]. To overcome this issue, it was decided to use a subset of 312 skin cells, located on the right arm, torso, and floating base of the robot, respectively, thereby accounting for a serial kinematic chain of nine DoFs. In this work, the measured data is processed *offline*[6] on a desktop Intel i7 computer. This allows the estimated signal to be compared with reference data, obtained using dedicated zero-phase-shift forward-backward filters. Accordingly, the reference joint quantities, to which the estimated signals are being compared in this work, were obtained by filtering of the raw joint position signal, using a third-order zero-phase shift forward-backward Butterworth filter with a cutoff frequency of 15Hz, followed by one (resp. two) central differentiation steps. The zero-delay filter was implemented using the "*filtfilt*" function of MATLAB. The tuning parameters $\alpha = 1 \cdot 10^{-2}, \beta = 2$ and $\kappa = 0$ of the UKF were chosen heuristically, based on the guidelines defined in [76]. For the considered 9-DoF kinematic chain, we have a total of $\boldsymbol{\Theta} \in \mathbb{R}^{51}$, resulting in $2 \times 51 + 1 = 103$ sigma points. The process noise and measurement noise covariance matrices $\boldsymbol{\Sigma_{vv}}$ and $\boldsymbol{\Sigma_{ww}}$ were chosen following the procedure described in [249].

4.5.1.1 Experiment Setup and Protocol: Simulation

A simulation, on the model of the REEM-C humanoid robot right arm and torso evaluates the feasibility of the proposed estimation approach. During this simulation, the right arm and the torso joints of the robot are tracking a predefined trajectory, in the form of sinusoidal signal with a time-dependent amplitude. The raw joint angle data from the simulator are corrupted with a white measurement noise with a standard deviation $\sigma_{encoder} = 1.75 \cdot 10^{-2} rad$ to reproduce the real system's imperfections. Similarly, the cells acceleration data, emulated from the

[5] Note that the neuromorphic event-driven approaches to artificial tactile sensing, such as [267, 268, 269] were not exploited yet and are here considered as future work.

[6] We believe that such a method could be adapted online and run directly on the robot. This is here considered as future work.



(i) Raw – simulated – acceleration data measured by a single skin cell on link 6 (expressed in the base coordinate frame).

(ii) Linear acceleration of link 6 estimated using (138) and the acceleration measurements of every skin cell attached to the link.

(iii) Angular acceleration of link 6 estimated using (138) and the acceleration measurements of every skin cell attached to the link.

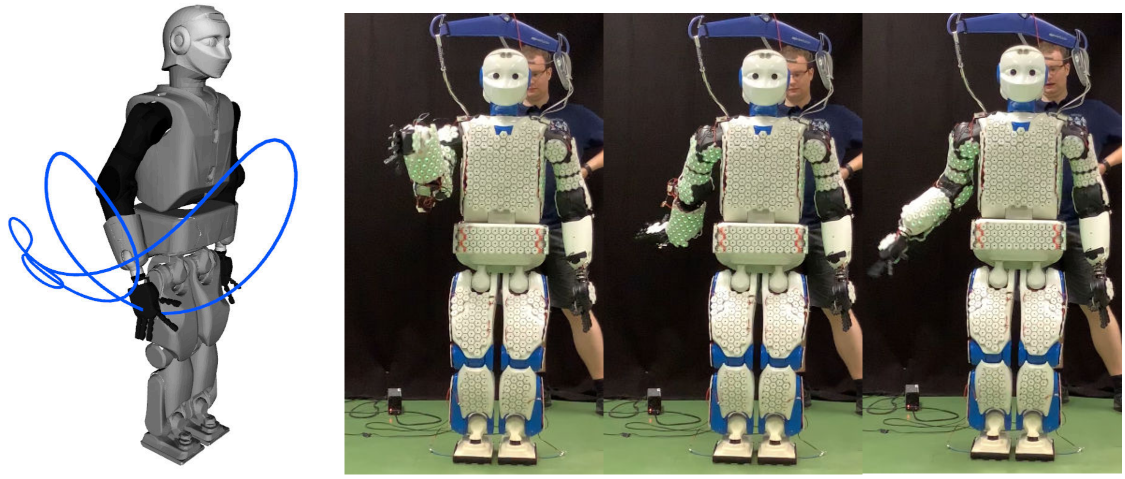
Figure 27 Simulation data

true system's linear acceleration, evaluated at every different cells mounting points, is corrupted by an additive white measurement noise with a standard deviation $\sigma_{cell} = 0.34m.s^{-2}$[7] (c.f. Fig. 27i). For the sake of simplicity, it is here assumed that the skin cells covering the robot are rigidly attached to it, perfectly calibrated and that the floating base of the robot is not moving[8]. From this point on, the estimates of the each link's linear and angular accelerations can be computed using eq. (138) (c.f. Fig. 27ii and Fig. 27iii). These link acceleration terms are then used within the measurement update equation (140) of the Kalman filter to get suitable link velocity and acceleration estimates.

4.5.1.2 Experiment Setup and Protocol: Real Robot

In the experiment with the real robot, the robot is tracking a desired trajectory and continuously broadcasts the data from its encoders, the main IMU, and from the 312 activated skin cells. Data acquisition was performed at a frequency of 200Hz on the robot and 250Hz (sub-sampled at 200Hz) on the skin. The skin accelerometers are set to a range of ± 4g as a compromise between accuracy and sensor saturation during the execution of dynamic behav-

[7] This value was determined from actual sensor measurements.
[8] Note that the last two assumptions will not hold during the experiment performed on the real system.



(i) Simulation (ii) Real experiment

Figure 28 Trajectory followed by the real robot during the validation experiment.

iors by the robot. Under these conditions the sensor can measure acceleration increments of 7.81mg [289]. The desired joint trajectories correspond to a permanent excitation movement such as those encountered in the context of dynamic parameter identification. This trajectory was obtained by the parametrization of the Fourier series as presented in [16]. Note that the robot stabilizer was deactivated during the experiments to produce noticeable movements on the floating base.

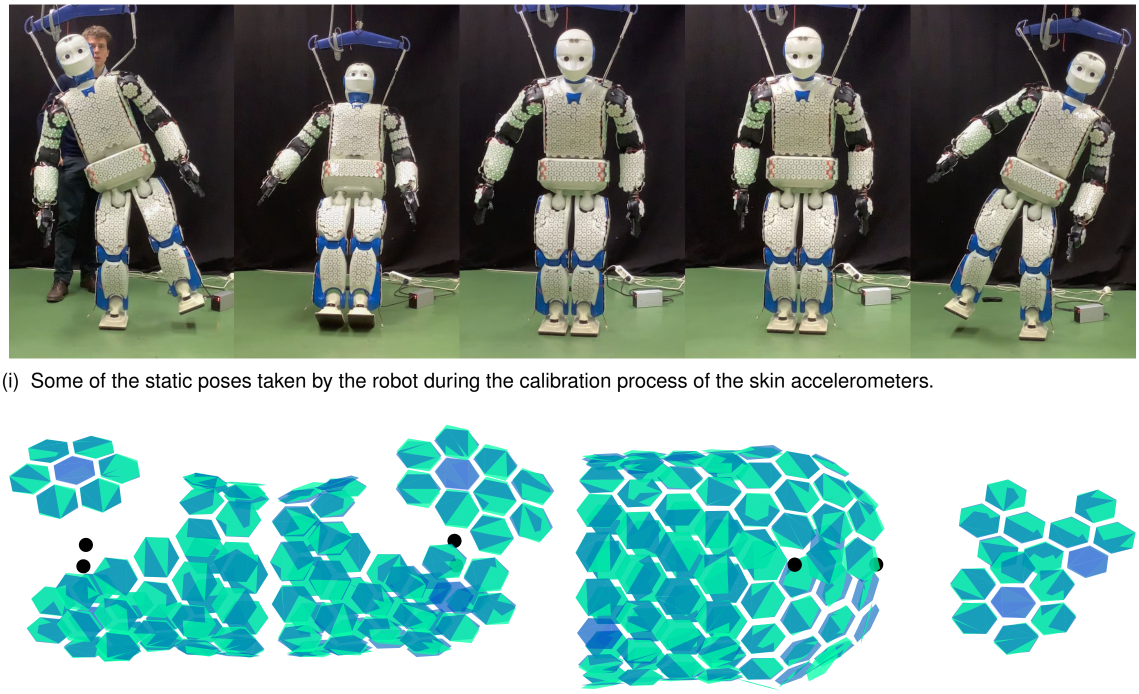
4.5.2. Accelerometer Calibration

Calibration is a major issue in the context of inertial sensing with multiple contributions reported in the scientific literature [290, 291, 292, 249]. As a matter of fact, the measurements of MEMS accelerometers are usually affected by multiple sources of errors, which can be regrouped into two main categories, namely the *manufacturing errors* (e.g. sensitive axis misalignment, bias) and the *application errors* (e.g. sensor placement or orientation error). Considering that a stationary accelerometer should only be measuring the gravity vector, sampling the measurements $\boldsymbol{a}_{calib}$ of a calibrated system on multiple different poses should result into a spherical measurement distribution with a radius 9.81m.s^{-2} or 1g. Conversely, in an uncalibrated system, such a measurement point cloud will exhibit the shape of a shifted-rotated ellipsoid:

$$(\boldsymbol{a}_{raw} - \boldsymbol{b})^\top \underbrace{\boldsymbol{A}_{fit}}_{ellipsoid} (\boldsymbol{a}_{raw} - \boldsymbol{b}) = 1 \tag{151}$$

where $\boldsymbol{b}$ is a bias term. In other words, the accelerometer calibration problem can be considered as an ellipsoid fitting problem, or determining the transformation matrix $\boldsymbol{M} \in \mathcal{GL}(3)$ such that:

$$\underbrace{(\boldsymbol{a}_{raw} - \boldsymbol{b})^\top \boldsymbol{M}^\top}_{\boldsymbol{a}_{calib}^\top} \underbrace{\boldsymbol{I}}_{sphere} \underbrace{\boldsymbol{M}(\boldsymbol{a}_{raw} - \boldsymbol{b})}_{\boldsymbol{a}_{calib}} = 1 \tag{152}$$



(i) Some of the static poses taken by the robot during the calibration process of the skin accelerometers.

(ii) Results of the orientation calibration phase executed on the right arm of the REEM-C robot: the uncalibrated cells, depicted in light green, are superimposed with the calibrated cells, in blue.

Figure 29 Sensor calibration process and results.

The measurement ellipsoid of each sensor can be estimated by sampling methods, for instance by performing a series of static gravity acceleration measurements in a set of different positions. In our case, a total of 16 different poses were considered, sampling the acceleration at 250 Hz for 10 seconds in each pose and for each sensor (c.f. Fig. 29i). Determining $\boldsymbol{M}$ can then be carried out using a singular value decomposition:

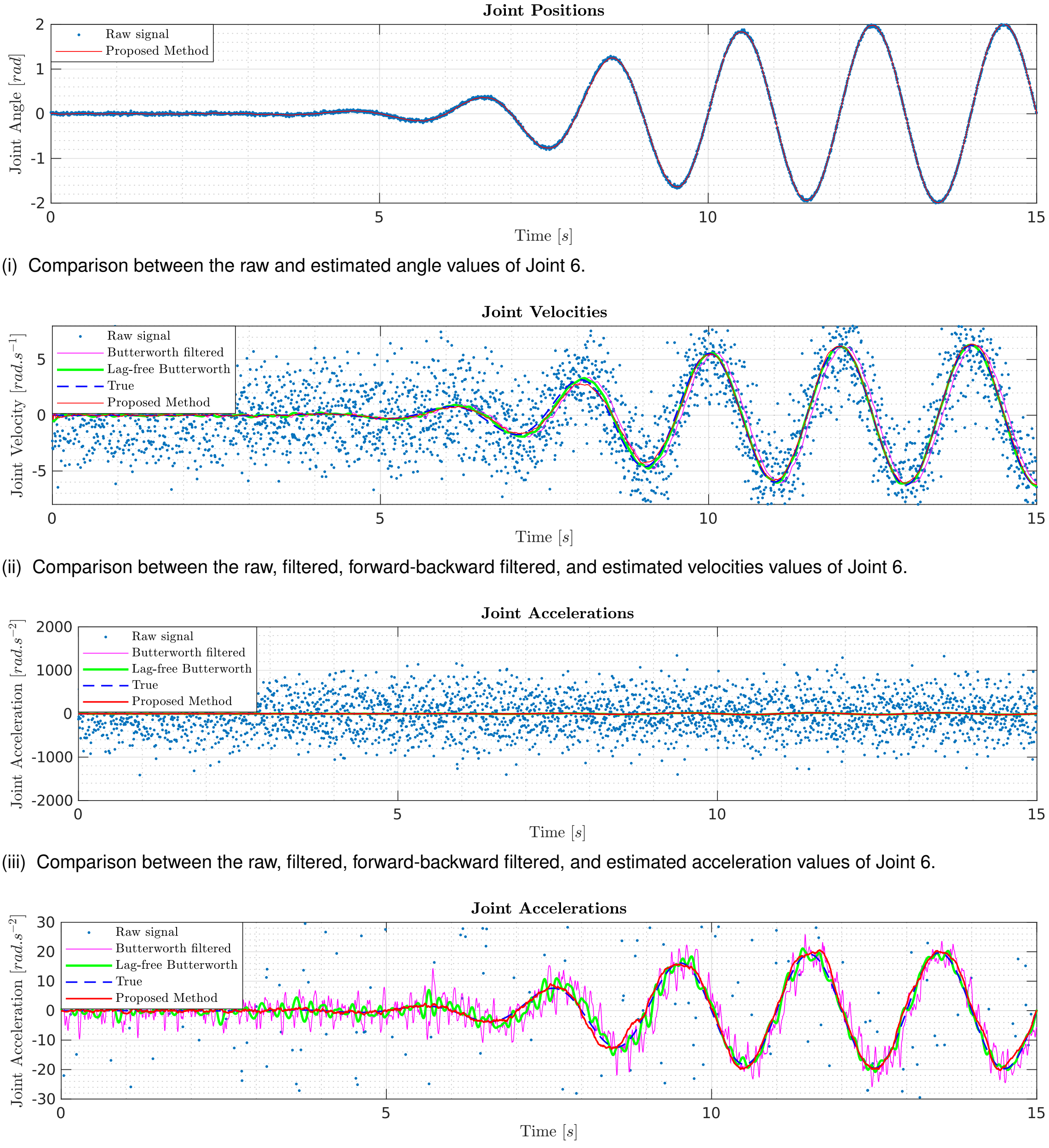
$$\boldsymbol{M} = \sqrt{\boldsymbol{\Lambda}}\boldsymbol{Q}^\top, \boldsymbol{A}_{fit} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^\top \tag{153}$$

Finally the calibrated acceleration can be expressed as $\boldsymbol{a}_{calib} = \boldsymbol{M}\left(\boldsymbol{a}_{raw} - \boldsymbol{b}\right)$.

4.5.3. Motion Estimation: Results and Discussion

4.5.3.1 Simulation Experiments

Part of the results of the simulation experiments are shown in figure 30 for the sixth robot joint (i.e. the elbow joint). On this figure, the raw noisy quantities are depicted as blue dots, the Butterworth-filtered values are shown in magenta, the zero-shift Butterworth-filtered values are shown in green, the true reference values from the simulator are shown in dashed blue (on the last three graphs) and finally the estimated values from the proposed UKF method are depicted in red. The results on Fig. 30 suggest that the proposed estimator reliably tracks the robot motion as well as its first and second order time derivatives. It is in particular interesting to note that this tracking is carried out with negligible phase shift, especially compared to the classic Butterworth filter approach. This is in particular visible on Fig. 30ii, as the proposed estimated signal nearly overlaps the reference and zero-shift Butterworth filtered signals. At



(i) Comparison between the raw and estimated angle values of Joint 6.

(ii) Comparison between the raw, filtered, forward-backward filtered, and estimated velocities values of Joint 6.

(iii) Comparison between the raw, filtered, forward-backward filtered, and estimated acceleration values of Joint 6.

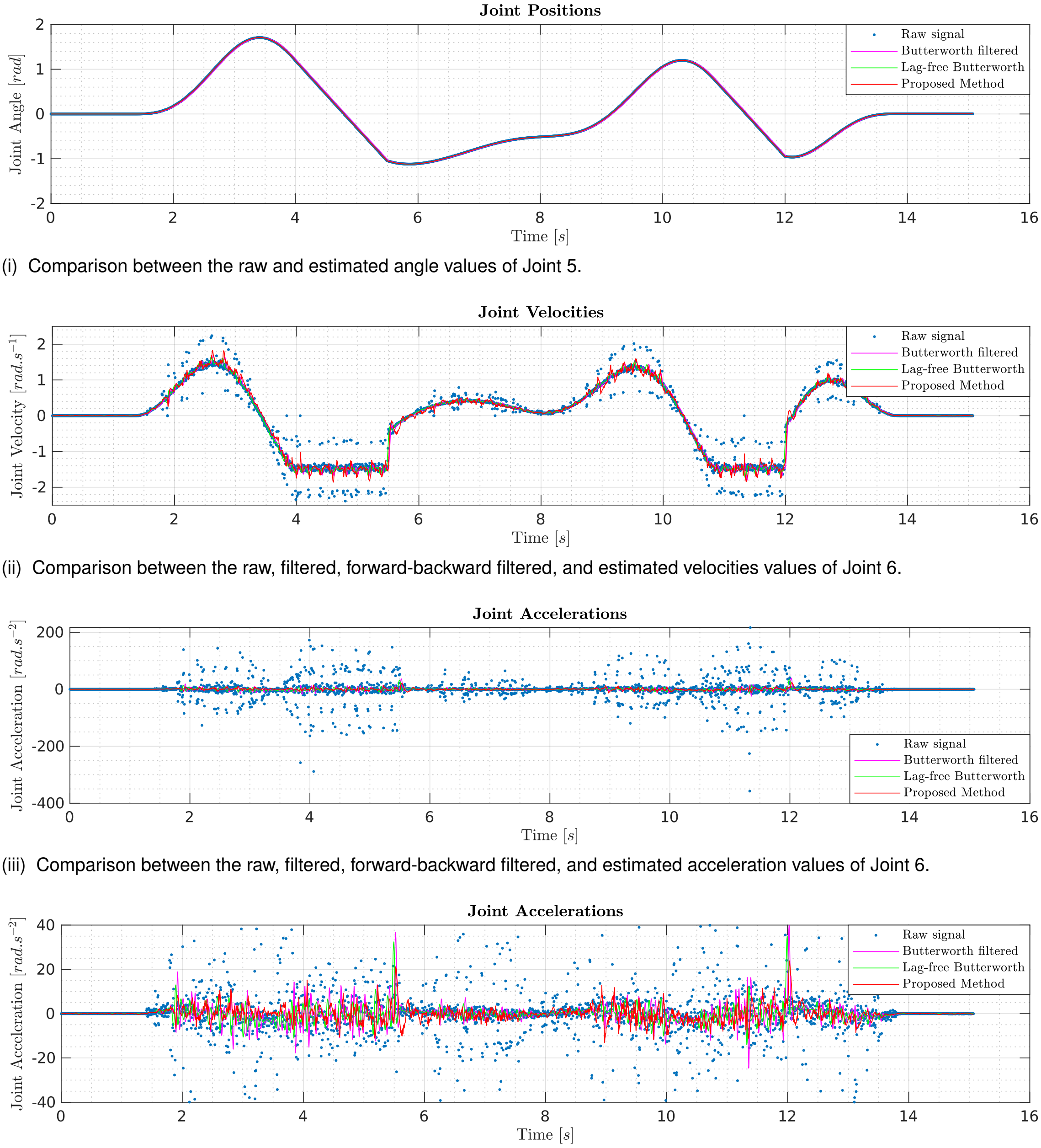
(iv) Comparison between the raw, filtered, forward-backward filtered, and estimated acceleration values of Joint 6 (zoomed).

Figure 30 Simulation results, joint 6 (elbow).

the acceleration level, the proposed approach appears to exhibit a better signal-to-noise ratio than the Butterworth filtered signals (c.f. Fig. 30iii and 30iv), although some deviations from the reference signal (in dashed blue) should be highlighted, such as for instance at $t = 9.5s$ or $t = 11.5s$. In the current state of this research, we believe these distortions are a consequence of an ill-tuned process noise covariance matrix.

4.5.3.2 Validation Experiments on the Real Robot

A subset of the experimental results obtained from the real robot data is presented in Fig. 31 for robot joint 5. As previously, the raw joint quantities are depicted in blue, the Butterworth-filtered values are shown in magenta, the zero-shift Butterworth-filtered values in green and



(i) Comparison between the raw and estimated angle values of Joint 5.

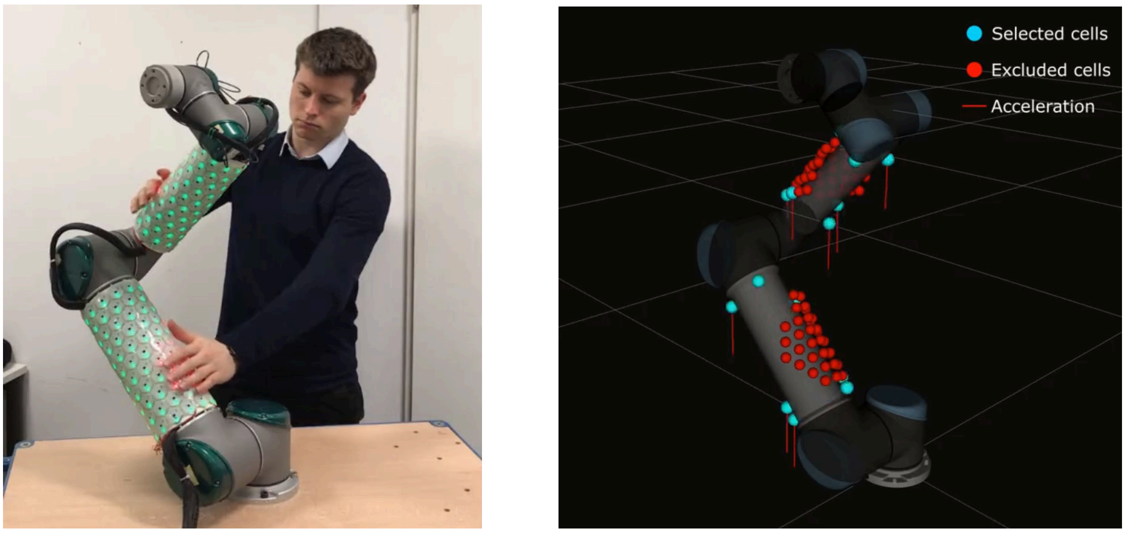
(ii) Comparison between the raw, filtered, forward-backward filtered, and estimated velocities values of Joint 6.

(iii) Comparison between the raw, filtered, forward-backward filtered, and estimated acceleration values of Joint 6.

(iv) Comparison between the raw, filtered, forward-backward filtered, and estimated acceleration values of Joint 6 (zoomed).

Figure 31 Experiment results, joint 6 (elbow).

finally the estimated values from the UKF state vector are depicted in red. From Fig. 31i, it is possible to observe that the joint angle estimator reliably tracks the encoder signal, although very little distortion can be noticed during sudden changes in the velocity as it is, for example, the case at $t = 5.5s$ and $t = 12s$. This is consistent with the very low standard deviation of the joint encoder noise on most electrically actuated robots ($\sigma_{encoder} = 1.7453 \cdot 10^{-05} rad$ on REEM-C). One of the interesting features in Fig. 31ii is the joint velocity saturation occurring from $t = 4.0s$ to $t = 5.5s$ and from $t = 10.9s$ to $t = 12.0s$. This is characterized by spikes in the acceleration signal (c.f. Fig. 31iv), the most noticeable of which are visible at $t = 5.5s$ and $t = 12s$, when the velocity signal changes abruptly. Although these peaks are clearly visible on the UKF acceleration estimate, on the lower figure, one can notice that they are



(i) Experiment setup on the UR5 robot

(ii) Selected cells are shown in cyan and excluded cells in red.

Figure 32 Experiment on the real UR5 robot covered with multimodal artificial skin

slightly delayed compared to the lag-free Butterworth estimate. This delay is estimated to $20ms$, which corresponds to 10 control periods. One can also notice some oscillations in the acceleration estimate, right after the spikes, although the Butterworth reference signal does not exhibit such phenomenon. This eventually results in several signal overshoots, in the UKF velocity estimate, on the middle plot, in particular at $t = 5.5s$ and $t = 12s$. Several hypotheses are currently being investigated to explain these observations. After ensuring that this delay was not the result of a lag into the measurements, our current guess is that this phenomenon is the result of an underestimated process noise covariance on the jerk model. Notwithstanding this, it should be noted that the UKF velocity estimate tracks the reference forward-backward filtered signal with equivalent noise levels and a phase shift about five times smaller than the conventionally low-pass filtered (magenta) signal provided by the manufacturer. Long term drift analysis of the estimate was not carried out in this thesis and is here considered as future work.

4.5.4. Description of the Sensor Selection Validation Setup and Protocol

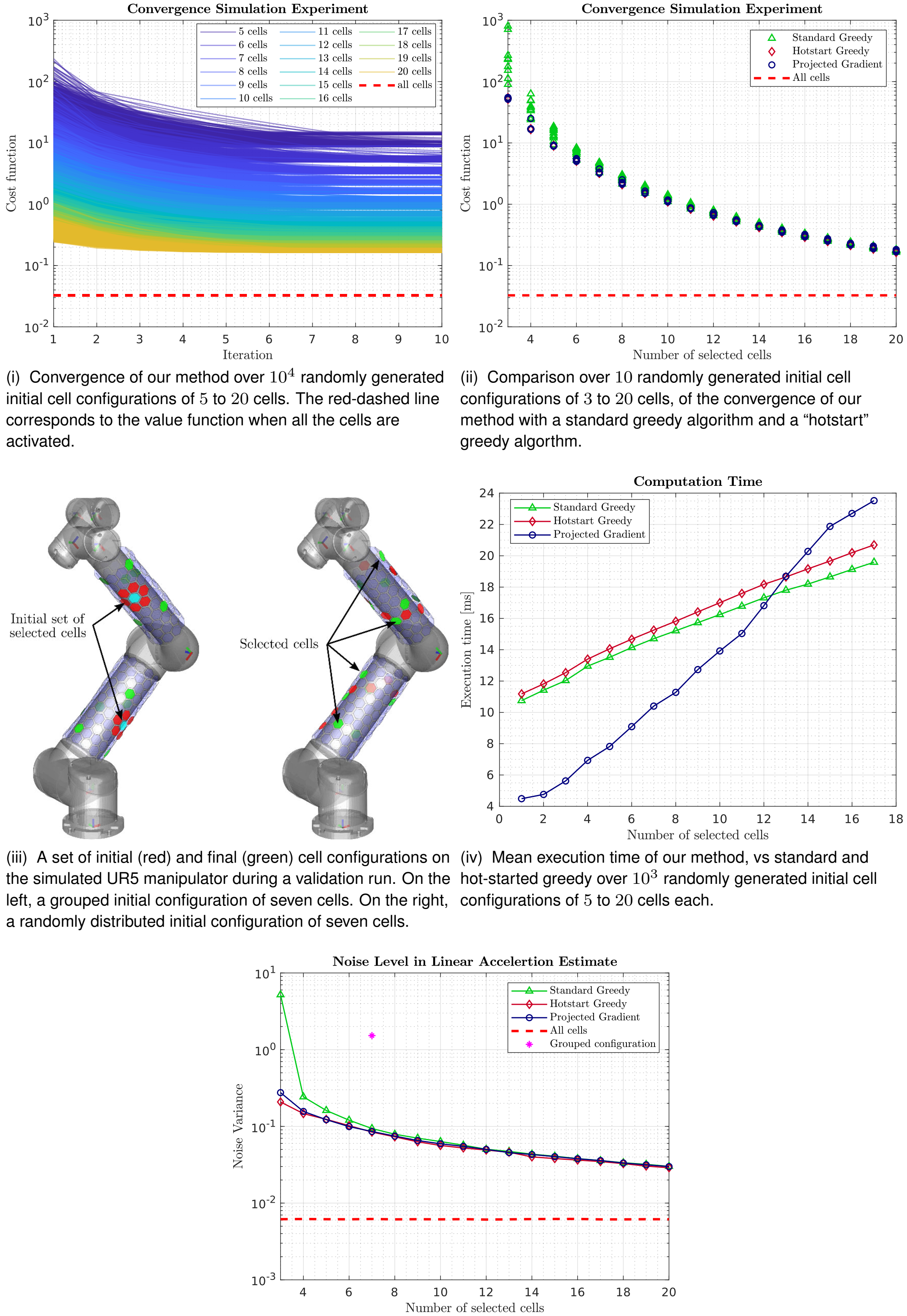
The validation setup for these experiment is presented in Fig. 24 and Fig. 32. We consider a 6-DoF UR5 industrial robot arm from Universal robotics, covered with two patches of artificial skin, accounting for a total of 253 cells. Cell positions and orientations were determined using calibration data, obtained from the actual skin system mounted on a real UR5. We used the self-configuration procedure detailed in [293]. The proposed validation protocol consists of three main steps. We first analyze the convergence properties of our method in simulation and compare them to that of a greedy algorithm by executing a Monte Carlo Simulation (MCS) on a set of randomly generated initial cell configurations ranging from 3 to 20 cells. Note that since a set of at least three cells is required to get reliable acceleration estimates using eq. (138), the greedy algorithm has to be initialized with a non-empty cell distribution, which, in this work, is either randomly generated or computed using the proposed projected gradient algorithm from a random initial set. Here the randomly initialized greedy is referred to as "*standard*" while the projected-gradient-initialized greedy is referred to as "*hot-started*". The second validation step aims at demonstrating – once again in simulation – the performance gain, in term of measurement noise mitigation, along a predefined trajectory executed by a simulated UR5 robot. The fact that this phase is executed in simulation is here justified by the need of a noiseless link acceleration reference and of perfectly-calibrated acceleration sensors, for suitable assessment of the estimation quality.

The simulated UR5 robot tracks a preset trajectory, defined as a weighted sum of sinusoids. Two sets of simulated skin cells, rigidly coupled to the robot body, measure the 3D acceleration along this trajectory. As for actual accelerometers, a normally distributed measurement noise with standard deviation $\sigma = 0.34 \cdot \boldsymbol{I}_{3\times3}$[9] is added to the acceleration measurements of each simulated skin cell. Estimates of the link linear and angular accelerations are then computed using eq. (138) and the skin acceleration data acquired with different cell numbers and configurations. In practice, these configurations are generated, on the one hand, using our method and, on the other hand, using the standard and hot-started greedy algorithms. We compare the estimated link accelerations to the reference accelerations computed using the closed form expressions of the robot second order kinematics and use the variance of the difference between these two signals as a noise-level metric. The final validation step consists in implementing the sensor selection algorithm in C++ and testing it on the real UR5 robot (c.f. Fig. 32). In every case the cost function coefficients are set to $k_1 = 10^{-4}$ and $k_2 = 1$ to emphasize the noise attenuation characteristics while penalizing badly conditioned configurations, verifying $\mathrm{cond}(\boldsymbol{\Sigma}_i^{-1}(\boldsymbol{z}_i)) >> 1$.

4.5.5. Sensor Selection: Results and Discussion

The results of the convergence experiment are displayed in Figs. 33i, 33ii and Fig. 33iv. From Fig. 33i, it is possible to see that convergence of the proposed algorithm is achieved in less than ten iterations. Nevertheless Fig. 33i reveals a non-negligible spread in the final cost values for a given number of cells, thereby suggesting a lack of robustness of the proposed algorithm to local minima. Note that the effects of unusable cells does not noticeably appear in this study and would deserve deeper treatment. This is here considered as future work. Fig. 33ii suggest that the quality of the resulting optimizer exceeds that of the standard greedy for cell numbers ranging from three to fifteen. This can be explained by the random nature of the initial cell distribution used in this algorithm. It is interesting to note that the performances of our algorithm are similar to those of the greedy algorithm when the latter is hot-started with an optimized initial distribution. One of the main advantages of this thesis' method lies in its execution time. This is clearly visible in Fig. 33iv, where we can see that our algorithm is up to two times faster than greedy algorithms for sets of three to thirteen cells, which generally corresponds to the number of sensing elements used on the links of a robot, as it offers a good compromise between network load and measurement accuracy. The mean

[9] This value was determined from actual sensor measurements.



(i) Convergence of our method over 10^4 randomly generated initial cell configurations of 5 to 20 cells. The red-dashed line corresponds to the value function when all the cells are activated.

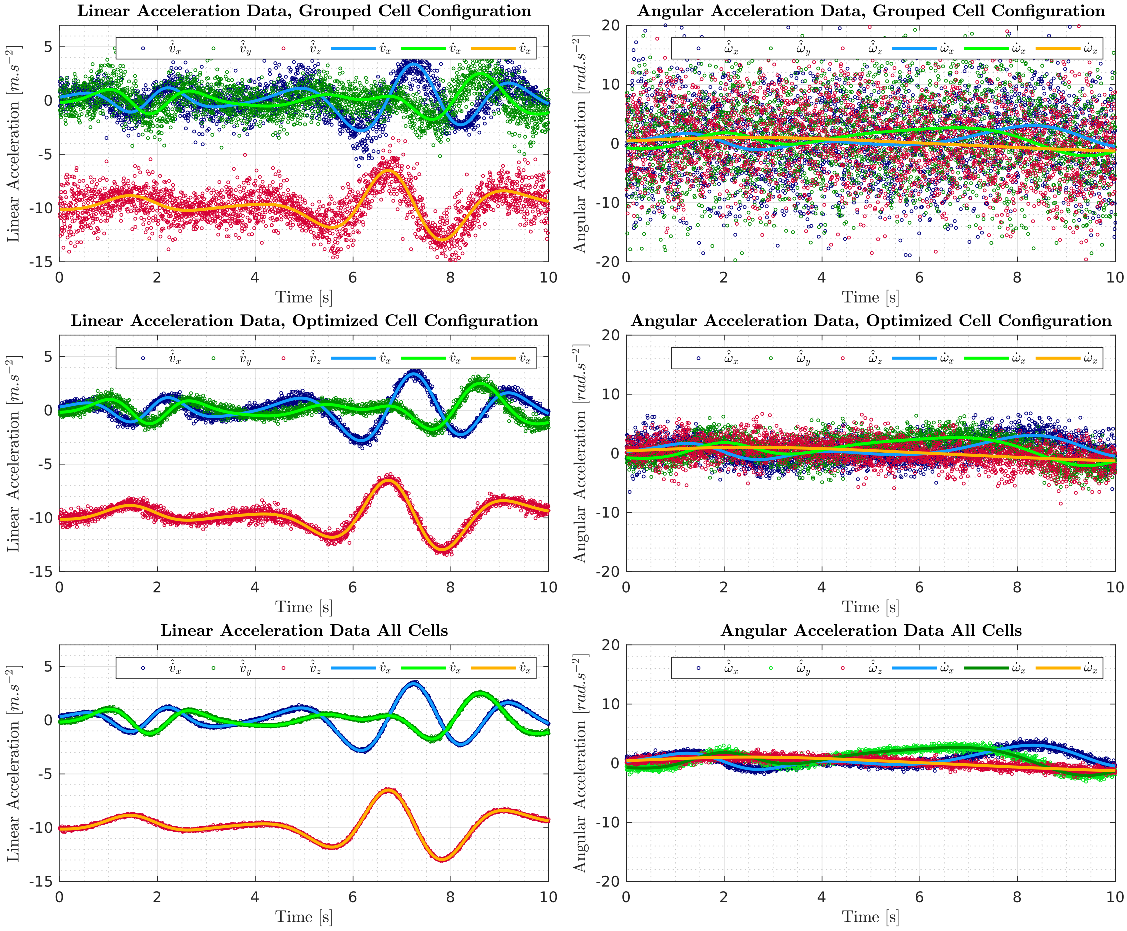
(ii) Comparison over 10 randomly generated initial cell configurations of 3 to 20 cells, of the convergence of our method with a standard greedy algorithm and a “hotstart” greedy algorthm.

(iii) A set of initial (red) and final (green) cell configurations on the simulated UR5 manipulator during a validation run. On the left, a grouped initial configuration of seven cells. On the right, a randomly distributed initial configuration of seven cells.

(iv) Mean execution time of our method, vs standard and hot-started greedy over 10^3 randomly generated initial cell configurations of 5 to 20 cells each.

(v) Noise variances obtained with multiple optimized cell configurations over a 10-sample MCS.

Figure 33 Convergence results



(i) Comparison between the raw, estimated and forward-backward filtered angle values of Joint 5.

(ii) Comparison between the raw, estimated and forward-backward filtered velocity values of Joint 5.

Figure 34 Linear accelerations of the second link of a simulated UR5 manipulator. The estimated values $\hat{\dot{v}}_x, \hat{\dot{v}}_y, \hat{\dot{v}}_z$ computed using eq. (138) and the measurements of 7 skin cells in three different configurations, namely *grouped* (c.f. Fig .33iii), *optimized* and *full-set*, are compared to the reference values $\dot{v}_x, \dot{v}_y, \dot{v}_z$.

convergence time is evaluated to 10.5ms on Matlab for 7-cells in a 144 cells patch, which is compatible with online use, considering in particular that for loops run much faster in C++. Execution of our sensor selection algorithm on an actual skin system, mounted on a real UR5 robot proved its reactivity and ability to run online, at the robot control interface frequency (i.e. 125Hz on the UR5). The results of the noise mitigation experiment are displayed in Fig. 34i and Fig. 34ii in terms of the linear and angular accelerations measured on the second link using eq. (138). The top graph shows in particular the consequences of an ill-selected cell configuration, referred to as “grouped” and visible in Fig. 33iii. The noise variance in this case is depicted as a magenta star marker on Fig. 33v. The benefits of the proposed cell selection algorithm in terms of measurement noise attenuation are clearly visible in the middle plot of Fig. 34i and in Fig .34ii. Again, it is worth noting the similarity of these results with those of the greedy algorithms.

4.6. Summary

In this chapter, we proposed a new derivation-free estimation method for second-order kinematics on floating-base robots along with a new method for online sensor selection. Relying on highly-redundant distributed acceleration measurement from an artificial sensory skin covering our robot, our approach to motion derivative estimation is two-fold. The first step consists in fusing the skin accelerations link-wise using a dedicated Maximum-Likelihood observer. The second step then involves fusing these measurements with that of the encoders and floating-base IMU. The results show that the motion derivative estimates have equivalent noise levels to those of the reference signals while maintaining a lower lag. Future work will consist, on the one hand, of extending the developed method to an entire humanoid robot, thereby verifying whether it can be executed on-line and, on the other hand, of evaluating the propensity of the estimated acceleration signal to be used in a high-gain control loop. Relying on a projected gradient heuristics, the proposed sensor selection method is capable of converging in less than ten iterations to new inertial sensor configurations that significantly reduce noise levels in the obtained acceleration estimates.

Chapter 5

Quantitative Analysis and Comparison of Inertial Parameter Identification Algorithms

This chapter aims at reviewing, analyzing and comparing a range of state-of-the-art approaches to inertial parameter identification in the context of robotics. Although inertial parameter identification is a widely explored research topic, with multiple contributions, there is currently no clear consensus as to the context of application of a given identification method depending on the platform, available sensory modalities or more generally, to the end purpose of the identification process. We introduce "*BIRDy*[1]" an open-source Matlab toolbox, allowing a systematic and formal performance assessment of the considered identification algorithms on either simulated or real serial robot manipulators.

Seventeen of the most widely used approaches found in the scientific literature are implemented and compared to each other, namely: the Inverse Dynamic Identification Model with Ordinary, Weighted, Iteratively Reweighted and Total Least-Squares (IDIM-OLS, -WLS, -IRLS, -TLS); the Instrumental Variables method (IDIM-IV), the Maximum Likelihood (ML) method; the Direct and Inverse Dynamic Identification Model approach (DIDIM); the Closed-Loop Output Error (CLOE) method; the Closed-Loop Input Error (CLIE) method; the Direct Dynamic Identification Model with Nonlinear Kalman Filtering (DDIM-NKF), the Adaline Neural Network (AdaNN), the Hopfield-Tank Recurrent Neural Network (HTRNN) and eventually a set of Physically Consistent (PC-) methods allowing the enforcement of parameter physicality using Semi-Definite Programming, namely the PC-IDIM-OLS, -WLS, -IRLS, PC-IDIM-IV, and PC-DIDIM. BIRDy is robot-agnostic and features a complete inertial parameter identification pipeline, from the generation of symbolic kinematic and dynamic models to the identification process itself. This includes functionalities for excitation trajectory computation as well as the collection and pre-processing of the experiment data. In this work, the proposed methods are first evaluated in simulation, following a Monte Carlo scheme on models of the 6-DoF TX40 and RV2SQ industrial manipulators, before being tested on the corresponding real robot platforms. The robustness, precision, computational efficiency and context of application of the different methods are investigated and discussed. This chapter is organized as follows: Section 5.1 introduces the issue of robot identification and discusses the state of the art. Section 5.2 then describes the proposed identification benchmark, emphasizing its structure and the adopted conventions. The different experiments, and their results, are presented in Section 5.3 and discussed in Section 5.4. Finally, Section 5.5 provides a brief conclusion and opens perspectives for further developments of BIRDy.

[1] **BIRDy** stands for **B**enchmark for **I**dentification of **R**obot **Dy**namics.

5.1. Benchmarking Identification Algorithms

5.1.1. Motivation and Related Works

As adaptive, predictive and passivity-based control strategies, relying on explicit formulations of system dynamics, become increasingly popular in robotics, substantial research efforts are being made in the field of model identification to improve the performance, robustness and adaptivity of these methods, see for example [294] and the references therein. In the context of rigid robots, techniques for identifying inertial parameters are generally based on a rigorous analysis of joint movements and torque signals over a predefined time horizon. A given robot is tracking a trajectory that *excites* the different components of its dynamic model and the residuals in term of motion and torque are used to refine the parameters estimates. The most common approach to offline or batch dynamic parameter identification is the Inverse Dynamic Identification Model with Ordinary Least-Squares estimation (referred to as IDIM-OLS in [11, 295]). This method is based on the assumption that the mapping between joint torques and the inertial parameters of a robot is linear[2]. Both efficient and easy to implement, IDIM-OLS and its variants – including Weighted Least-Squares (IDIM-WLS) and Total Least Squares (IDIM-TLS) methods described in [43] – however lack noise immunity and exhibit a strong dependence on the conditioning of the robot's trajectory, as depicted in [66, 29]. As a matter of fact, whether due to measurement noise, incorrect data filtering or insufficient excitation, IDIM-OLS parameter estimates may eventually prove to be highly biased, to the point of losing all physical coherence, yielding, for example, negative link masses, friction coefficients, or non-positive-definite rotational inertia.

Over the past two decades, much research effort has focused on solving these issues and multiple promising methods were proposed. These approaches can generally be divided into two distinct classes, whether the objective is to improve the statistical consistency of the estimates – physical consistency naturally stemming from statistical consistency[3] – or to more explicitly enforce physicality by means of dedicated constrained optimization techniques. In both cases, performance evaluation is generally carried out through direct comparison with unconstrained IDIM-OLS or IDIM-WLS. For instance, [38] compared the performance of the IDIM-WLS method with that of an EKF estimator, on a 2 degree of freedom (DoF) SCARA robot. The results indicate that unlike IDIM-WLS, the EKF estimator has convergence issues and exhibits a strong dependence to both initial conditions and tuning parameters. In [296] the authors presented an approach based on Set Membership Uncertainty (SMU). This approach was tested on a 2 DoF SCARA robot and compared to IDIM-OLS but did not show significant improvement. In [16, 60, 40], the authors compared IDIM-WLS with a Maximum Likelihood (ML) estimator on 2- and 3 DoF robots. The results suggest that ML techniques may reduce the bias of the parameter estimates in case of noisy joint measurements, at the price however, of a much greater computational effort. Interestingly, only a few studies have

[2] This holds provided that the robot links are rigid, that friction non-linearity is negligible and that the joints are not subject to backlash.

[3] Although the opposite is not true.

actually performed a formal comparison between more than two identification approaches at the same time, and on robots with a large number of degrees of freedom, such as industrial manipulators or humanoids. In [66, 28], the authors used a 6 DoF TX40 industrial robot manipulator to compare the Direct and Inverse Dynamic Identification Model (DIDIM) method, the Closed Loop Output Error (CLOE) method, the Closed Loop Input Error (CLIE) method and the IDIM-WLS. In [29] the same research team compared the Instrumental Variable approach (IDIM-IV) with IDIM-WLS, the Total Least Squares approach (IDIM-TLS) and CLOE. The results suggest that DIDIM, IDIM-IV, CLOE and CLIE are significantly less sensitive to noise than Least Squares approaches. The authors moreover enhanced the fact that DIDIM and IDIM-IV require much lower computational effort compared to CLOE and CLIE due to the reduced number of model evaluations. However, it is worth noting that DIDIM and IDIM-IV were not directly compared to each other. In [81, 72], the authors compared five identification methods, namely the IDIM-OLS, Adaline neural networks, Hopfield neural networks, EKF and genetic algorithms, to a 5-DoF SCARA robot. Among these five methods, only the IDIM-OLS and the Adaline neural network gave results that were accurate enough to be exploited. The results also suggest that other methods may lack regularity, as they do not converge for all the considered parameters. Recently, several works highlighted the possibility of explicitly enforcing physical consistency by means of constrained optimization techniques, rather than implicitly, through statistical consistency. With some notable exceptions (e.g. [93, 94, 20, 297]), the classic approach to physically consistent parameter identification consists in expressing physicality constraints in the form of a Linear Matrix Inequality (LMI), thereby allowing to reformulate the whole identification process as a constrained optimization problem, solvable using Semi-Definite Programming (SDP) methods, as for example proposed in [25, 98, 99, 21], or using optimization on manifolds as proposed in [97, 104, 103]. Note that although several approaches such as [298, 299], use Deep Neural Networks (DNN) to learn the full robot dynamics, these contributions will not be considered in this work, as the mapping between DNN hyper-parameters and robot dynamic parameters is not yet fully understood, thereby making a comparison with conventional identification methods irrelevant.

5.1.2. Problem Formulation and Proposed Contributions

Although it seems possible from the previous studies to infer specific tendencies as to the performance and context of application of the different identification algorithms [122], it is worth noting that no study currently provides a general guideline, based on quantitative arguments, such as encoder noise level, sampling frequency or knowledge of the control law. To be valid, such a study should be carried out within the same framework, on a wide range of methods, under well-defined experimental conditions and using a Monte Carlo analysis scheme to provide statistically significant results. Although benchmarking is a common practice in the automatic control community – see e.g. [300, 301, 302] among others – it is less common in robotics, with some notable exceptions [303, 304, 305]. This may be explained by the fact that researchers in robotics are used to deal with real-world data. In practice, evaluating the performance of an identification method on a real system proves to be a difficult task, since the latter can never be perfectly modeled. This is a direct consequence of the wide variety of complex physical phenomena – such as backlash or non-linear friction – that have a significant influence on the system behavior while being relatively challenging to model. On the contrary, the identification of a simulated system allows a rigorous quantification of the performance of an algorithm, since the characteristics and parameters of the simulated model are well known and can therefore be used as a reference. Simulation also makes it possible to highlight the influence of specific physical phenomena – e.g. joint friction or measurement noise – on the quality of the estimates. Therefore, our first contribution is to propose a dedicated framework, in the form of an open-source Matlab toolbox, that makes it possible to formally evaluate the performance of multiple identification algorithms on the same robot model and under the same conditions or assumptions. We use this benchmark to formally evaluate the performance of a set of algorithms among the most widely used off-line computational approaches to robot dynamic parameter identification. These methods are the IDIM-LS (including the WLS, IRLS and TLS variants), ML, IDIM-IV, Output-Error methods (CLOE, CLIE, DIDIM), the Extended Kalman Filter (EKF) as well as several of its Sigma-Point and Square-Root alternatives (SPKF), the Adaline Neural Network (AdaNN), the Hopfield-Tank Recurrent Neural Network (HTRNN) and finally a set of Semi-Definite Programming (SDP) approaches with Linear Matrix Inequalities to enforce physical consistency (PC-IDIM-OLS, PC-IDIM-WLS, PC-IDIM-IRLS, PC-DIDIM). Though it is always interesting to formally compare different methodologies with each other, the usefulness of such a comparison remains questionable if no conceptual relationship can ultimately be established between these approaches. For instance, stating that IDIM-IV or DIDIM outperforms IDIM-OLS in the case of improper data filtering and/or too noisy data is not really helpful nor constructive for practitioners since this gain of robustness against noises is expected from IDIM-IV and Output-Error approaches. Instead it is more interesting to show how the data filtering *helps* IDIM-OLS to provide results that match those provided by IDIM-IV and Output-Error approaches. Our second contribution is therefore to establish some relationships between different methods in order to emphasize their similarities and differences. Finally, establishing relationships allows providing some *guidelines* to practitioners non-experts in robot identification in order to help them to choose an identification method among all the available approaches: this is the third and last contribution of this work.

5.2. Introducing the BIRDy Matlab Toolbox for Identification and Quantitative Performance Assessment

In this work, we conduct a rigorous and systematic performance analysis of the most popular identification algorithms described in chapter 2. This analysis is carried out within a single framework and on the same robot models, following a Monte Carlo process to ensure the statistical relevance of our results. Our main objective is ultimately to provide a set of general guidelines as to the applicability of a given identification method to a particular experimental context. To our knowledge, no comparison has been made to date on such a scale. In order to ensure the repeatability of our results and their generalization to different robot models or experimental conditions, we propose a new open-source identification framework in the form of a dedicated Matlab toolbox named BIRDy (i.e. *Benchmark for Identification of Robot Dynamics*). BIRDy offers a wide range of functionalities, allowing the end-user to simulate, debug and identify his own robots while focusing on the mathematical aspects of identification rather than on the implementation of his own experimental framework. As observed by [306, 17], offline parameter identification methods often follow a similar workflow, whose main steps usually involve:

1. selecting an appropriate model for the studied system,
2. designing a state trajectory which *excites* the different components of this model,
3. collecting a large amount of experimental data by having the – real or simulated – system follow the generated excitation trajectory,
4. executing the selected identification process,
5. evaluating the quality of the results by comparing the predicted and actual torques along a validation trajectory.

The very structure of BIRDy is directly inspired by this pipeline of operation (c.f. Fig. 35). The various aspects of this configuration are presented and discussed in the following subsections, along with their operation modalities.

5.2.1. Symbolic Model Generation

Model generation is the very first step toward identification: not only should the robot be simulated in a realistic manner but its dynamics should also be expressed in the form of a system of linear equations (c.f. eq. (13)) with a well conditioned identification regressor. BIRDy features a symbolic model generation engine, capable of computing the complete kinematics, dynamics and identification models of any fixed-base serial robot[4]. Similar to the robotic-system-toolbox proposed by [307], the simulated robots are described using dedicated parameter files, containing a Denavit-Hartenberg[5] table as well as the corresponding reference dynamic, friction and control parameters of each link. The dynamic model is generated using the Euler-Lagrange equations of motion (c.f. [9]). It is worth noting that in case the robot is gear-driven, the inertia of the actuators and gearboxes can be precisely taken into account within eq. (1) following the approach of [308]. A simpler alternative – used in BIRDy – consists in modeling the drive-chain inertia as an additional diagonal term $\boldsymbol{I}_a \in \mathbb{R}^{n \times n}$ within the inertia tensor $\boldsymbol{M}(\boldsymbol{\chi}, \boldsymbol{q})$, following the approach of [30]. The symbolic expression of the regressor matrix $\boldsymbol{Y}_{\boldsymbol{\chi}}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q})$ is computed following the approach detailed in [309] and given in the Appendix A.2.4 of this thesis. Finally the vector $\boldsymbol{\beta}$ of base dynamic parameters and the corresponding identification regressor $\boldsymbol{Y}_{\boldsymbol{\beta}}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q})$ are generated numerically, following the QR decomposition method of [22] described in section 2.1.2 of this thesis.

[4] The cases of parallel robots – such as stewart platforms – or floating-base robots – such as Humanoids – are not yet implemented in this approach and are here considered as future works.

[5] Both classic (distal) and modified (proximal) Denavit-Hartenberg conventions are supported.

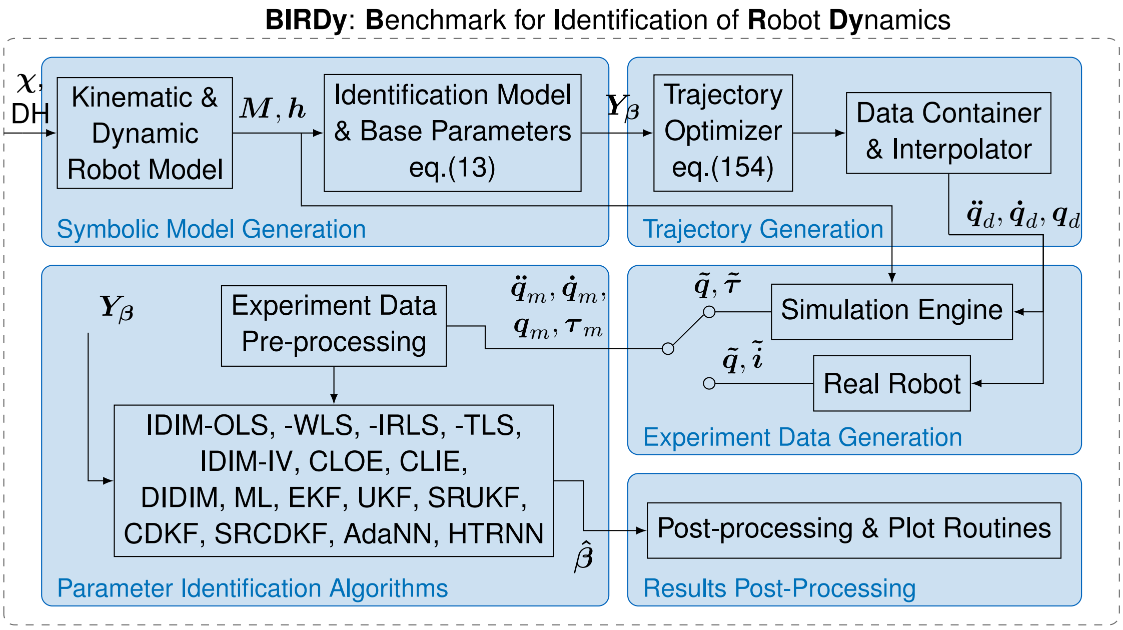

BIRDy: Benchmark for Identification of Robot Dynamics
$\mathcal{X}$, DH
Kinematic & Dynamic Robot Model
$\boldsymbol{M}, \boldsymbol{h}$
Identification Model & Base Parameters eq.(13)
$\boldsymbol{Y}_\beta$
Trajectory Optimizer eq.(154)
Data Container & Interpolator
Symbolic Model Generation
Trajectory Generation
$\ddot{\boldsymbol{q}}_d, \dot{\boldsymbol{q}}_d, \boldsymbol{q}_d$
$\boldsymbol{Y}_\beta$
Experiment Data Pre-processing
$\ddot{\boldsymbol{q}}_m, \dot{\boldsymbol{q}}_m, \boldsymbol{q}_m, \boldsymbol{\tau}_m$
$\tilde{\boldsymbol{q}}, \tilde{\boldsymbol{\tau}}$
Simulation Engine
$\tilde{\boldsymbol{q}}, \tilde{\boldsymbol{i}}$
Real Robot
Experiment Data Generation
IDIM-OLS, -WLS, -IRLS, -TLS, IDIM-IV, CLOE, CLIE, DIDIM, ML, EKF, UKF, SRUKF, CDKF, SRCDKF, AdaNN, HTRNN
$\hat{\boldsymbol{\beta}}$
Post-processing & Plot Routines
Parameter Identification Algorithms
Results Post-Processing

Figure 35 Block diagram of BIRDy, the Benchmark for Identification of Robot Dynamics. In this figure, $\ddot{\boldsymbol{q}}_d, \dot{\boldsymbol{q}}_d, \boldsymbol{q}_d$ refers to the desired trajectory data, obtained from the trajectory interpolator, $\tilde{\boldsymbol{q}}, \tilde{\boldsymbol{\tau}}, \tilde{\boldsymbol{i}}$ respectively denote the noisy joint positions, torques and (in case of a real robot) drive-currents measurements. The actual data $\ddot{\boldsymbol{q}}_m, \dot{\boldsymbol{q}}_m, \boldsymbol{q}_m, \boldsymbol{\tau}_m$ used for identification, is obtained after a short pre-processing step of the raw experiment data.

5.2.2. Trajectory Data Generation

In order to avoid data rank deficiency of the observation matrix $\boldsymbol{W}$, the different components of the dynamic model should be sufficiently excited during the experiments. This can be achieved by means of specific joint trajectories, referred to as exciting. The design of excitation trajectories has been widely investigated over the past two decades, see e.g. [13, 120, 14, 15, 16, 121, 17, 122, 18, 19, 20, 310, 21]. From these works, it appears that the conditioning of the observation matrix $\boldsymbol{W}$ is a relevant quality indicator for the generated trajectory. In our case, joint trajectories are obtained by parametrization of finite Fourier series as presented in [16, 121, 18, 21]. The following cost criterion J_t is minimized over the experiment time horizon using a standard nonlinear programming solver (in our case the "*fmincon*" and "*ga*" Matlab functions):

$$J_t = k_1 \cdot \mathrm{cond}(\boldsymbol{W}^\top \boldsymbol{W}) + \frac{k_2}{\sigma_{min}}, \tag{154}$$

where [311] $\mathrm{cond}(\cdot)$ refers to the condition number of a matrix; σ_{min} is the smallest singular value of $\boldsymbol{W}$; and $k_1, k_2 \in \mathbb{R}_+$ are tuning gains defined by the user. It should be emphasized that although this method allows the generated trajectories to be constrained in order to comply with the physical limitations of the considered robot – such as joint, velocity, torque or operational space limits – it cannot be used "*as is*" for the generation of excitation trajectories on floating-base robots [19]. This specific topic being a research field of its own, it will not be addressed in this implementation and is here considered as future works. Once the trajectory is generated, it is stored in a dedicated data-container. We provide multiple interpolation routines, based on the "*interp1*" Matlab function, so that relevant trajectory data can

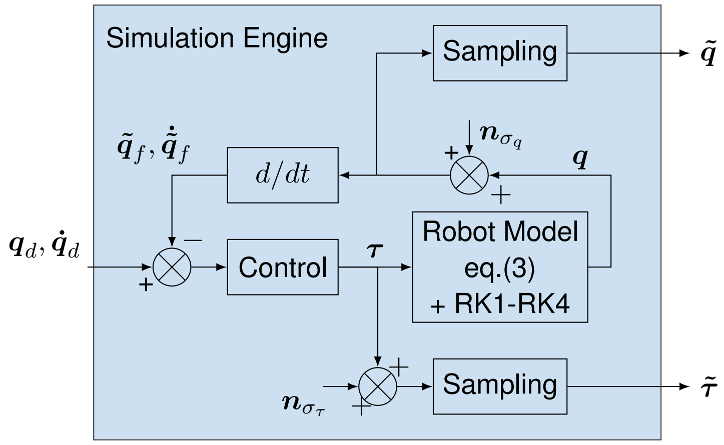

Simulation Engine
Sampling
$\tilde{\boldsymbol{q}}$
$\boldsymbol{n}_{\sigma_q}$
$\tilde{\boldsymbol{q}}_f, \dot{\tilde{\boldsymbol{q}}}_f$
d/dt
+
+
$\boldsymbol{q}$
$\boldsymbol{q}_d, \dot{\boldsymbol{q}}_d$
−
+
Control
$\boldsymbol{\tau}$
Robot Model
eq.(3)
+ RK1-RK4
$\boldsymbol{n}_{\sigma_\tau}$
+
+
Sampling
$\tilde{\boldsymbol{\tau}}$

Figure 36 Detail of the "*Simulation Engine*" of Fig 35.

be obtained at any epoch, from a limited number of points, either generated by the optimizer or coming from a pre-existing file, created outside BIRDy. Note that any third-party trajectory file containing a time series of joint values can be imported into the benchmark, provided its data is first stored in one of its predefined data containers.

5.2.3. Experiment Data Generation and Pre-processing

Identifying the dynamic parameters of a robot manipulator requires the collection of appropriate measurements in terms of joint angles $\boldsymbol{q}$ and torques $\boldsymbol{\tau}$[6] over a predefined time horizon. Such data can be collected when a robot – *real or simulated* – tracks an exciting trajectory. As depicted in Fig. 35, BIRDy provides a set of simulation routines, allowing the previously-generated robot models to be used for this purpose. The data collection process in the case of a simulated robot is depicted in Fig. 36. The direct dynamic model (DDM) of the robot, computed according to equation (3), is subjected to a control torque $\boldsymbol{\tau}$. BIRDy contains implementations of multiple friction models, in particular the *Viscous-Coulomb*, *Stribeck* and *LuGre* models, allowing for more realistic data generation (c.f. [312] and Fig. 37). The resulting joint acceleration signal $\ddot{\boldsymbol{q}}$ is integrated twice, using a fixed-step Runge-Kutta method (RK1, RK2 or RK4) before a zero-mean Gaussian noise $\boldsymbol{n}_{\sigma_q}$ with a standard deviation $\boldsymbol{\sigma}_q$ is added to it. The resulting signal $\tilde{\boldsymbol{q}}$ is then derived and eventually fed back into the controller without filtering. This allows to better account for the effects of the numerical differentiation process occurring on real digital controllers, in terms of amplification of the measurement noise. The control algorithm itself is described in equation (12). The robot tracks the reference trajectory $(\boldsymbol{q}_d, \dot{\boldsymbol{q}}_d)$ at an update frequency f_c. The data sampling process occurs at a frequency $f \leq f_c$, both for $\tilde{\boldsymbol{q}}$ and for the torque measurement $\tilde{\boldsymbol{\tau}}$ (namely the torque signal $\boldsymbol{\tau}$ corrupted by a zero-mean Gaussian noise $\boldsymbol{n}_{\sigma_\tau}$ with a standard deviation $\boldsymbol{\sigma}_\tau$). Note that the reference trajectory files created by BIRDy can also be used to generate desired behaviors directly on a real robot manipulator. However in this case, the data collection process is achieved independently using the robot communication interface. Unlike the BIRDy simulation interface, which provides direct feedback in terms of joint position and torque, the real

[6] Note that contact forces can also be used in the case of floating-base robots, as explained in section 2.4.1.

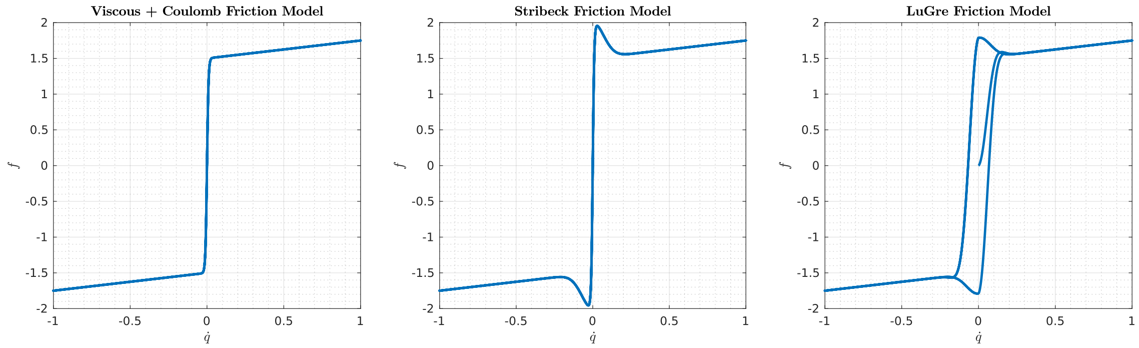


Figure 37 The characteristics of the different friction models implemented in BIRDy.

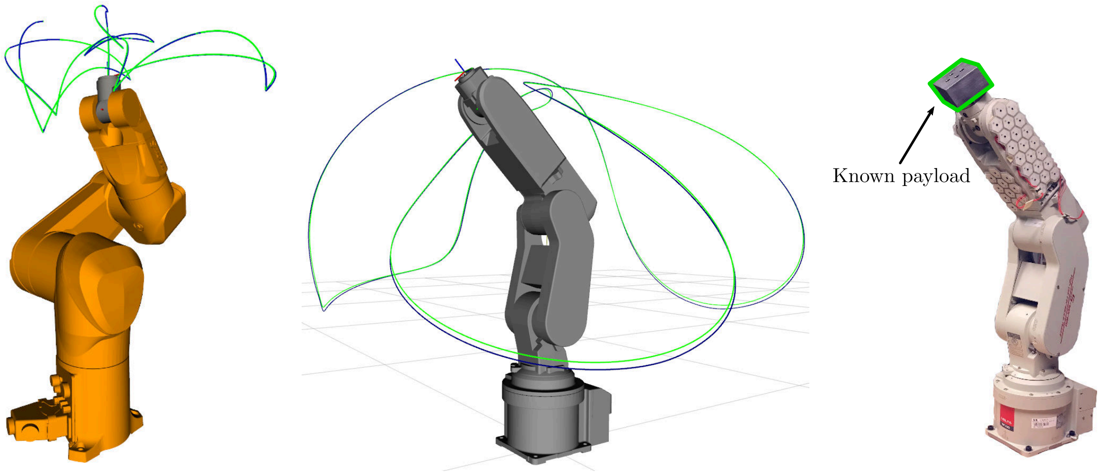
robot interface often provides feedback only in terms of joint position $\tilde{\boldsymbol{q}}$ and drive currents $\tilde{\boldsymbol{i}}$. Nevertheless a reliable prior knowledge of the robot drive gains $\boldsymbol{K}_\tau$ of (12) makes it possible to reconstruct the torque signal $\tilde{\boldsymbol{\tau}}$. In practice, the robot drive gains can be identified following a dedicated Least-Squares method such as that proposed by [313, 314]. The "raw"[7] signals $\ddot{\boldsymbol{q}}_m, \dot{\boldsymbol{q}}_m, \boldsymbol{q}_m, \boldsymbol{\tau}_m$ used for identification purpose, are eventually built from $\tilde{\boldsymbol{q}}, \tilde{\boldsymbol{\tau}}$ by sequential time derivation followed by a parallel decimation step in order to eliminate torque ripple and high-frequency noise components. The parallel decimation process in BIRDy is carried out with a zero-shift low-pass Tchebyshef filter, implemented using the *decimate* Matlab function. Similar to [11, 29] the cutoff frequency f_{dec} is set to be approximately one order of magnitude higher than the joint bandwidths. The decimation ratio n_d by which the input data is being sub-sampled is defined as $n_d = \lfloor 0.8f/2f_{dec} \rceil$ where f is the sampling frequency. As additional band-pass filtering may be required by some algorithms (such as IDIM-OLS), the pre-processing pipeline of BIRDy also features an implementation of zero-shift Butterworth filter that can potentially be applied to the raw joint position signal $\tilde{\boldsymbol{q}}$ before time derivation.

5.3. Benchmarking Parameter Identification Algorithms: Monte Carlo Simulations and Validation using BIRDy

5.3.1. Hardware Description and Experiment Setup

This work takes the form of a *case study*, carried out on two different robots, both in simulation and on the real systems. These robots are the 6-DoF **RV2SQ** industrial manipulator from Mitsubishi, depicted in Fig. 38ii and the 6-DoF **TX40** robot from Staubli, depicted in Fig. 38i and described using the *proximal* DH convention. Widely used in both industry and research, these systems are ideal test platforms for having a similar kinematic structure but rather different communication interfaces and control characteristics. Generally speaking, evaluating the performance of multiple identification methods on different robot models within the same framework is relevant as it allows observing the influence of system-specific factors, such as the sampling rate or the knowledge of the control structure, on the overall algorithm perfor-

[7] The term "raw" here refers to the fact that the signals used for identification are not necessarily filtered. This makes it possible to test the robustness of the different identification algorithms to measurement or numerical differentiation noise.



(i) Staubli TX40 6-DoF robot. (ii) Mitsubishi RV2SQ 6-DoF robot. (iii) Drive-Gains identification.

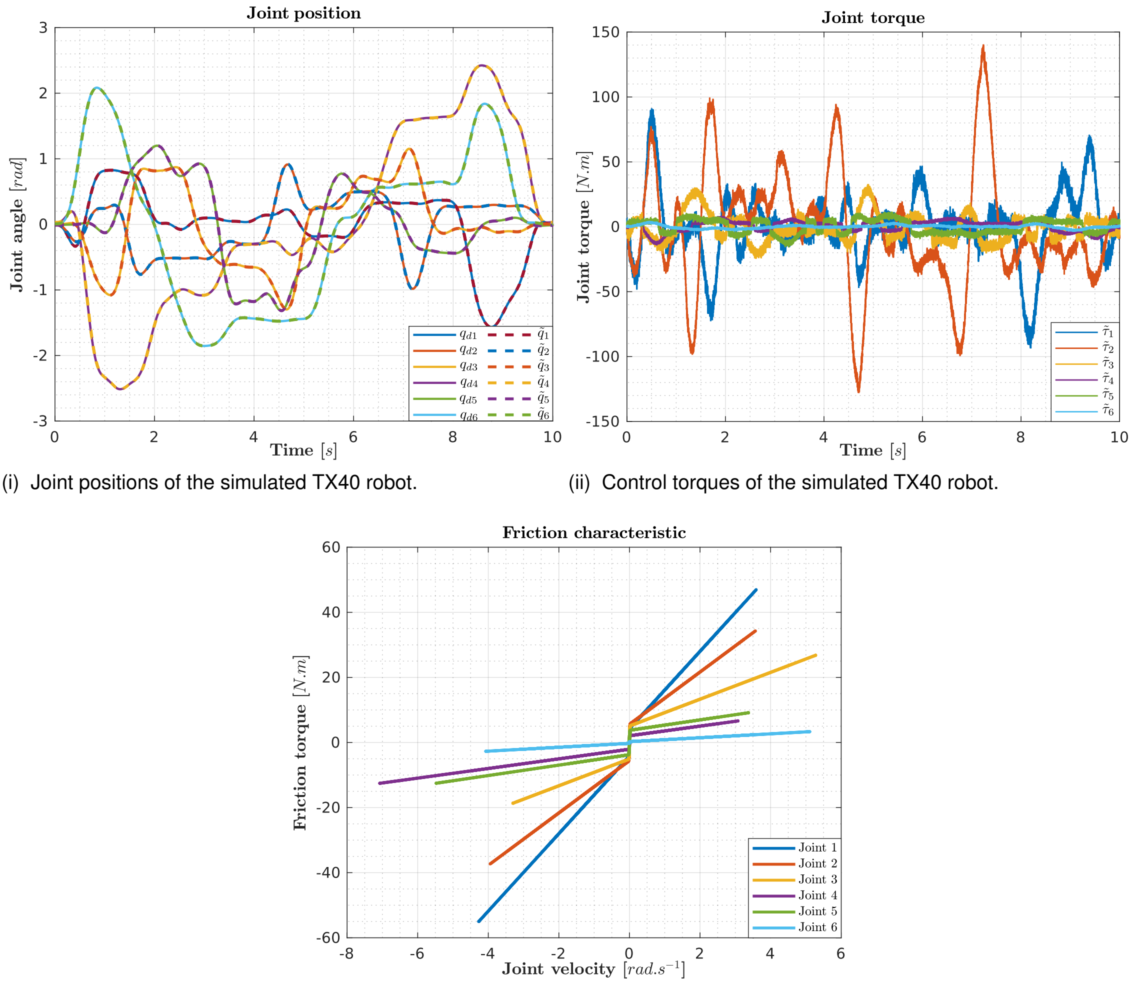
Figure 38 (i, ii): End-effector trajectories – desired in green and actual in dark blue – executed during the experiments on the real robots and visualized on `Rviz` using the Matlab-ROS* interface. The excitation trajectory of the TX40 was manually generated, with trapezoidal velocity profiles and was later imported within BIRDy, while that of the RV2SQ was directly generated using the benchmark constrained Fourier trajectory generation tool. ROS (**R**obot **O**perating **S**ystem) is a middle-ware, i.e. a collection of libraries and communication protocols used for fast development and deployment of robot programs. (iii): Experiment setup used for drive gains identification on the Mitsubishi RV2SQ. Note the external payload, with known inertial parameters, attached to the end-effector of the robot.

mance. This in turns makes it possible to infer guidelines regarding the selection process of an identification algorithm depending on the experimental context. In practice, while the TX-40 has a high-speed communication interface allowing the joint position and torque to be sampled at rates of up to $f = 5kHz$, the data acquisition process on the RV2SQ is achieved at a much lower frequency, namely $f = 140Hz$. Moreover, although the low-level control characteristics of the TX40 are known with a sufficient level of accuracy (see in particular [29]), it must be emphasized that no prior knowledge of control structure, gains nor bandwidth of the RV2SQ is available. In this work, the low-level control structures are assumed to be of cascaded PD type – as described in section 2.1.3 – with a control bandwidth $f_c = 5kHz$. Note that during the simulation experiments the control structure and gains used for data generation and identification are the same. Although hardly verifiable in practice, this hypothesis has critical implications as to the convergence properties of identification algorithms that are based on successive closed-loop DDM simulation runs such as DIDIM, CLIE, CLOE or NKF. We considered an integrated Viscous-Coulomb friction model for both data generation and parameter identification processes[8] (c.f. Fig. 39iii). In this model, the j^{th} component ζ_j of the friction force vector $\boldsymbol{\zeta}$ is expressed in the form of a linear function of two parameters, namely the Coulomb friction force $Fc_j \in \mathbb{R}_+$ and of the viscous friction coefficient $Fv_j \in \mathbb{R}_+$ as

$$\zeta_j(\chi_j, \dot{q}_j) = \underbrace{Fc_j \cdot \text{sign}(\dot{q}_j)}_{\text{Coulomb friction}} + \underbrace{Fv_j \cdot \dot{q}_j}_{\text{Viscous friction}} \tag{155}$$

As a result, the friction coefficients can be directly included within the link's standard dynamic parameter vector $\boldsymbol{\chi}_j \in \mathbb{R}^{13}$, as shown in eq. (2). A specificity of the TX40 is the coupling

[8] This choice is deliberate and justified by the results exposed in [315, 29].



(i) Joint positions of the simulated TX40 robot.

(ii) Control torques of the simulated TX40 robot.

(iii) Friction characteristic of the simulated TX40 robot (without coupling between the joints 5 and 6).

Figure 39 Experiment data obtained from of the simulation engine with a model of the TX40 industrial robot.

existing between the joints 5 and 6 of the robot. As explained in [67], this can be accounted for using two additional friction parameters denoted Fvm_6 and Fcm_6. Although it was decided to neglect this coupling during the simulation experiments, this was later taken into account during the validation phase on the real robot. In the context of simulation experiments, the model generation step resulted in a vector $\boldsymbol{\beta} \in \mathbb{R}^{52}$ of the base dynamic parameters for each robot. During the validation experiments on the TX40, this base parameter vector was of dimension 54 due to Fvm_6 and Fcm_6. It is worth noting that in the more general case where joint friction is nonlinear, provided that the friction model is not *load dependent*, the IDM given by (1) can be identified using a separable approach as suggested by [315, 47]. The detail of such method being out of the scope of this chapter, the reader is redirected to [316] and the references therein for a more detailed review of friction modeling and identification. The reference trajectories used for the experiments were generated using the optimization method presented in section 5.2.2 with the gains $k_1 = 1$ and $k_2 = 100$, and over a time horizon of $10s$. Two trajectories are considered for each robot, for identification and validation purposes respectively. The generation of experiment data is carried out using Runge-Kutta fixed-step integration (RK4) of the close-loop DDM (c.f. Figs.39i and 39ii). The computations were realized using Matlab R2020a on a AMD-Threadripper 1920X workstation with 32GB of RAM. When possible the parallel computing toolbox from Matlab was used, alongside with the Matlab C/C++ Coder to enhance the execution speed of concatenated for-loops, which are found in sigma-point Kalman filters. The number of parallel threads allocated to each algorithm was limited to six, for repeatability reasons. It is worth mentioning that the closed-loop model simulations required by some of identification algorithms – such as IDIM-IV, DIDIM, CLIE, CLOE or NKF – were carried out at the same frequency f_c as in the data generation process. Nevertheless in this case, the DDM time integration was performed using an Euler (RK1) algorithm for performance reasons[9].

The obtained closed-loop simulation data was then sampled and decimated at the very same frequencies as in the experiments, before being used in the corresponding algorithms. In order to obtain statistically relevant data, the simulation experiments were executed following a Monte Carlo scheme, consisting in a set of b independent runs, where b denotes the dimension of the base parameter vector $\boldsymbol{\beta}$. For each method and each independent run, the identification process was repeated 25 times with different initial parameter vectors, in order to investigate the sensitivity to initial conditions, eventually resulting in a total of $M = 25 \cdot b = 1300$ runs for each identification method and on each robot. The 25 initial parameter vectors were obtained using the values of the reference parameters (i.e. used for the collection of simulation data) distorted by a relative error of 15%, which is representative of the tolerances obtained with computer-aided design (CAD) software. The validation experiments were performed on the real robots. In this case, the identification methods requiring an initial parameter estimate – namely IDIM-IV, DIDIM, CLOE, CLIE, ML, NKF, AdaNN and HTRNN – were initialized using the filtered IDIM-OLS estimate.

5.3.2. Selected Figures of Merit for Performance Evaluation

Besides the element-wise errors between the reference parameter vector $\boldsymbol{\beta}_{ref}$ and the estimated parameter vector $\hat{\boldsymbol{\beta}}$, we define the following figures of merit (FOM), in order to quantify the performance of the evaluated parameter identification algorithms:

1. the average relative angle difference $d_{\boldsymbol{q}}$ defined as

$$d_{\boldsymbol{q}} = \frac{1}{N} \sum_{k=1}^{N} \left\| \boldsymbol{q}_{\boldsymbol{\beta}_{ref}}(t_k) - \boldsymbol{q}_{\hat{\boldsymbol{\beta}}}(t_k) \right\|_2 / \left\| \boldsymbol{q}_{\boldsymbol{\beta}_{ref}}(t_k) \right\|_2 \tag{156}$$

   where $\boldsymbol{q}_{\boldsymbol{\beta}_{ref}}$ and $\boldsymbol{q}_{\hat{\boldsymbol{\beta}}}$ are obtained respectively by direct measurements on a – real or simulated – robot and closed-loop noise-free simulation of the DDM using the estimated parameter vector $\hat{\boldsymbol{\beta}}$,

[9] Aside the significantly larger computation time, no significant difference was noticed in the identification results when using more advanced integration algorithms, such as RK2 or RK4.

2. the average relative torque difference $d_{\boldsymbol{\tau}}$ defined as

$$d_{\boldsymbol{\tau}} = \frac{1}{N} \sum_{k=1}^{N} \left\| \boldsymbol{\tau}_{\boldsymbol{\beta}_{ref}}(t_k) - \boldsymbol{\tau}_{\hat{\boldsymbol{\beta}}}(t_k) \right\|_2 / \left\| \boldsymbol{\tau}_{\boldsymbol{\beta}_{ref}}(t_k) \right\|_2 \tag{157}$$

where $\boldsymbol{\tau}_{\boldsymbol{\beta}_{ref}}(t_k)$ is the measured torque and $\boldsymbol{\tau}_{\hat{\boldsymbol{\beta}}}(t_k) = \boldsymbol{Y}_{\boldsymbol{\beta}}(\ddot{\boldsymbol{q}}_d(t_k), \dot{\boldsymbol{q}}_d(t_k), \boldsymbol{q}_d(t_k))\hat{\boldsymbol{\beta}}$, $\forall k = \{1, \cdots, N\}$,

3. the mean total time d_t that is required to compute one parameter estimate,

4. the mean number $d_{N_{it}}$ of iterations until convergence,

5. the mean number $d_{N_{sim}}$ of model simulations (for methods which require it).

In the context of Monte Carlo experiments, we will discuss the mean values and standard deviations of these different figures of merit across the $M = 1300$ runs.

5.3.3. Implementation details

5.3.3.1 IDIM-OLS, -WLS, -IRLS and -TLS implementations:

Besides IDIM-OLS, -WLS and -IRLS, BIRDy features an implementation of the Total Least-Squares IDIM-TLS algorithm. For each of these methods, two distinct experimental scenarios were considered. In the first scenario, the least-squares methods were executed directly on the raw – unfiltered – data set $\ddot{\boldsymbol{q}}_m, \dot{\boldsymbol{q}}_m, \boldsymbol{q}_m, \boldsymbol{\tau}_m$ obtained by successive central differentiation and parallel decimation of $\tilde{\boldsymbol{q}}, \tilde{\boldsymbol{\tau}}$. In the second scenario, identification was carried out using the filtered signals. Data filtering was implemented in a similar fashion as in [11, 27], namely through zero-lag Butterworth band-pass filtering of the raw joint positions[10] signal $\tilde{\boldsymbol{q}}$ in the range $[0, \omega_f]$, followed by one (resp. two) zero-shift central differentiation steps, to obtain the corresponding joint velocities and accelerations. We here used the "*filtfilt*" Matlab function to implement the desired zero-lag forward-backward Butterworth filter with a bandwidth of 50Hz.

5.3.3.2 ML implementation:

Similarly to the IDIM-LS methods, two different variations of the ML identification algorithm were evaluated, namely with and without data filtering. In the proposed implementation of the ML identification method, the values of the diagonal variance matrix $\boldsymbol{\sigma}_k^2$ are set based on variance measurements of the error signal between the measured joint derivatives $\tilde{\boldsymbol{q}}, \tilde{\dot{\boldsymbol{q}}}, \tilde{\ddot{\boldsymbol{q}}}$ and the simulation reference $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$. Note that it is usually difficult to obtain the value of $\boldsymbol{\sigma}_k^2$ on a real system as the reference quantities $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$ are not available. The resolution of the nonlinear optimization problem eq. (32) is carried out using the Levenberg-Marquardt algorithm through the *lsqnonlin* function from the Matlab Optimization Toolbox. The Jacobian matrices $\boldsymbol{G}_k$ are computed numerically, with finite differences and a tolerance of 10^{-7}. The

[10] Note that in general, only the joint positions are filtered and the raw – potentially decimated – torque values $\tilde{\boldsymbol{\tau}}$ are used directly.

algorithm stops once any of the following criteria is reached:

- $\left(\|\boldsymbol{\rho}_{ML}^{i+1}\|_2 - \|\boldsymbol{\rho}_{ML}^{i}\|_2\right) / \|\boldsymbol{\rho}_{ML}^{i}\|_2 < \mathrm{tol}_1$ where $\boldsymbol{\rho}_{ML}^{i}$ denotes the error at iteration i, defined following eq. (32) as $\boldsymbol{\rho}_{ML}^{i} = \dfrac{1}{2}\sum_{k=1}^{N} \boldsymbol{\varepsilon}^\top(t_k) \left(\boldsymbol{G}_k \boldsymbol{\sigma}_k^2 \boldsymbol{G}_k^\top\right)^{-2} \boldsymbol{\varepsilon}(t_k)$;
- $\max\limits_{j} \left|\left(\hat{\boldsymbol{\beta}}_{ML}^{i+1}(j) - \hat{\boldsymbol{\beta}}_{ML}^{i}(j)\right) / \hat{\boldsymbol{\beta}}_{IV}^{i}(j)\right| < \mathrm{tol}_2$, $\forall \hat{\boldsymbol{\beta}}_{ML}^{i}(j) \neq 0$ with j the index and i the iteration number.
- maximum number of iterations: $i_{max} = 10$;

where the values of the function tolerance tol_1 and step tolerance tol_2 are set according to the guidelines given in [40]. In our case, we have $\mathrm{tol}_1 = 0.1\%$ and $\mathrm{tol}_2 = 2.5\%$.

5.3.3.3 IDIM-IV implementation:

The IDIM-IV method is only executed on the unfiltered data set. The instrument matrix $\boldsymbol{Z}^i$ at iteration i is filled with the noise-free simulated data, obtained by integration of the closed-loop DDM using the current parameter estimate $\hat{\boldsymbol{\beta}}_{IV}^{i-1}$ (c.f. equation (27)). The control structure and gains are the same as those used during the experiment data generation process. Integration is carried out using the Euler (RK1) at the rate $f_c = 5kHz$ of the data generation loop. The data pre-processing step only involves sampling and decimation, in order to fit the sampled data-set. To stop the sequence of linear LS problems solved by the IDIM-IV method, a set of three stop criteria are implemented:

- $\left(\|\boldsymbol{\rho}_{IV}^{i+1}\|_2 - \|\boldsymbol{\rho}_{IV}^{i}\|_2\right) / \|\boldsymbol{\rho}_{IV}^{i}\|_2 < \mathrm{tol}_1$ where $\boldsymbol{\rho}_{IV}^{i}$ denotes the error at iteration i, defined as $\boldsymbol{\rho}_{IV}^{i} = \boldsymbol{y}_\tau - \boldsymbol{W}^{(i-1)}\hat{\boldsymbol{\beta}}_{IV}^{i}$;
- $\max\limits_{j} \left|\left(\hat{\boldsymbol{\beta}}_{IV}^{i+1}(j) - \hat{\boldsymbol{\beta}}_{IV}^{i}(j)\right) / \hat{\boldsymbol{\beta}}_{IV}^{i}(j)\right| < \mathrm{tol}_2$, $\forall \hat{\boldsymbol{\beta}}_{IV}^{i}(j) \neq 0$ with j the index and i the iteration number;
- maximum number of iterations: $i_{max} = 10$.

where the values of the function tolerance tol_1 and step tolerance tol_2 are set according to the guidelines given in [29, 47]. In our case, we have $\mathrm{tol}_1 = 2.5\%$ and $\mathrm{tol}_2 = 2.5\%$.

5.3.3.4 DIDIM implementation:

Similarly to the IDIM-IV, the simulation data are obtained by Euler (RK1) integration of the closed-loop DDM at a rate of 5kHz. The following three stop criteria were implemented within the DIDIM method:

- $\left(\|\boldsymbol{\rho}_{DIDIM}^{i+1}\|_2 - \|\boldsymbol{\rho}_{DIDIM}^{i}\|_2\right) / \|\boldsymbol{\rho}_{DIDIM}^{i}\|_2 < \mathrm{tol}_1$ where $\boldsymbol{\rho}_{DIDIM}^{i}$ denotes the torque error at iteration i, defined as $\boldsymbol{\rho}_{DIDIM}^{i} = \boldsymbol{y}_\tau - \boldsymbol{W}_s^{(i-1)}\hat{\boldsymbol{\beta}}_{DIDIM}^{i}$. Note that unlike IDIM-IV, we here make use of the simulated joint positions to compute the observation matrix;

- $\max_j \left| \left(\hat{\boldsymbol{\beta}}^{i+1}_{DIDIM}(j) - \hat{\boldsymbol{\beta}}^{i}_{DIDIM}(j) \right) / \hat{\boldsymbol{\beta}}^{i}_{DIDIM}(j) \right| < \mathrm{tol}_2, \forall \hat{\boldsymbol{\beta}}^{i}_{DIDIM}(j) \neq 0$ with j the parameter index and i the iteration number;
- maximum number of iterations: $i_{max} = 10.$

where the values of the function tolerance tol_1 and step tolerance tol_2 are set according to the guidelines given in [66], namely $\mathrm{tol}_1 = 2.5\%$ and $\mathrm{tol}_2 = 2.5\%$.

5.3.3.5 Relevant details of the CLIE and CLOE implementations:

The proposed implementation of CLIE and CLOE makes use of the Levenberg-Marquardt algorithm, and is implemented using the *lsqnonlin* function from the Matlab Optimization Toolbox. The algorithm stops once any of the following criteria is reached:

- $\left(\left\| \boldsymbol{\rho}^{i+1} \right\|_2 - \left\| \boldsymbol{\rho}^{i} \right\|_2 \right) / \left\| \boldsymbol{\rho}^{i} \right\|_2 < \mathrm{tol}_1$ where $\boldsymbol{\rho}^i$ denotes the error at iteration i. In the case of CLIE one has, $\boldsymbol{\rho}_i = \boldsymbol{\varepsilon}^i_{CLIE}$ as defined in (39) while in the case of CLOE, we have $\boldsymbol{\rho}_i = \boldsymbol{\varepsilon}^i_{CLOE}$ as defined in (36).
- $\max_j \left| (\hat{\boldsymbol{\beta}}^{i+1}_{CLIE}(j) - \hat{\boldsymbol{\beta}}^{i}_{CLIE}(j)) / \hat{\boldsymbol{\beta}}^{i}_{CLIE}(j) \right| < \mathrm{tol}_2$, (resp. $\max_j \left| (\hat{\boldsymbol{\beta}}^{i+1}_{CLOE}(j) - \cdots \right.$ $\left. \cdots \hat{\boldsymbol{\beta}}^{i}_{CLOE}(j)) / \hat{\boldsymbol{\beta}}^{i}_{CLOE}(j) \right| < \mathrm{tol}_2$), $\forall \hat{\boldsymbol{\beta}}^{i}_{CLIE}(j) \neq 0$ (resp. $\hat{\boldsymbol{\beta}}^{i}_{CLOE}(j) \neq 0$) with j the parameter index and i the iteration number;
- maximum number of iterations: $i_{max} = 10.$

where the values of the function tolerance tol_1 and step tolerance tol_2 are set according to the guidelines given in [47]. In our case, we have $\mathrm{tol}_1 = 2.5\%$ and $\mathrm{tol}_2 = 2.5\%$. Note that although only the Levenberg-Marquardt approach is investigated in this chapter, BIRDy actually contains multiple CLIE and CLOE implementations, based on genetic algorithm, particle swarm optimization and the Nelder-Mead nonlinear simplex methods. These methods could be easily implemented using respectively the *ga*, *particleswarm* and *fminsearch* function from the Matlab Optimization Toolbox.

5.3.3.6 DDIM-NKF implementation:

Based on the guidelines of [76], it was decided to exploit the joint filtering approach to parameter identification. BIRDy actually features multiple flavors of nonlinear Kalman filters, namely the Extended Kalman Filter (EKF), the Unscented Kalman Filter (UKF), the Central Difference Kalman Filter (CDKF), the bootstrap Particle Filter (PF) as well as the improved numerically stable implementations of these filters, known as Square Root Extended Kalman Filter (SREKF), Square Root Unscented Kalman Filter (SRUKF), and Square Root Central Difference Kalman Filter (SRCDKF). Aside the particle filter – whose results were rather unsatisfactory – each of these methods were considered in this work. We used the same set of tuning parameters for each filter. The initial covariance matrix $\boldsymbol{P}_0$ and the process noise covariance matrix $\boldsymbol{Q}$ are respectively given by: $\boldsymbol{P}_0 = \mathrm{diag}\left(\left[p_1 \cdot \mathbf{1}_{n\times n} \; p_2 \cdot \mathbf{1}_{n\times n} \; 0.15 \cdot \mathrm{diag}(\left| \boldsymbol{\beta}^0 \right| + \epsilon) \right] \right)$ and $\boldsymbol{Q} = \mathrm{diag}\left([q_1 \cdot \mathbf{1}_{n\times n} \; q_1 \cdot n_d / f \cdot \mathbf{1}_{n\times n} \; q_2 \cdot \mathbf{1}_{p\times p}] \right)$, where $\boldsymbol{\beta}^0$ refers to the initial parameter estimate, $\epsilon \in \mathbb{R}^b_+$ provides additional tolerance for small values of $\hat{\boldsymbol{\beta}}_0$, f is the sampling frequency and n_d is the decimation rate. The scalar values (p_1, p_2) and (q_1, q_2) were set heuristically to $(0.1, 0.1)$ and $\left(10^{-4}, 10^{-5}\right)$ respectively. Following the ideas developed in [76], q_2 was annealed during identification. The initial measurement noise covariance matrix $\boldsymbol{R}$ is given by $\boldsymbol{R} = \sigma_q^2 \cdot \mathbf{1}_{n \times n}$ (in rad^2) according to the data generation procedure. The Jacobian matrices of the EKF and SREKF are computed numerically, with finite differences and a tolerance of 10^{-7}. The tuning coefficients of the UKF and SRUKF, referred to as (α, β, κ) in [76], were set to $\left(10^{-2}, 2, 0\right)$ respectively. Finally, the tuning coefficient h of the CDKF and SRCDKF, is set to $h = \sqrt{3}$, following the recommendations of [76].

5.3.3.7 AdaNN implementation:

BIRDy features two different implementations of the Adaline neural network, namely with classic[11] gradient descent and with stochastic gradient descent. In this work, we considered the stochastic gradient descent implementation. In this implementations, the data is reshuffled at each training epoch in order to avoid cycles. Similarly to IDIM-OLS methods, two executions of the AdaNN were considered, namely with and without joint position data band-pass filtering. The maximum number of training epochs was set heuristically to $10 \cdot N$.

5.3.3.8 HTRNN implementation:

Similarly to IDIM-OLS and AdaNN methods, two versions of the HTRNN were implemented, namely with and without data filtering. The practical implementation of the HTRNN parameter estimator simply consists in re-injecting eq. (51) into eq. (49). As explained in [92], special attention must be given to the parameter bounds, defined by $\boldsymbol{\alpha}$. In our case, we selected $\boldsymbol{\alpha} = 1.5 \cdot \max\left(|\hat{\boldsymbol{\beta}}_0| + \epsilon\right)$, in order to account for the uncertainty in the initial parameter estimate $\hat{\boldsymbol{\beta}}_0$, where $\epsilon \in \mathbb{R}_+^b$ provides additional tolerance for small values of $\hat{\boldsymbol{\beta}}_0$. The maximum number of training epochs was set heuristically to $10 \cdot N$. Finally the learning rate $\eta \in \mathbb{R}_+^\star$ was set to $\eta = 10^{-6}$ for the experiments performed on the TX40 and to $\eta = 10^{-7}$ for the experiments performed on the RV2SQ robot.

5.3.3.9 Physically Consistent PC-OLS, -WLS, -IRLS, -IV and -DIDIM implementations:

In this work, we used the CVX optimization framework ([317]) alongside with MOSEK ([318]) in order to solve the SDP (64), (66) and (65) subject to physicality constraints. As previously, PC-IDIM-OLS, -WLS and -IRLS were tested with and without data filtering and the tolerances in PC-DIDIM and PC-IDIM-IV were set to the same level as DIDIM and IDIM-IV. When activated, the regularization factor λ was set to a value of $\lambda = 10^{-2}$ and was otherwise maintained to zero.

5.3.4. Case study on the simulated TX40 and RV2SQ

In this work, we executed a set of fifteen different Monte Carlo Simulations (MCS) experiments, each containing 1300 runs. The same identification methods were executed on the

[11] By *classic*, we refer to the fact that the whole observation matrix is used in the gradient computation, unlike the stochastic gradient, where only one sample of $\boldsymbol{W}$ is considered.

MCS experimental conditions	**TX40 decim 1**	**TX40 decim 4**	**RV2SQ decim 1**
$\boldsymbol{\sigma_q} = 10^{-4}$rad, $\boldsymbol{\sigma_\tau} = 5 \cdot 10^{-2}$N.m	MCS-TX40-1-1	MCS-TX40-4-1	MCS-RV2SQ-1-1
$\boldsymbol{\sigma_q} = 10^{-3}$rad, $\boldsymbol{\sigma_\tau} = 5 \cdot 10^{-2}$N.m	MCS-TX40-1-2	MCS-TX40-4-2	MCS-RV2SQ-1-2
$\boldsymbol{\sigma_q} = 10^{-2}$rad, $\boldsymbol{\sigma_\tau} = 5 \cdot 10^{-2}$N.m	MCS-TX40-1-3	MCS-TX40-4-3	MCS-RV2SQ-1-3
$\boldsymbol{\sigma_q} = 10^{-3}$rad, $\boldsymbol{\sigma_\tau} = 10^{-1}$N.m	MCS-TX40-1-4	MCS-TX40-4-4	MCS-RV2SQ-1-4
$\boldsymbol{\sigma_q} = 10^{-2}$rad, $\boldsymbol{\sigma_\tau} = 10^{-1}$N.m	MCS-TX40-1-5	MCS-TX40-4-5	MCS-RV2SQ-1-5

Table 1 Overview and naming convention of the different experiments performed during the the MCS. The first row of the table refers to noise conditions that are close to what can be encountered on an industrial robot manipulator. The four other rows explore the performance of identification in case of substantially higher joint position and torque noise levels.

TX40 and on the RV2SQ for data decimation frequencies of 500Hz and 100Hz – corresponding to decimation ratios of 1 and 4 – and under five different joint-position and torque noise levels. The detail of the different experiment conditions is given in Table 1. Note that the high joint levels of position noise explored during these experiments result in substantial degradation of tracking performance, visible in the results. Although interesting on a theoretical perspective, this is however not representative of the reality of a commercial robot platform. Given the substantial amount of generated data, only a subset of relevant results will be included in this chapter. The reader is referred to the complementary reports for a more detailed overview of the results of the MCS experiments.

5.3.5. Validation experiments on the real TX40 and RV2SQ

A set of validation experiments were conducted on the real Staubli TX-40 and Mitsubishi RV2SQ robots. The aim of these experiments was to assess the performance of the different identification algorithms presented in this chapter on real systems, with unknown sensor characteristics, unknown control structure, and potentially non-negligible model errors stemming from non-linearities in the joint friction. During these experiments, the coupling between the fifth and sixth joint of the TX40 was considered for better precision resulting in a set of 54 base parameters as opposed to the 52 considered during the MCS experiments. In both cases the robot drive gains were identified following the Least-Squares approach developed by [313, 314], which consists of having the robot track a permanently exciting trajectory, both with and without a well-known external payload, rigidly attached to the end-effector as depicted in Fig. 38iii in the case of the RV2SQ. Drive gain identification is made possible by analyzing variations in the control current, sampled along the same excitation trajectories, both with and without the external payload attached to the end-effector. A good prior knowledge of the payload parameters is, of course, essential to the quality of the estimation. In practice, this is made possible either by direct measurements or through the use of modern CAD software, given the generally simple geometry of the payload. During the experiments, the benchmark was executed a *single time* for each robot, along a dedicated validation trajectory. The joint position and drive current data were sampled at a rate of 1kHz on the TX40 and at a rate of 140Hz on the RV2SQ. The detailed results of these experiments in terms of the values of the different figures of merit defined in section 5.3.2, are presented in tables 2, and 3 for the real TX40 and the real RV2SQ. Finally the reconstructed torque signals for the IDIM-IV and DIDIM methods[12] are displayed on Fig. 41 for the TX40 and on Fig. 42 for the RV2SQ.

5.4. Results, Discussion and Perspectives

This section presents the different experimental results obtained during the MCS with BIRDy and proposes possible interpretations based on quantitative criteria.

5.4.1. Analysis and Discussion of the Results

For the sake of clarity, the present discussion is organized around a set of specific comparison points, namely the *noise-immunity*, the *estimation accuracy*, *convergence properties* and finally the *computational cost*. These comparison points will be discussed within three dedicated sub-sections. Note that due to the substantial amount of experimental results obtained during the MCS, only the most relevant graphs and result tables are actually included as part of this thesis. Other graphs and experimental results mentioned in the discussion are made available to the reader in a dedicated online database (c.f. Appendix D.1).

5.4.1.1 Noise Immunity

The Monte Carlo simulation results in terms of the different figures of merit shown in the tables of the supplementary material for different levels of noise clearly indicate that the AdaNN, HTRNN, ML, IDIM-OLS, -WLS, -IRLS and -TLS methods are not robust to the presence of noise in the joint-position signal and its temporal derivatives, especially when the sampling rate is “high”, as is for instance the case in *MCS-TX40-1-1–MCS-TX40-4-5*. On a theoretical point of view, this stems from the noise-induced correlation between the observation matrix $\boldsymbol{W}$ and the vector of sampled torque errors $\boldsymbol{\varepsilon}$, yielding $\mathrm{E}(\boldsymbol{W}^\top\boldsymbol{\varepsilon}) \neq \boldsymbol{0}$. The influence of data filtering is clearly visible in Fig. 40, where a clear improvement can be observed between IDIM-WLS without and with joint position zero-shift filtering. It should be emphasized that since the torque noise is purely additive, it does not affect the statistical consistency of the estimates, but rather their statistical efficiency. This can be demonstrated by noticing the similarity between the results of *MCS-TX40-1-4*, *MCS-TX40-4-4* and *MCS-RV2SQ-1-4*, (c.f. supplementary material, section D.1), where the joint torque noise level is set to $\boldsymbol{\sigma_\tau} = 10^{-1}$ N·m, with that of *MCS-TX40-1-2*, *MCS-TX40-4-2* and *MCS-RV2SQ-1-2* (c.f. supplementary material, section D.1), where the torque noise is set to be twice as low. It should also be noted in practice that a low torque signal-to-noise ratio (SNR) could also imply that the trajectories are not exciting enough or that the hardware is not well designed. The most critical noise is, actually, the one corrupting the joint angle readings as it significantly contributes to the bias of the IDIM-LS methods. The influence of noise in the joint angle measurements is made

[12] Only the IDIM-IV and DIDIM reconstructed torques are displayed in the case of Fig. 41 and Fig. 42 for the sake of clarity.

visible within the experiments described in the first three rows of Table 1, namely in [*MCS-TX40-1-1*, *MCS-TX40-1-2*, *MCS-TX40-1-3*], [*MCS-TX40-4-1*, *MCS-TX40-4-2*, *MCS-TX40-4-3*], and [*MCS-RV2SQ-1-1*, *MCS-RV2SQ-1-2*, *MCS-RV2SQ-1-3*]. Within these experiments, the robots are actually tracking the same excitation trajectories, with the same control laws and initial parameter estimates, but with three different orders of magnitude in terms of joint position noise, namely $\boldsymbol{\sigma_q} = 10^{-4}$ rad, $\boldsymbol{\sigma_q} = 10^{-3}$ rad and $\boldsymbol{\sigma_q} = 10^{-2}$ rad. Although the first noise level within these experiments is consistent with observations made on actual robots, it should be emphasized that the last two noise levels, are here only provided for indicative purposes. In practice, re-injecting such signals into the low-level control loop of a robot would in turn result in substantial noise in the control signals–as shown by the degraded values of the $d_{\boldsymbol{\tau}}$ figure of merit obtained for $\boldsymbol{\sigma_q} = 10^{-3}$ rad and $\boldsymbol{\sigma_q} = 10^{-2}$ rad–eventually leading to poor control performance.

The experimental results suggest that without data-filtering, the AdaNN, HTRNN, ML, IDIM-OLS, -WLS, -IRLS and -TLS methods provide biased estimates that poorly – or at least improperly – match the objective values, although the enhanced robustness of ML, IDIM-WLS and IDIM-IRLS compared to AdaNN, HTRNN, IDIM-OLS and -TLS should be noted. In any case, it appears that performing a tailor-made data-filtering based on a zero-shift forward-backward Butterworth filter significantly improves the results. This can be explained by the fact that filtering turns the noisy observation matrix $\boldsymbol{W}$ into a noise-free matrix denoted by $\boldsymbol{W}_{nf}$. Loosely speaking, this breaks the correlation between $\boldsymbol{W}$ and the vector ε of sampled errors. It is here worth pointing out that the use of simple forward low-pass filters generally leads to strongly biased estimates. This is justified in [38] by the fact that the phase shift induced by such filters is not accounted for in the IDM and is, therefore, considered to be a modeling error yielding $\boldsymbol{W} \neq \boldsymbol{W}_{nf}$.

The DIDIM, CLIE and CLOE methods appear to be robust in general to the measurement noise in the joint position signal. This is usually expected since the DDM simulation step occurring in DIDIM, IDIM-IV, CLOE and the IDM simulation occurring in CLIE are noiseless. It should be noted that although IDIM-IV can also be considered to be robust to noise as suggested in particular by the results of *MCS-TX40-4-1* and *MCS-TX40-4-2*, it tends to fail when the noise level is unreasonably high, as is for example the case in *MCS-TX40-4-3* or *MCS-TX40-4-2*. These observations are consistent with those published in [29]. It should moreover be highlighted that when the sampling rate is high, typically above 500 Hz on a robot, there might actually be a strong correlation between the measured samples, resulting in poor conditioning of the observation matrix and consequently biased parameter estimates. In practice, sub-sampling the signal helps breaking this correlation. Nevertheless, this should be executed carefully as the high-frequency components of the signal tend to alias during the process. Filtering the signal before sub-sampling, or in other words performing parallel decimation, helps mitigating this effect. The reader is redirected toward [319] for a more detailed discussion on the topic. The influence of parallel decimation is clearly visible on the results of *MCS-TX40-4-1–MCS-TX40-4-5*, in the corresponding FOM tables, where one may observe a homogenization of the performance of the different methods with, however, the notable exception of the Kalman filters (this specific point will be addressed in more details within Section 5.4.1.2 of this chapter). Although the performance is already satisfactory for IDIM-IV and even good for DIDIM in the case of standard joint position noise levels (typically $\boldsymbol{\sigma_q} = 10^{-4}$ rad and $\boldsymbol{\sigma_q} = 10^{-3}$ rad), it should be noted that parallel decimation has a positive influence over the condition number of the Jacobian in OE algorithms as pointed in the discussion of [28].

5.4.1.2 Estimation Accuracy

The results of the three Monte Carlo Simulation (MCS) experiments indicate that the DIDIM and IDIM-IV methods generally tend to provide the most accurate parameter estimates, and accordingly the best torque tracking performance in the context of standard joint position noise levels (i.e., MCS-RV2SQ-1-1, MCS-RV2SQ-1-2, MCS-TX40-1-1, MCS-TX40-1-2, MCS-TX40-4-1 and MCS-TX40-4-2 in the supplementary material). Note that by accuracy, we here refer to the error $\boldsymbol{\beta}_{err}$ between the estimated parameter vector $\hat{\boldsymbol{\beta}}$ and the reference $\boldsymbol{\beta}_{ref}$ used for experiment data generation. This is visible in the experiment reports, provided alongside with this chapter (c.f. section D.1). One may of course argue that these performances are noticeably similar to that of AdaNN, HTRNN, IDIM-OLS, -WLS, -IRLS, ML with filtered or decimated data as well as CLIE and –to a lower extent– CLOE.

Interestingly, IDIM-IV and DIDIM does not perform as well on the real RV2SQ robot although the resulting torque tracking performance remains satisfactory as shown in the *attached experiment reports*. This can be explained by the rather constraining hypothesis made during the simulation experiments, namely that the control law is well known and that these are not model errors (e.g., friction or coupling). One may also notice that CLIE provides better results than CLOE, which can be mainly explained by the lack of sensitivity of the simulated joint positions and velocities against parameters' variations due to the control. When developing a controller, the following approximation $\boldsymbol{q} \approx \boldsymbol{q}_d$ is expected which means that the controller must be robust enough against parameters' variations. This result agrees with those published in [66]. Please note that when badly initialized or in the presence of modeling errors, iterative methods such as IDIM-IV, DIDIM, CLIE, CLOE, NKF or AdaNN usually fail to converge to the objective values as the internal DDM or IDM simulation process converge to an inconsistent state. In practice, a reasonable initialization value for the parameters can be obtained from a modern CAD software or from filtered IDIM-OLS. Although AdaNN and IDIM-OLS are asymptotically equivalent, one may note that the results in chapters 2.1 and 22.1 of the *attached experiment reports* suggest that AdaNN is less efficient than IDIM-OLS – statistically speaking – as the variance of the AdaNN estimate appears to be at least one order of magnitude higher than that of the IDIM-OLS estimate. Finally, although the NKF identification methods provide good results in the case of the TX40 robot with undecimated data (c.f. Table MCS-TX40-1-1), the performance of the estimator sharply deteriorates when fed with decimated or sub-sampled data (c.f. Table MCS-TX40-4-1 and Table MCS-RV2SQ-1-1). Nevertheless, Kalman filtering techniques generally tend –as with the CLOE method–to suffer from the same lack of sensitivity with respect to parameters' variations since the controller of the simulated robot is robust against these variations as demonstrated in [38].

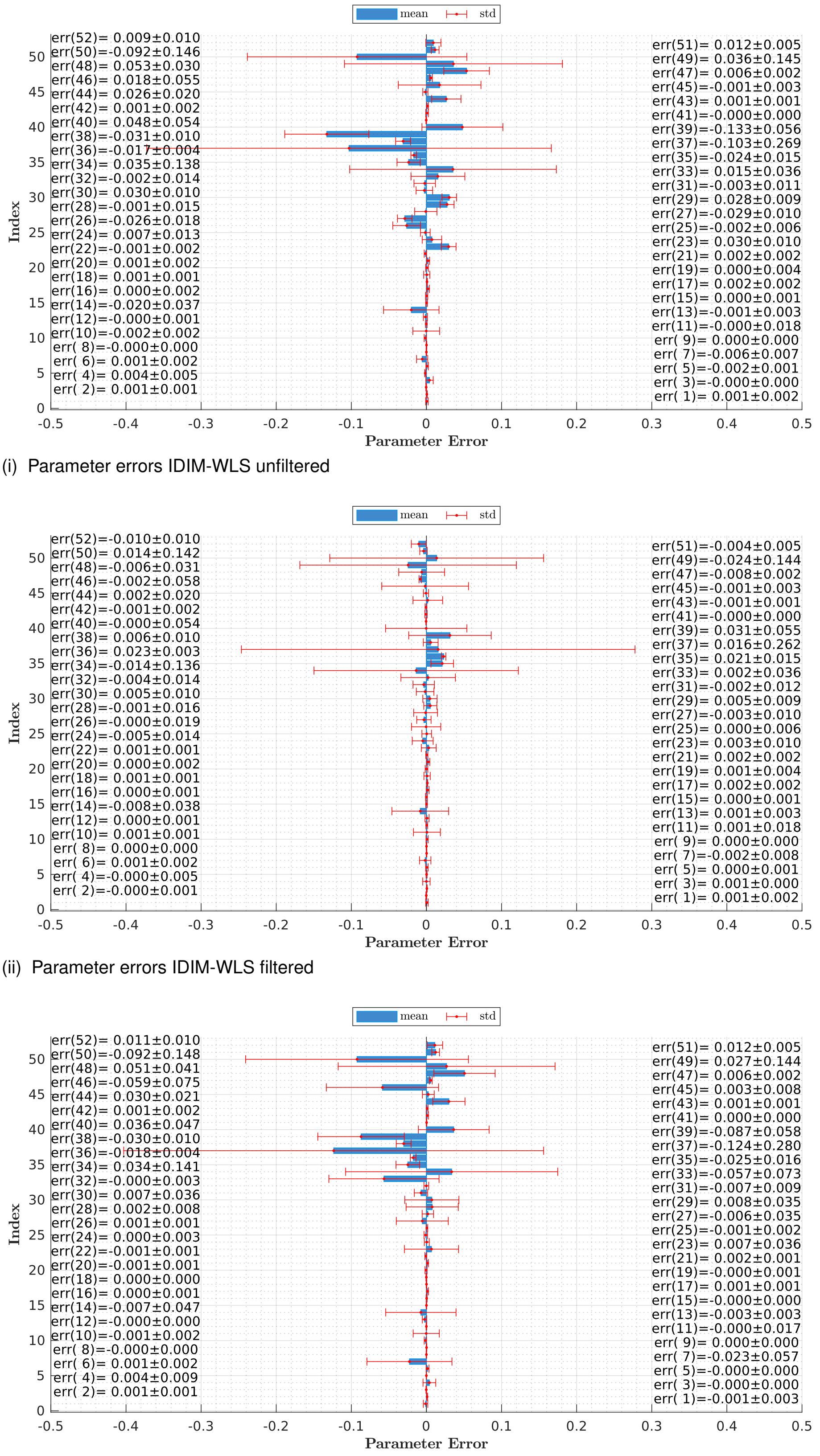
5.4.1.3 Convergence and Computational Complexity

The convergence of the DIDIM iterative process is fast, similar to IDIM-IV, with an average computation time of 1 second, as a result of 4 (resp. 3) iterations on average and as many model simulations. This is consistent with the results of [27] and [29]. Please note that this is approximately five times higher than IDIM-OLS, which can be explained by the process of DDM integration occurring during the simulation step within DIDIM and IDIM-IV. As expected, physically consistent identification methods, based on semi-defined programming algorithms, take about three times longer than their unconstrained counterparts, which is consistent with the performance of current SDP solvers (in our case CVX with MOSEK) and aligned with the results of [100]. It is worth noting that the number of iterations – and hence model recalculation – required by PC-IDIM-IV and PC-DIDIM are similar, in the case of proper convergence, to that of the unconstrained IDIM-IV and DIDIM, respectively. The computation time of CLIE and CLOE is generally about one order of magnitude higher than the unconstrained IDIM-IV and DIDIM, which is not really surprising considering the large number $E(d_{N_{sim}})$ of DDM simulations required to evaluate the Jacobian at the current estimates (c.f. the different FOM tables). Again, this is consistent with the computation time given in [66]. A careful reader will probably notice that the computation time does not scale linearly with the number of samples, in particular when making use of parallel decimation. This observation is a direct consequence of the fact that model simulation, within IDIM-IV, DIDIM, CLIE, and CLOE is carried out at the controller frequency, namely $f_c = 5$ kHz. In the case of CLIE, CLOE, and ML, the sensitivity function in the Levenberg–Marquardt optimization routine can be computed in parallel, thereby further increasing the execution speed. In our case, these computations were distributed on 12 parallel threads, which was made possible thanks to the Matlab parallel computing toolbox. Unlike the other methods, the time taken by the NKF to find a solution scales linearly with the number of samples. In the case of sigma-point Kalman filters, the whole distribution of prior estimates must be propagated through the nonlinear DDM. In our case, considering the set of 52 base parameters and the 12-dimensional robot state vector, a total of 129 sigma-points must be propagated through the DDM at each filter iteration. To speed up the computations, we took the decision to distribute the burden on four different threads. Finally, the sample propagation loop was compiled into a mex file to further accelerate the execution speed. In this manner, the time ratio between an EKF execution and a UKF execution could be reduced to 4.

5.4.2. General Discussion and Perspectives

5.4.2.1 On the choice of the identification algorithm

The present discussion is somehow related to the Epilogue of [54] where the author gives an insightful presentation on what can be considered as '*good*', '*bad*' or '*optimal*'. Even though the underlying goal of benchmarking is to *find* the best identification method, a *black and*



(i) Parameter errors IDIM-WLS unfiltered

(ii) Parameter errors IDIM-WLS filtered

(iii) Parameter errors PC-IDIM-WLS unfiltered

Figure 40 Illustration of the effects of filtering on IDIM-WLS in MCS-TX40-4-2.

white answer can seldom be obtained. Indeed, if the choice of a method often depends on the circumstances of identification, namely how the dynamic parameters of a robot can be

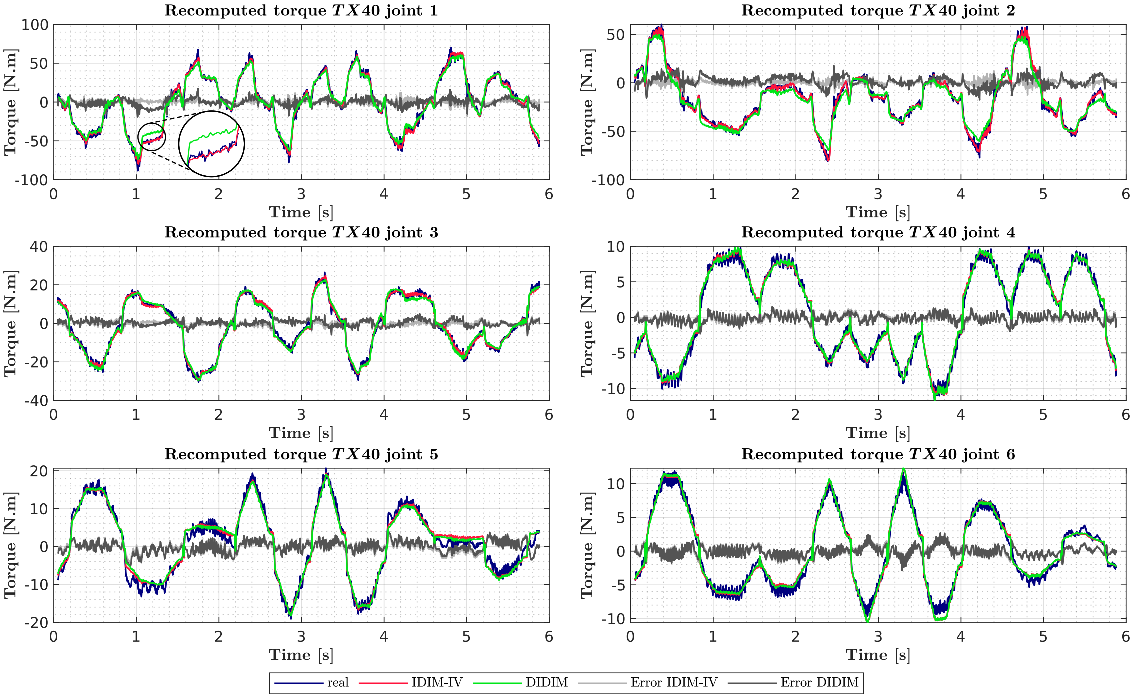


Figure 41 Reconstructed IDIM-IV and DIDIM torque signals during the validation experiment on the **real** TX40 robot.

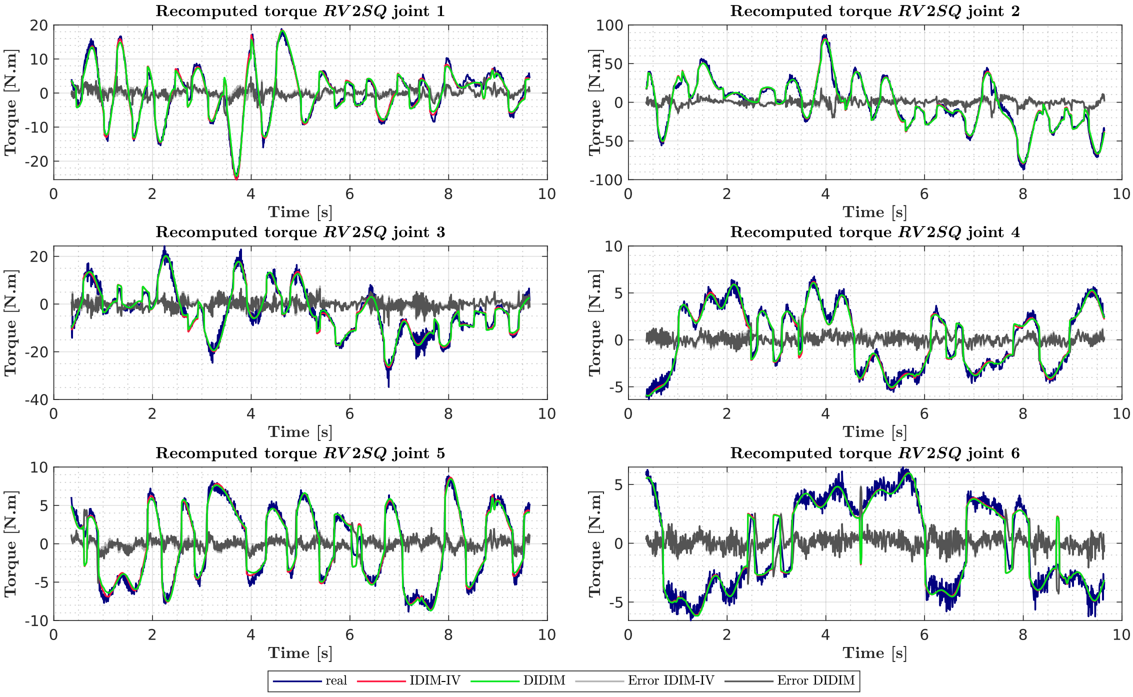


Figure 42 Reconstructed IDIM-IV and DIDIM torque signals during the validation experiment on the **real** RV2SQ robot.

identified, it may sometimes depend on the final objectives of identification, or in other words why these parameters are being identified. Based on the different Monte Carlo simulations performed in this chapter, the IDIM-IV and DIDIM methods seem to be the most appealing for offline identification of fixed-base robots, as they converge quickly, do not require custom data filtering, combine the inverse and direct dynamic models and are well-suited to the

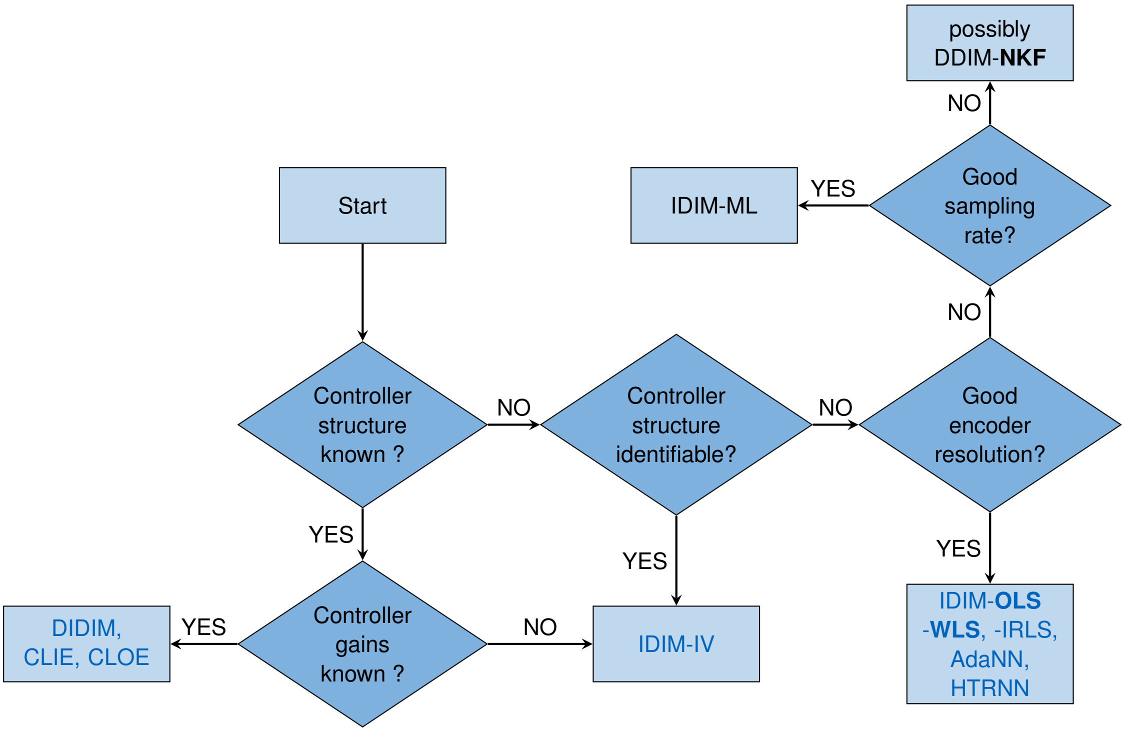


Figure 43 Decision process for parameter identification algorithm selection. Physical consistency of the estimates can be enforced using SDP within the algorithms displayed in blue. Methods in **bold** caption are, or can potentially be formulated in a recursive manner which makes them capable of running in a real-time control loop. Other methods have to be formulated in batches to enable online implementation.

enforcement of physical constraints through semi-definite programming. Since the dynamic models are validated simultaneously, a complete robot simulator can be obtained, making it possible to design an accurate model-based control. Besides the fact that these algorithms are difficult to implement in real-time control loops – thereby raising serious issues regarding their implementation in adaptive control strategies – their main limitations directly stems from the need for a precise knowledge of the robot's low-level control characteristics. This dependence is clearly observable when comparing the MCS results – where the control law is assumed to be perfectly known – and that obtained with the real systems. In practice, the incomplete knowledge regarding the true nature of the implemented low-level control loops has tangible consequences on the behavior of the internal dynamic simulation routine of IDIM-IV and DIDIM, and more generally of CLIE, CLOE and NKF. This in turn induces a bias in the parameter estimates and may even in the worst case lead to a divergence in the identification process, as the behavior of the simulation routine no longer matches that of the real system. Note that this statement can be refined by distinguishing between the ideal case where a good prior knowledge of the low-level control structure and gains is available, the case where only the control structure is known, and the case where neither the control structure nor the gains are known. In practice, knowing the structure of a controller makes it possible to identify its gains, as exposed in [39]. The fact that IDIM-OLS, -WLS, -TLS, ML, AdaNN and HTRNN do not require any prior knowledge of the low-level control characteristics, and that their performance does not depend on any initial estimate or tuning coefficients, makes them comparatively easy to implement and therefore attractive to the practitioner. It is also worth mentioning that NKF, IDIM-OLS and -WLS are, or can be formulated recursively and are therefore suitable for implementation in real-time adaptive control loops. This last point must of course be tempered by the fact that batch implementations of the remaining algorithms – except CLIE, CLOE and ML which require nonlinear optimization techniques – may also lead to real-time capable identification methods. In this context, the lack of excitability can be addressed through suitable relaxation of the problem.

As already pointed out in [38], the main drawback of Kalman filtering techniques in the context of parameter identification turns out to be their extreme sensitivity to the initial values of a set of adjustment parameters, and in particular to the values of the initial process-noise and measurement-noise covariance. In practice, the tuning process proves to be tedious although the computed torque approach would deserve to be further investigated. Therefore, in case the objective of identification is not the offline refinement of CAD parameters values for reliable simulation purpose, but rather consists in developing robust controllers based on an online rough estimation of the robot inertial parameters, the IDIM-OLS, -WLS and NKF methods, appear to be suitable choices. It should be noted that a more detailed study of the effects of different amplitudes or statistical distributions of noise, resulting for example from higher quantization errors – i.e. lower sensor resolutions – would be relevant and is here considered as future work. In terms of sampling rate, most current robot controllers operate in the 1 kHz range, which is generally sufficient to obtain good approximations of joint speeds and accelerations. Below 100 Hz, special care must be taken since the calculation of joint velocities and accelerations is no longer reliable enough. In this context, interpolation methods or the use of IDIM-IV or OE methods, based on DDM simulations – which can be considered as a kind of data interpolation – should be preferred. The previous observations can be summarized in the form of a flow diagram in Fig. 43.

5.4.2.2 Possible Improvements of the Proposed Benchmark

Multiple improvements of BIRDy are currently being investigated, with a focus on enhancing the scalability of the benchmark. We indeed noticed that the current symbolic model generation engine, based on the MuPad symbolic kernel could take considerable time to generate identification model of robots with more than 7-DOF (we so far tested the generation routine with up to 9 DOF). We moreover noticed a strong correlation of the performance of the symbolic toolbox with the version of Matlab. A number of approaches are currently being explored to tackle this issue, including the use of other symbolic kernels such as that of Wolfram Mathematica or the open-source SymPy, also used in the OpenSYMORO+ toolbox from [320, 321]. Besides these issues, future developments of BIRDy will mainly consist in generalizing the process of identification-model computation using the Unified Robot Description Format (URDF) rather than DH parameters, considering the use of modern highly-efficient dynamic libraries such as the Rigid Body Dynamics Library (RBDL [322]) or the Kinematics and Dynamics Library (KDL [323]) for model numerical simulation and eventually the inclusion of additional sensor modalities in the process of robot identification, such as accelerometers, gyroscopes or force sensors. On a longer time scale, the possibility of identifying parallel manipulators such as the stewart platforms or floating-base robots such as humanoid robots (as for instance performed in [324, 106]) will also be explored.

5.4.2.3 What an Artificial Skin could Potentially Bring in the Context of Identification

It is worth noting that the presence of an artificial skin on a robot can potentially be used for inertial parameter identification purposes. Indeed, although the presence of the skin itself modifies – sometimes considerably – the dynamic parameters of the robot to which it is attached, it is still possible, knowing its mass properties as well as its spatial distribution, to have a reasonable estimate of the new set of dynamic parameters of the robot, which in turn allows to hot-start several algorithms such as CLOE. It should also be emphasized that when a self-organizing multimodal artificial skin, such as [6], is mounted on a robot whose CAD model is not available, the shape reconstruction functionality of the skin can potentially be leveraged to enforce the physicality of the parameter identification process, following an approach similar to [101, 102, 325]. In essence, the surface of the skin covering the robot can be turned into a convex solid using a similar approach to that presented in section 2.5.4.3 in the context of self-collision avoidance. This solid can then be spatially sampled into a set of point masses and hence be used as boundaries for the considered link CoM. More importantly, when the artificial skin is equipped with accelerometers, these can potentially be used for online identification purposes, by allowing derivative-free computation of the estimates of the first and second order motion derivatives used in the calculation of the regression matrix (c.f. chapter 4). It is worth mentioning that although acceleration feedback was already leveraged for parameter identification purposes [326] in the context of a robot with elastic joints, it was so far, to the best of our knowledge, never used in the case of rigid robotic systems. Finally, the contact and pre-contact modalities of the skin could also potentially be used for identification of a robot manipulating a payload, by providing information on the location, and potentially the shape of the payload coupled to the robot.

TX40, Real, decim 4	$E(d_{\boldsymbol{q}}), \sigma_{\boldsymbol{q}}$	$E(d_{\boldsymbol{\tau}}), \sigma_{\boldsymbol{\tau}}$	$E(d_t), \sigma_t$	$E(d_{N_{it}}), \sigma_{N_{it}}$	$E(d_{N_{sim}}), \sigma_{N_{sim}}$
IDIM-OLS	0.166% (0.07%)	19.33% (14.20%)	0.38s (13.95%)	1.00 (0.00%)	N.A.
IDIM-OLS filt.	0.166% (0.07%)	19.27% (14.14%)	0.36s (4.17%)	1.00 (0.00%)	N.A.
IDIM-WLS	0.165% (0.07%)	19.43% (14.33%)	0.37s (4.42%)	1.00 (0.00%)	N.A.
IDIM-WLS filt.	0.165% (0.07%)	19.36% (14.24%)	0.36s (2.20%)	1.00 (0.00%)	N.A.
IDIM-TLS	0.172% (0.06%)	21.51% (17.42%)	0.36s (3.07%)	1.00 (0.00%)	N.A.
IDIM-TLS filt.	0.171% (0.07%)	20.84% (16.51%)	0.36s (2.81%)	1.00 (0.00%)	N.A.
IRLS	0.166% (0.07%)	19.44% (14.15%)	0.52s (6.17%)	1.00 (0.00%)	N.A.
IRLS filt.	0.166% (0.07%)	19.41% (14.12%)	0.39s (3.79%)	1.00 (0.00%)	N.A.
ML	0.165% (0.07%)	19.29% (14.06%)	7.71s (68.72%)	1.00 (0.00%)	N.A.
ML filt.	0.165% (0.07%)	19.22% (14.01%)	6.79s (38.38%)	1.00 (0.00%)	N.A.
IDIM-IV	0.166% (0.07%)	19.59% (14.64%)	1.35s (2.05%)	3.00 (0.00%)	3.00 (0.00%)
DIDIM	0.162% (0.07%)	24.90% (20.83%)	1.53s (1.28%)	4.00 (0.00%)	4.00 (0.00%)
CLIE	0.171% (0.07%)	21.66% (15.78%)	48.31s (73.27%)	4.32 (83.39%)	275 (76%)
CLOE	0.172% (0.07%)	22.69% (16.32%)	29.11s (50.13%)	6.44 (55.10%)	400 (50%)
EKF	166% (58%)	26.41% (17.10%)	1.13s (2.70%)	1.00 (0.00%)	N.A.
SREKF	199% (53%)	27.68% (17.28%)	1.72s (3.11%)	1.00 (0.00%)	N.A.
UKF	0.167% (0.09%)	25.57% (16.78%)	4.46s (4.31%)	1.00 (0.00%)	N.A.
SRUKF	0.165% (0.08%)	24.84% (16.30%)	2.25s (2.82%)	1.00 (0.00%)	N.A.
CDKF	157% (70%)	23.12% (15.46%)	3.88s (4.97%)	1.00 (0.00%)	N.A.
SRCDKF	0.167% (0.09%)	25.83% (16.97%)	2.82s (2.91%)	1.00 (0.00%)	N.A.
ANN	0.166% (0.07%)	19.14% (13.35%)	0.56s (6.99%)	1.00 (0.00%)	N.A.
ANN filt.	0.165% (0.07%)	19.14% (13.32%)	0.61s (11.60%)	1.00 (0.00%)	N.A.
HTRNN	154% (53%)	448% (415%)	0.43s (3.94%)	1.00 (0.00%)	N.A.
HTRNN filt.	166% (46%)	235% (214%)	0.44s (5.50%)	1.00 (0.00%)	N.A.
PC-IDIM-OLS	0.166% (0.07%)	19.47% (14.35%)	0.84s (4.89%)	1.00 (0.00%)	N.A.
PC-IDIM-OLS filt.	0.167% (0.07%)	19.42% (14.26%)	0.84s (5.26%)	1.00 (0.00%)	N.A.
PC-IDIM-WLS	0.166% (0.07%)	19.56% (14.38%)	0.85s (4.82%)	1.00 (0.00%)	N.A.
PC-IDIM-WLS filt.	0.166% (0.07%)	19.54% (14.34%)	0.85s (5.35%)	1.00 (0.00%)	N.A.
PC-IDIM-IRLS	0.166% (0.07%)	19.58% (14.24%)	3.99s (7.78%)	1.00 (0.00%)	N.A.
PC-IDIM-IRLS filt.	0.166% (0.07%)	19.61% (14.27%)	3.63s (7.58%)	1.00 (0.00%)	N.A.
PC-IDIM-IV	0.167% (0.07%)	19.67% (14.63%)	1.88s (3.39%)	3.00 (0.00%)	3.00 (0.00%)
PC-DIDIM	0.163% (0.07%)	23.73% (18.88%)	2.84s (15.65%)	3.76 (11.59%)	3.76 (11.59%)

Table 2 Various FoM computed for the Staubli TX40 robot, with a control frequency of $5kHz$, a sampling frequency of $1kHz$ and a decimation frequency of $100Hz$. These results are obtained using the average values $E(d)$ and standard deviations σ of the 1300 parameter estimates obtained during the experiments on the actual robot. Data is filtered using a Butterworth forward-backward filter with a cutoff frequency of $50Hz$.

RV2SQ, Real, decim 1	$E(d_{\boldsymbol{q}}), \sigma_{\boldsymbol{q}}$	$E(d_{\boldsymbol{\tau}}), \sigma_{\boldsymbol{\tau}}$	$E(d_t), \sigma_t$	$E(d_{N_{it}}), \sigma_{N_{it}}$	$E(d_{N_{sim}}), \sigma_{N_{sim}}$
IDIM-OLS	0.261% (0.14%)	18.46% (14.80%)	0.20s (46.51%)	1.00 (0.00%)	N.A.
IDIM-OLS filt.	0.261% (0.14%)	18.46% (14.80%)	0.04s (44.69%)	1.00 (0.00%)	N.A.
IDIM-WLS	0.261% (0.14%)	18.43% (14.93%)	0.06s (31.14%)	1.00 (0.00%)	N.A.
IDIM-WLS filt.	0.261% (0.14%)	18.43% (14.93%)	0.06s (35.19%)	1.00 (0.00%)	N.A.
IDIM-TLS	0.261% (0.14%)	18.84% (15.37%)	0.03s (19.02%)	1.00 (0.00%)	N.A.
IDIM-TLS filt.	0.261% (0.14%)	18.84% (15.37%)	0.04s (27.51%)	1.00 (0.00%)	N.A.
IRLS	0.261% (0.14%)	18.43% (14.88%)	0.08s (10.88%)	1.00 (0.00%)	N.A.
IRLS filt.	0.261% (0.14%)	18.43% (14.88%)	0.09s (11.17%)	1.00 (0.00%)	N.A.
ML	0.261% (0.14%)	18.49% (14.88%)	14.57s (58.66%)	1.00 (0.00%)	N.A.
ML filt.	0.261% (0.14%)	18.43% (14.76%)	11.57s (38.98%)	1.00 (0.00%)	N.A.
IDIM-IV	305% (88%)	52% (37%)	3.79s (11.16%)	9.80 (10.20%)	9.80 (10.20%)
DIDIM	225% (88%)	109% (27%)	2.22s (46.69%)	6.12 (40.39%)	6.12 (40.39%)
CLIE	0.261% (0.14%)	23.62% (17.80%)	79.15s (37.45%)	5.92 (43.04%)	362 (39%)
CLOE	342% (111%)	24.24% (18.96%)	23.76s (1.11%)	1.08 (25.64%)	110 (0.00%)
EKF	0.261% (0.14%)	20.15% (16.73%)	1.82s (6.79%)	1.00 (0.00%)	N.A.
SREKF	0.261% (0.14%)	20.10% (16.66%)	2.83s (5.63%)	1.00 (0.00%)	N.A.
UKF	0.261% (0.14%)	21.04% (17.17%)	6.90s (3.55%)	1.00 (0.00%)	N.A.
SRUKF	0.261% (0.14%)	20.77% (17.46%)	3.57s (6.42%)	1.00 (0.00%)	N.A.
CDKF	271% (81%)	23.37% (18.15%)	5.71s (4.39%)	1.00 (0.00%)	N.A.
SRCDKF	0.261% (0.14%)	20.65% (17.31%)	4.53s (5.74%)	1.00 (0.00%)	N.A.
ANN	62% (90.03%)	18.80% (14.60%)	0.26s (8.19%)	1.00 (0.00%)	N.A.
ANN filt.	0.261% (0.14%)	18.59% (14.54%)	0.27s (15.36%)	1.00 (0.00%)	N.A.
HTRNN	0.261% (0.14%)	18.46% (14.80%)	0.12s (15.36%)	1.00 (0.00%)	N.A.
HTRNN filt.	0.261% (0.14%)	18.46% (14.80%)	0.11s (19.52%)	1.00 (0.00%)	N.A.
PC-IDIM-OLS	0.261% (0.14%)	19.49% (15.49%)	0.84s (7.13%)	1.00 (0.00%)	N.A.
PC-IDIM-OLS filt.	0.261% (0.14%)	19.49% (15.49%)	0.61s (7.84%)	1.00 (0.00%)	N.A.
PC-IDIM-WLS	0.261% (0.14%)	19.63% (15.87%)	0.62s (5.82%)	1.00 (0.00%)	N.A.
PC-IDIM-WLS filt.	0.261% (0.14%)	19.63% (15.87%)	0.68s (15.05%)	1.00 (0.00%)	N.A.
PC-IDIM-IRLS	0.261% (0.14%)	19.68% (15.97%)	3.02s (16.59%)	1.00 (0.00%)	N.A.
PC-IDIM-IRLS filt.	0.261% (0.14%)	19.68% (15.97%)	3.18s (18.02%)	1.00 (0.00%)	N.A.
PC-IDIM-IV	0.261% (0.14%)	19.71% (16.08%)	1.22s (3.77%)	3.00 (0.00%)	3.00 (0.00%)
PC-DIDIM	0.261% (0.14%)	19.60% (15.77%)	1.72s (18.86%)	3.24 (13.45%)	3.24 (13.45%)

Table 3 Various FoM computed for the Mitsubishi RV2SQ robot, with a control frequency of $5kHz$, a sampling frequency of $140Hz$ and a decimation frequency of $100Hz$. These results are obtained using the average values $E(d)$ and standard deviations σ of the 1300 parameter estimates obtained during the MCS over a set of 10s experiments, with $\boldsymbol{\sigma_\tau} = 5 \cdot 10^{-2} N.m$ and $\boldsymbol{\sigma_q} = 2 \cdot 10^{-6} deg$. Data is filtered using a Butterworth forward-backward filter with a cutoff frequency of $50Hz$.

MCS-TX40-1-1	$E(d_{\boldsymbol{q}}), \sigma_{\boldsymbol{q}}$	$E(d_{\boldsymbol{\tau}}), \sigma_{\boldsymbol{\tau}}$	$E(d_t), \sigma_t$	$E(d_{N_{it}}), \sigma_{N_{it}}$	$E(d_{N_{sim}}), \sigma_{N_{sim}}$
IDIM-OLS	0.010% (0.07%)	8.02% (5.82%)	0.24s (20.43%)	1.00 (0.00%)	N.A.
IDIM-OLS filt.	0.001% (0.00%)	7.89% (5.78%)	0.25s (10.27%)	1.00 (0.00%)	N.A.
IDIM-WLS	0.002% (0.01%)	7.94% (5.80%)	0.49s (6.39%)	1.00 (0.00%)	N.A.
IDIM-WLS filt.	0.001% (0.00%)	7.90% (5.79%)	0.51s (10.55%)	1.00 (0.00%)	N.A.
IDIM-TLS	0.018% (0.11%)	8.25% (6.00%)	0.19s (12.27%)	1.00 (0.00%)	N.A.
IDIM-TLS filt.	0.005% (0.01%)	7.97% (5.85%)	0.20s (15.31%)	1.00 (0.00%)	N.A.
IRLS	0.002% (0.01%)	7.93% (5.80%)	1.03s (12.05%)	1.00 (0.00%)	N.A.
IRLS filt.	0.001% (0.00%)	7.90% (5.79%)	0.70s (9.61%)	1.00 (0.00%)	N.A.
ML	0.010% (0.07%)	8.02% (5.82%)	33.82s (26.18%)	1.00 (0.00%)	N.A.
ML filt.	0.001% (0.00%)	7.89% (5.78%)	30.46s (26.30%)	1.00 (0.00%)	N.A.
IDIM-IV	0.001% (0.00%)	7.89% (5.79%)	1.47s (8.78%)	3.00 (0.00%)	3.00 (0.00%)
DIDIM	0.001% (0.00%)	7.90% (5.79%)	1.94s (20.72%)	3.50 (14.30%)	3.50 (14.30%)
CLIE	0.006% (0.01%)	8.17% (5.88%)	20.65s (63.73%)	3.15 (76.98%)	204 (67%)
CLOE	0.008% (0.02%)	8.29% (5.93%)	25.50s (68.32%)	4.60 (75.93%)	284 (68%)
EKF	0.002% (0.01%)	7.91% (5.79%)	14.69s (10.40%)	1.00 (0.00%)	N.A.
SREKF	0.002% (0.01%)	7.91% (5.79%)	17.20s (8.35%)	1.00 (0.00%)	N.A.
UKF	0.012% (0.10%)	8.10% (5.99%)	50.87s (11.94%)	1.00 (0.00%)	N.A.
SRUKF	0.007% (0.05%)	8.02% (5.94%)	20.50s (4.36%)	1.00 (0.00%)	N.A.
CDKF	0.004% (0.01%)	8.15% (6.12%)	24.90s (21.58%)	1.00 (0.00%)	N.A.
SRCDKF	0.003% (0.01%)	7.99% (5.93%)	15.55s (2.05%)	1.00 (0.00%)	N.A.
AdaNN	414% (1194%)	8.10% (5.85%)	0.28s (8.20%)	1.00 (0.00%)	N.A.
AdaNN filt.	0.001% (0.00%)	7.90% (5.79%)	0.30s (7.84%)	1.00 (0.00%)	N.A.
HTRNN	0.010% (0.07%)	8.02% (5.82%)	0.17s (9.16%)	1.00 (0.00%)	N.A.
HTRNN filt.	0.001% (0.00%)	7.89% (5.79%)	0.19s (8.51%)	1.00 (0.00%)	N.A.
PC-IDIM-OLS	0.004% (0.01%)	7.99% (5.78%)	2.33s (4.71%)	1.00 (0.00%)	N.A.
PC-IDIM-OLS filt.	0.003% (0.01%)	8.02% (5.79%)	2.36s (3.54%)	1.00 (0.00%)	N.A.
PC-IDIM-WLS	0.003% (0.01%)	8.01% (5.79%)	2.60s (3.66%)	1.00 (0.00%)	N.A.
PC-IDIM-WLS filt.	0.003% (0.00%)	8.03% (5.80%)	2.59s (3.58%)	1.00 (0.00%)	N.A.
PC-IDIM-IRLS	0.003% (0.01%)	8.01% (5.80%)	9.15s (32.58%)	1.00 (0.00%)	N.A.
PC-IDIM-IRLS filt.	0.003% (0.00%)	8.03% (5.80%)	7.02s (38.34%)	1.00 (0.00%)	N.A.
PC-IDIM-IV	0.003% (0.01%)	8.03% (5.80%)	1.97s (4.77%)	3.00 (0.00%)	3.00 (0.00%)
PC-DIDIM	0.003% (0.01%)	8.03% (5.80%)	5.42s (21.50%)	3.41 (14.43%)	3.41 (14.43%)

Table 4 Various FoM computed for the Staubli TX40 model, with a control frequency of $5kHz$, a sampling frequency of $1kHz$ and a decimation frequency of $500Hz$. These results are obtained using the average values $E(d)$ and standard deviations σ of the 1300 parameter estimates obtained during the MCS over a set of 10s experiments, with $\boldsymbol{\sigma_\tau} = 5 \cdot 10^{-2} N.m$ and $\boldsymbol{\sigma_q} = 2 \cdot 10^{-6} deg$. Data is filtered using a Butterworth forward-backward filter with a cutoff frequency of $50Hz$.

MCS-TX40-4-1	$E(d_{\boldsymbol{q}}), \sigma_{\boldsymbol{q}}$	$E(d_{\boldsymbol{\tau}}), \sigma_{\boldsymbol{\tau}}$	$E(d_t), \sigma_t$	$E(d_{N_{it}}), \sigma_{N_{it}}$	$E(d_{N_{sim}}), \sigma_{N_{sim}}$
IDIM-OLS	0.001% (0.00%)	7.89% (5.79%)	0.60s (12.33%)	1.00 (0.00%)	N.A.
IDIM-OLS filt.	0.001% (0.00%)	7.89% (5.79%)	0.59s (5.20%)	1.00 (0.00%)	N.A.
IDIM-WLS	0.001% (0.00%)	7.89% (5.79%)	0.71s (5.96%)	1.00 (0.00%)	N.A.
IDIM-WLS filt.	0.001% (0.00%)	7.90% (5.79%)	0.73s (7.48%)	1.00 (0.00%)	N.A.
IDIM-TLS	0.001% (0.00%)	7.90% (5.80%)	0.62s (6.37%)	1.00 (0.00%)	N.A.
IDIM-TLS filt.	0.002% (0.00%)	7.90% (5.80%)	0.60s (10.77%)	1.00 (0.00%)	N.A.
IRLS	0.001% (0.00%)	7.89% (5.79%)	0.68s (12.45%)	1.00 (0.00%)	N.A.
IRLS filt.	0.001% (0.00%)	7.90% (5.79%)	0.68s (11.85%)	1.00 (0.00%)	N.A.
ML	0.003% (0.01%)	7.91% (5.76%)	7.72s (24.42%)	1.00 (0.00%)	N.A.
ML filt.	0.003% (0.01%)	7.91% (5.76%)	7.88s (24.40%)	1.00 (0.00%)	N.A.
IDIM-IV	0.001% (0.00%)	7.89% (5.79%)	1.99s (7.40%)	3.00 (0.00%)	3.00 (0.00%)
DIDIM	0.001% (0.00%)	7.90% (5.79%)	2.39s (7.57%)	4.00 (0.00%)	4.00 (0.00%)
CLIE	0.006% (0.01%)	8.13% (5.86%)	45.61s (60.06%)	3.97 (73.23%)	254 (63%)
CLOE	0.009% (0.02%)	8.29% (5.92%)	29.76s (62.35%)	5.73 (68.47%)	344 (62%)
EKF	0.011% (0.01%)	8.37% (5.73%)	3.78s (6.34%)	1.00 (0.00%)	N.A.
SREKF	0.011% (0.02%)	8.34% (5.74%)	8.03s (5.00%)	1.00 (0.00%)	N.A.
UKF	442% (1015%)	10.41% (6.25%)	20.04s (10.90%)	1.00 (0.00%)	N.A.
SRUKF	0.005% (0.01%)	7.99% (5.75%)	7.38s (2.12%)	1.00 (0.00%)	N.A.
CDKF	429% (984%)	9.10% (5.91%)	11.51s (1.83%)	1.00 (0.00%)	N.A.
SRCDKF	0.006% (0.01%)	8.02% (5.75%)	9.13s (4.43%)	1.00 (0.00%)	N.A.
AdaNN	0.001% (0.00%)	7.90% (5.79%)	0.87s (2.32%)	1.00 (0.00%)	N.A.
ANN filtered	0.001% (0.00%)	7.89% (5.79%)	0.88s (1.75%)	1.00 (0.00%)	N.A.
HTRNN	0.019% (0.06%)	9.07% (6.56%)	0.57s (8.49%)	1.00 (0.00%)	N.A.
HTRNN filt.	0.020% (0.06%)	9.05% (6.54%)	0.53s (6.98%)	1.00 (0.00%)	N.A.
PC-IDIM-OLS	0.003% (0.01%)	8.01% (5.79%)	1.29s (4.71%)	1.00 (0.00%)	N.A.
PC-IDIM-OLS filt.	0.003% (0.01%)	8.02% (5.79%)	1.29s (4.67%)	1.00 (0.00%)	N.A.
PC-IDIM-WLS	0.003% (0.01%)	8.03% (5.80%)	1.36s (4.94%)	1.00 (0.00%)	N.A.
PC-IDIM-WLS filt.	0.003% (0.00%)	8.02% (5.79%)	1.35s (4.98%)	1.00 (0.00%)	N.A.
PC-IDIM-IRLS	0.003% (0.01%)	8.04% (5.81%)	3.06s (36.66%)	1.00 (0.00%)	N.A.
PC-IDIM-IRLS filt.	0.003% (0.00%)	8.03% (5.80%)	2.87s (33.21%)	1.00 (0.00%)	N.A.
PC-IDIM-IV	0.003% (0.01%)	8.03% (5.80%)	3.29s (6.94%)	3.00 (0.00%)	3.00 (0.00%)
PC-DIDIM	0.003% (0.01%)	8.03% (5.80%)	4.12s (7.10%)	3.95 (5.35%)	3.95 (5.35%)

Table 5 Various FoM computed for the Staubli TX40 model, with a control frequency of $5kHz$, a sampling frequency of $1kHz$ and a decimation frequency of $100Hz$. These results are obtained using the average values $E(d)$ and standard deviations σ of the 1300 parameter estimates obtained during the MCS over a set of 10s experiments, with $\boldsymbol{\sigma_\tau} = 5 \cdot 10^{-2} N.m$ and $\boldsymbol{\sigma_q} = 2 \cdot 10^{-6} deg$. Data is filtered using a Butterworth forward-backward filter with a cutoff frequency of $50Hz$.

MCS-RV2SQ-1-1	$E(d_{\boldsymbol{q}}), \sigma_{\boldsymbol{q}}$	$E(d_{\boldsymbol{\tau}}), \sigma_{\boldsymbol{\tau}}$	$E(d_t), \sigma_t$	$E(d_{N_{it}}), \sigma_{N_{it}}$	$E(d_{N_{sim}}), \sigma_{N_{sim}}$
IDIM-OLS	244% (133%)	13.66% (9.54%)	0.06s (27.91%)	1.00 (0.00%)	N.A.
IDIM-OLS filt.	206% (160%)	13.67% (9.55%)	0.06s (22.57%)	1.00 (0.00%)	N.A.
IDIM-WLS	0.004% (0.00%)	12.15% (9.41%)	0.09s (22.29%)	1.00 (0.00%)	N.A.
IDIM-WLS filt.	0.004% (0.00%)	12.16% (9.41%)	0.09s (10.86%)	1.00 (0.00%)	N.A.
IDIM-TLS	281% (97%)	17.47% (12.14%)	0.05s (9.30%)	1.00 (0.00%)	N.A.
IDIM-TLS filt.	270% (91%)	17.47% (12.14%)	0.05s (9.38%)	1.00 (0.00%)	N.A.
IRLS	0.004% (0.00%)	11.55% (9.07%)	0.19s (11.21%)	1.00 (0.00%)	N.A.
IRLS filt.	0.004% (0.00%)	11.56% (9.07%)	0.19s (9.59%)	1.00 (0.00%)	N.A.
ML	211% (160%)	13.63% (9.54%)	7.97s (29.11%)	1.00 (0.00%)	N.A.
ML filt.	207% (160%)	13.64% (9.55%)	8.64s (27.34%)	1.00 (0.00%)	N.A.
IDIM-IV	0.006% (0.01%)	13.65% (10.25%)	1.16s (8.86%)	3.96 (4.86%)	3.96 (4.86%)
DIDIM	0.003% (0.00%)	10.85% (8.90%)	2.17s (30.83%)	6.80 (26.66%)	6.80 (26.66%)
CLIE	0.004% (0.00%)	10.85% (8.86%)	31.43s (17.95%)	5.87 (18.91%)	364 (16.69%)
CLOE	0.003% (0.00%)	10.83% (8.86%)	45.01s (3.28%)	9.02 (1.46%)	530 (1.32%)
EKF	0.004% (0.00%)	12.21% (9.41%)	1.64s (7.36%)	1.00 (0.00%)	N.A.
SREKF	0.004% (0.00%)	12.21% (9.41%)	3.29s (6.53%)	1.00 (0.00%)	N.A.
UKF	0.004% (0.00%)	12.21% (9.43%)	8.47s (3.84%)	1.00 (0.00%)	N.A.
SRUKF	0.004% (0.00%)	12.24% (9.42%)	4.04s (5.83%)	1.00 (0.00%)	N.A.
CDKF	0.005% (0.00%)	12.40% (9.46%)	10.34s (26.43%)	1.00 (0.00%)	N.A.
SRCDKF	291% (114%)	12.26% (9.37%)	7.32s (26.02%)	1.00 (0.00%)	N.A.
AdaNN	0.005% (0.00%)	12.04% (8.85%)	0.81s (1.83%)	1.00 (0.00%)	N.A.
AdaNN filt.	0.005% (0.00%)	12.23% (8.92%)	0.71s (31.05%)	1.00 (0.00%)	N.A.
HTRNN	104% (73%)	13.48% (9.40%)	0.28s (15.99%)	1.00 (0.00%)	N.A.
HTRNN filt.	211% (160%)	13.49% (9.40%)	0.24s (39.32%)	1.00 (0.00%)	N.A.
PC-IDIM-OLS	0.007% (0.01%)	13.65% (9.42%)	1.60s (26.12%)	1.00 (0.00%)	N.A.
PC-IDIM-OLS filt.	0.007% (0.01%)	13.66% (9.43%)	1.65s (23.05%)	1.00 (0.00%)	N.A.
PC-IDIM-WLS	0.004% (0.00%)	12.06% (9.32%)	0.57s (8.07%)	1.00 (0.00%)	N.A.
PC-IDIM-WLS filt.	0.004% (0.00%)	12.06% (9.32%)	0.58s (7.69%)	1.00 (0.00%)	N.A.
PC-IDIM-IRLS	0.004% (0.00%)	11.50% (9.05%)	3.36s (2.14%)	1.00 (0.00%)	N.A.
PC-IDIM-IRLS filt.	0.004% (0.00%)	11.50% (9.05%)	3.47s (2.94%)	1.00 (0.00%)	N.A.
PC-IDIM-IV	0.005% (0.01%)	12.86% (9.57%)	5.23s (37.02%)	3.42 (14.44%)	3.42 (14.44%)
PC-DIDIM	0.003% (0.00%)	10.82% (8.86%)	5.00s (30.33%)	6.73 (26.08%)	6.73 (26.08%)

Table 6 Various FoM computed for the Mitsubishi RV2SQ model, with a control frequency of $5kHz$, a sampling frequency of $140Hz$ and a decimation frequency of $100Hz$. These results are obtained using the average values $E(d)$ and standard deviations σ of the 1300 parameter estimates obtained during the MCS over a set of 10s experiments, with $\boldsymbol{\sigma_{\tau}} = 5 \cdot 10^{-2} N.m$ and $\boldsymbol{\sigma_q} = 2 \cdot 10^{-6} rad$. Data is filtered using a Butterworth forward-backward filter with a cutoff frequency of $50Hz$.

5.5. Summary

In this chapter, a sample of the most popular approaches to inertial parameter identification for fixed-based robotic systems is evaluated and benchmarked. These methods are IDIM-OLS, IDIM-WLS, IDIM-IRLS, IDIM-TLS, ML, IDIM-IV, DIDIM, CLIE, CLOE, EKF, SREKF, UKF, SRUKF, CDKF, SRCDKF, AdaNN, HTRNN, PC-IDIM-OLS, PC-IDIM-WLS, PC-IDIM-IRLS, PC-DIDIM and PC-IDIM-IV. Each method is implemented and evaluated experimentally within a dedicated framework, named BIRDy, which was specifically developed for this purpose. BIRDy features a complete identification pipeline, allowing one to generate the kinematic and dynamic models of a given robot, to compute a trajectory that excites its dynamic parameters, to simulate the system's behavior along this trajectory, collect experimental data under well-defined conditions, proceed to parameter identification using a pool of dedicated algorithms and eventually to compare the identification performances using a set of suitable metrics. In this work, we used BIRDy to perform Monte Carlo simulations on two models of 6-DoF industrial robot manipulators, namely the Staubli TX40 and the Mitsubishi RV2SQ. Experiments were also carried out on the real robots, thereby providing helpful insight on the influence of multiple factors, including prior knowledge of the control architecture, sampling frequency, or friction model. The results allow to provide a set of general guidelines based on quantitative arguments regarding the applicability of a given identification method to a particular experimental context.

Chapter 6

Online Physically-Consistent Parameter Identification: Leveraging Multimodal Tactile Feedback

This chapter introduces a concept of online physically consistent identification routine, triggered by a multimodal artificial skin. The proposed identification algorithm leverages an alternative formulation of the robot base inertial parameters in order to provide physically consistent estimates without explicitly formulating the constraints. As a result, the complete physically consistent identification problem can be formulated as a sequence of Quadratic Programs (SQP) and solved online on the robot. The Riemannian structure of the parameter physicality set is leveraged for both relaxation and interpolation purposes. The artificial skin covering the robot enables relevant simplifications of the identification problem in the context of whole-body manipulation and can moreover be used for hot-starting the SQP identification process. This chapter is structured as follows: Section 6.1 introduces the motivation an related works. Section 6.2 provides a short introduction to some of the main concepts of Riemannian geometry and on their applications to the field of physically consistent inertial parameter identification. Section 6.3 then presents the proposed identification method and discusses how a self-calibrating artificial robot skin can be leveraged for both offline and online identification purposes. Convergence and performance of the proposed identification algorithm are investigated in section 6.4 using the BIRDy identification framework. Finally section 6.5 describes the experiments executed on a real humanoid robotic system.

6.1. Leveraging Distributed Tactile Feedback for Online Identification and Adaptive Stabilization Purposes

6.1.1. Motivation and Related Works

To become part of human society, humanoid robots must be endowed with human-like levels of adaptability, implying in particular the capability of adjusting their control or stabilization strategy based on the context in which they operate. This inevitably raises multiple challenges, not only in terms of control design, but also in terms of perception and planning. The fundamental principles underlying adaptive control of robot manipulators were formulated in the late 1980s. Two main families of adaptive controllers have been proposed, namely passivity-based adaptive controllers, pioneered by Slotine and Li [327, 328], and adaptive computed torque controllers, proposed by Craig et al. [329]. In both cases, the core component of the adaptive control strategy proves to be a reliable and fast-converging online inertial parameter identification loop. Significant research efforts have been deployed in the fields of parameter identification and adaptive control of floating-base mechanisms, such as humanoids. Important results have in particular been obtained by Venture, Ayusawa, Bonnet, Jovic et al. [110, 109, 20, 118] in the field of identification, demonstrating in particular that joint torque measurements were not mandatory to identify a floating-base mechanism as long as the external wrenches applied to the system were measurable. Conversely adaptive control strategies essentially focused on robust torque control applications as for instance Pucci et al. [330]. Position- and torque-controlled floating-base systems being naturally unstable and prone to tilt, they require a robust *stabilization loop* and *trajectory generator*, both relying on an accurate knowledge of the robot's center of mass [331, 332] as well as its linear and angular momenta [333, 334, 335]. Most stabilization strategies currently assume that the robot model is known. This can of course be guaranteed through suitable offline identification, as for instance, proposed by Jovic et al. [20] or Bonnet et al. [118]. Nevertheless it must be emphasized that adaptive stabilization and trajectory generation prove to be especially relevant in the context of a manipulation task, to the extent that a robot might experience substantial parameter changes that may eventually jeopardize its stability or result in unfeasible trajectories. When a floating-base robot passively interacts with its environment, the forces resulting from this interaction are usually considered as disturbances, and hence intended to be absorbed by the stabilizer. On the contrary, when the robot behaves proactively with its environment, for instance during a manipulation task, it becomes relevant to identify the dynamic coupling that may occur with its payload. This coupling can then potentially be exploited for enhanced stabilization or control purposes. Online estimation of the CoM and momentum has recently received special attention from multiple research teams around the world. Most approaches take the form of a tailor-made observer, leveraging a simplified model of the robot dynamics. For instance Rotella et al. [336] proposed an observer to capable of estimating the CoM as well as the momenta based on a CoM dynamic model. Using the ZMP-CoM dynamics, Piperakis et al. [337] formulated a CoM and contact force estimator while Masuya et al. [338] proposed using a Kalman filter for CoM estimation purposes. Carpentier et al. [339] proposed an approach to estimate the CoM of a walking mechanism using a complementary filter. Comparatively little research has considered the identification of the complete robot dynamics for CoM and momentum estimation purposes. Although such approaches offer the possibility of constraining the parameter estimates, which in turn proves relevant in the context of control or manipulation, the development of a real-time capable formulation often turns out to be problematic for computational complexity reasons. In this context, it should however be noted that Mori et al. [340], proposed a recursive least squares approach for online identification and CoM estimation of a humanoid robot coupled to an unknown load. Although this approach has been shown to work in real time with good performance, the authors have reported a number of issues related to the lack of excitation of some parameters. In this context, a set of novel techniques such as geometric relaxation [294, 103] appear to be promising research directions for the development of more robust estimators. It should also be emphasized that the physicality of the identified parameters was not taken into account.

6.1.2. Proposed Approach and Contributions

In this chapter, we propose a new algorithm to identify the inertial parameters of a floating-base robot in an efficient and physically consistent manner. The use of an alternative formulation of the base inertial parameters makes it possible to provide physically consistent parameter estimates without having to explicitly state the physicality constraints in the form of LMIs. As a result, it becomes possible to reformulate the whole physically consistent identification problem as a sequence of quadratic programs (SQP), that can be solved – potentially in real time – on an embedded robot computer. This new algorithm will be referred to as PC-IDIM-SQP in the following sections. This identification routine is executed in adjustable batches within a dedicated thread, spinning on the robot. In order to avoid abrupt changes in the values of the identified inertial parameters transmitted to the controller, we propose interpolating the estimates at each control epoch along a geodesic of the Riemannian manifold of physicality. Combined with suitable geometric relaxation of the problem, this makes it possible to alleviate the effects of data rank deficiency. In addition the use of an artificial skin makes it possible, to hot-start the sequential identification process and to determine the location of the coupling between the robot and its payload, thereby leading to relevant simplifications of the identification problem. The proposed approach is validated both in simulation – using the previously developed BIRDy identification benchmark – and on a real REEM-C humanoid robot handling an unknown payload.

6.2. Fundamentals of Riemannian Geometry in the Context of Inertial Parameter Identification

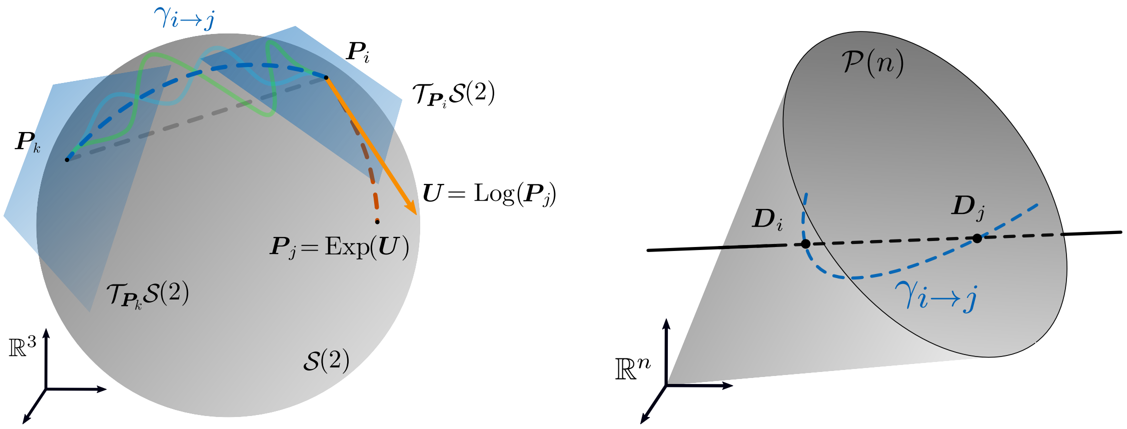
This section introduces a set of key concepts in the field of Riemannian geometry and discusses about their potential uses in the context of inertial parameter identification, in particular for relaxation and interpolation purposes.

6.2.1. Fundamental Concepts of Riemannian Geometry

6.2.1.1 Riemannian Manifolds, Metrics and Tangent Spaces

As defined in Jaquier et al. [341], a n-dimensional manifold $\mathcal{M}$ is a Hausdorff topological space[1] that has the notable property of being locally Euclidean. As a matter of fact, any point $\boldsymbol{P}$ located on such manifold has an open neighborhood $\mathcal{N}_{\boldsymbol{P}}$ which is *homeomorphic* to an open subset of the Euclidean space $\mathbb{R}^n$. In other words there exists a smooth bijective map $\phi_{\boldsymbol{P}}$ – referred to as a chart or coordinate system – between the neighborhood $\mathcal{N}_{\boldsymbol{P}}$ of point $\boldsymbol{P}$ and an open subset of $\mathbb{R}^n$. We call *tangent space* $\mathcal{T}_{\boldsymbol{P}}\mathcal{M}$ in a point $\boldsymbol{P}$ of a manifold $\mathcal{M}$ the space formed by the tangent vectors to all 1-dimensional curves on $\mathcal{M}$ passing through $\boldsymbol{P}$. An intuitive example of 2-dimensional manifold is that of the surface of a unit 3D sphere embedded in $\mathbb{R}^3$ and defined as $\mathcal{S}(2) = \left\{\boldsymbol{x} \in \mathbb{R}^3 : \ \boldsymbol{x}^\top\boldsymbol{x} = 1\right\}$ (c.f. Fig. 44i). In this context,

[1] A Hausdorff space is a topological space such that there exists, for any two distinct points $\boldsymbol{P}_i$ and $\boldsymbol{P}_j$, neighborhoods $\mathcal{N}_{\boldsymbol{P}_i}$ and $\mathcal{N}_{\boldsymbol{P}_j}$ that are disjoint from each other.



$\gamma_{i\to j}$
$\boldsymbol{P}_i$
$\mathcal{T}_{\boldsymbol{P}_i}\mathcal{S}(2)$
$\boldsymbol{P}_k$
$\boldsymbol{U} = \operatorname{Log}(\boldsymbol{P}_j)$
$\boldsymbol{P}_j = \operatorname{Exp}(\boldsymbol{U})$
$\mathcal{T}_{\boldsymbol{P}_k}\mathcal{S}(2)$
$\mathbb{R}^3$
$\mathcal{S}(2)$
$\mathcal{P}(n)$
$\boldsymbol{D}_i$
$\boldsymbol{D}_j$
$\gamma_{i\to j}$
$\mathbb{R}^n$

(i) Surface of a 3D sphere seen as a 2D manifold $\mathcal{S}(2) = \{\boldsymbol{x} \in \mathbb{R}^3 : \boldsymbol{x}^\top \boldsymbol{x} = 1\}$. $\forall \boldsymbol{P} \in \mathcal{S}(2)$, $\mathcal{T}_{\boldsymbol{P}}\mathcal{S}(2)$ is depicted as a blue plane. The Euclidean and Riemannian geodesic curves are depicted in dashed black and dashed blue respectively.

(ii) Illustration of the physical consistency Riemannian manifold $\mathcal{P}(n)$. The Euclidean and Riemannian geodesic curves between the points $\boldsymbol{D}_i$ and $\boldsymbol{D}_j$ are depicted in dashed black and dashed blue respectively.

Figure 44 Illustration of Riemannian manifolds

the tangent space $\mathcal{T}_{\boldsymbol{P}}\mathcal{S}(2)$ in a point $\boldsymbol{P}$ is defined as a 2D plane intersecting the manifold in $\boldsymbol{P}$, as depicted in blue within Fig. 44i. A Riemannian manifold, denoted as ($\mathcal{M}$, $\langle \cdot, \cdot \rangle_{\boldsymbol{P}}$) is a structure consisting of a real, smooth – i.e. differentiable – manifold $\mathcal{M}$ equipped in any point $\boldsymbol{P}$ with a smoothly-varying positive-definite inner product $\langle \cdot, \cdot \rangle_{\boldsymbol{P}} : \mathcal{T}_{\boldsymbol{P}}\mathcal{M} \times \mathcal{T}_{\boldsymbol{P}}\mathcal{M} \to \mathbb{R}$ (referred to as a *Riemannian metric*) defined on the tangent space $\mathcal{T}_{\boldsymbol{P}}\mathcal{M}$.

6.2.1.2 Geodesics on a Riemannian Manifold

The norm of a vector $\boldsymbol{U}$ defined on the tangent space $\mathcal{T}_{\boldsymbol{P}}\mathcal{M}$ of a Riemannian manifold $\mathcal{M}$ can be computed using the inner product $\|\boldsymbol{U}\|^2_{\boldsymbol{P}} = \langle \boldsymbol{U}, \boldsymbol{U} \rangle_{\boldsymbol{P}}$ in $\boldsymbol{P}$. By extension, the length of a curve $\boldsymbol{c}(t)$ defined on ($\mathcal{M}$, $\langle \cdot, \cdot \rangle_{\boldsymbol{c}(t)}$) can be computed by accounting for the smooth variation of the Riemannian metric along this curve, yielding

$$\boldsymbol{c}(t) : \mathbb{R} \to \mathcal{M} \qquad l(\boldsymbol{c}(t)) = \int_0^1 \|\dot{\boldsymbol{c}}(t)\|_{\boldsymbol{c}(t)}\, dt \tag{158}$$

Accordingly, the Riemannian distance between two points $\boldsymbol{P}_i \equiv \boldsymbol{c}(t = 0)$ and $\boldsymbol{P}_j \equiv \boldsymbol{c}(t = 1)$ of a Riemannian manifold ($\mathcal{M}$, $\langle \cdot, \cdot \rangle_{\boldsymbol{P}}$) can be defined as the minimal length, over all possible smooth curves joining these two points on the manifold, namely

$$\boldsymbol{d}_{\mathcal{M}}(\boldsymbol{P}_i, \boldsymbol{P}_j) = \min_{\boldsymbol{c}_{i\to j}(t)} l(\boldsymbol{c}_{i\to j}(t)) \tag{159}$$

The curves $\gamma_{i\to j}(t)$ resulting in extremal elements of the length function $l(\boldsymbol{c}_{i\to j}(t))$ are referred to as *geodesics*. Such curves are for instance visible as dashed blue or dashed orange on the 2D sphere manifold $\mathcal{S}(2)$ depicted on Fig.44i, as opposed to the green and light blue curves connecting the points $\boldsymbol{P}_i$ and $\boldsymbol{P}_k$. For a given manifold $\mathcal{M}$, one can define the so called exponential map $\operatorname{Exp}_{\boldsymbol{P}_i} : \mathcal{T}_{\boldsymbol{P}_i}\mathcal{M} \to \mathcal{M}$ in $\boldsymbol{P}_i$ as the operation transforming any point $\boldsymbol{U}$ of the tangent space $\mathcal{T}_{\boldsymbol{P}_i}\mathcal{M}$ into an element $\boldsymbol{P}_j$ of the manifold $\mathcal{M}$ such that $\boldsymbol{P}_j$ is reached at time $t = 1$ by the geodesic starting in $\boldsymbol{P}_i$ in the direction $\boldsymbol{U}$ (c.f. Fig.44i). The corresponding inverse map is referred to as the Logarithm map $\operatorname{Log}_{\boldsymbol{P}_i} : \mathcal{M} \to \mathcal{T}_{\boldsymbol{P}_i}\mathcal{M}$ [341].

6.2.1.3 The Manifold of Symmetric Positive-Definite Matrices

Let $\mathbb{S}^{k\times k} = \left\{\boldsymbol{A} \in \mathbb{R}^{k\times k} : \boldsymbol{A} = \boldsymbol{A}^\top\right\}$ be the space of real symmetric matrices of dimension $k\times k$ and $\mathbb{S}^{k\times k}_{++} = \left\{\boldsymbol{A} \in \mathbb{S}^{k\times k} : \ \forall \boldsymbol{x} \neq \boldsymbol{0}_n \ \boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} > 0\right\}$ be the space of real symmetric positive definite matrices of dimension $k \times k$. As a matter of fact, $\mathbb{S}^{k\times k}_{++}$ can be defined as the interior of an opened convex cone of $\mathbb{R}^n$ (c.f. Chapter 2.3 and Fig. 44ii), yielding

- $\forall \boldsymbol{A}, \boldsymbol{B} \in \mathbb{S}^{k\times k}_{++}, 0 < t < 1, \ t\boldsymbol{A} + (1-t)\boldsymbol{B} \in \mathbb{S}^{k\times k}_{++}$ (*convexity*)
- $\forall \boldsymbol{A} \in \mathbb{S}^{k\times k}_{++}, t > 0, \ t\boldsymbol{A} \in \mathbb{S}^{k\times k}_{++}$ (*cone structure*)
- The boundary of $\mathbb{S}^{k\times k}_{++}$ is the set of singular positive semi-definite matrices.

It can be shown [342, 343, 341] that $\mathbb{S}^{k\times k}_{++}$ is a smooth manifold of dimension[2] $k(k+1)/2$ whose tangent space $\mathcal{T}_{\boldsymbol{D}}\mathbb{S}^{k\times k}_{++} = \{\boldsymbol{D}\} \times \mathbb{S}^{k\times k}$ in any point $\boldsymbol{D} \in \mathbb{S}^{k\times k}_{++}$ is often identified with $\mathbb{S}^{k\times k}$ for simplicity. Endowing $\mathbb{S}^{k\times k}_{++}$ with an inner product $\langle\cdot,\cdot\rangle_{\boldsymbol{D}}$ on $\mathcal{T}_{\boldsymbol{D}}\mathbb{S}^{k\times k}_{++}$, smoothly depending on the base point $\boldsymbol{D}$ such that

$$\forall \boldsymbol{U}, \boldsymbol{V} \in \mathcal{T}_{\boldsymbol{D}}\mathbb{S}^{k\times k}_{++}, \quad \langle \boldsymbol{U}, \boldsymbol{V}\rangle_{\boldsymbol{D}} = \operatorname{tr}(\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{U}\boldsymbol{D}^{-1}\boldsymbol{V}\boldsymbol{D}^{-\frac{1}{2}}) \tag{160}$$

allows considering the structure $(\mathbb{S}^{k\times k}_{++}, \langle\cdot,\cdot\rangle_{\boldsymbol{D}})$ as a Riemannian manifold, denoted as $\mathcal{P}(k)$. It should be highlighted that such metric is referred to as the *affine-invariant* metric in [342, 343, 344, 341]. The corresponding logarithm and exponential operators can be formulated [341] as

$$\begin{aligned} \operatorname{Exp}_{\boldsymbol{D}} : \mathcal{T}_{\boldsymbol{D}}\mathcal{M} &\to \mathcal{M} \\ \boldsymbol{A} &\to \boldsymbol{D}^{\frac{1}{2}} \exp\left(\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}}\right)\boldsymbol{D}^{\frac{1}{2}} \end{aligned} \tag{161}$$

$$\begin{aligned} \operatorname{Log}_{\boldsymbol{D}} : \mathcal{M} &\to \mathcal{T}_{\boldsymbol{D}}\mathcal{M} \\ \boldsymbol{B} &\to \boldsymbol{D}^{\frac{1}{2}} \log\left(\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{B}\boldsymbol{D}^{-\frac{1}{2}}\right)\boldsymbol{D}^{\frac{1}{2}} \end{aligned} \tag{162}$$

where $\forall \boldsymbol{X} \in \mathbb{R}^{k\times k}, \exp(\boldsymbol{X}) = \sum_{i=0}^{\infty} \frac{\boldsymbol{X}^i}{i!}$ and $\log(\boldsymbol{X}) = -\sum_{i=1}^{\infty} \frac{(\boldsymbol{1}_{k\times k} - \boldsymbol{X})^i}{i}$ denote the matrix exponential and logarithm maps, respectively. In this context, the geodesic distance between two points $\boldsymbol{D}_i$ and $\boldsymbol{D}_j$ of $\mathcal{P}(k)$ can be computed as

$$\boldsymbol{d}_{\mathcal{P}(k)}(\boldsymbol{D}_i, \boldsymbol{D}_j) = \left\|\operatorname{Log}_{\boldsymbol{D}_i}(\boldsymbol{D}_i^{-\frac{1}{2}}\boldsymbol{D}_j\boldsymbol{D}_i^{-\frac{1}{2}})\right\|_F = \left[\sum_{i=1}^{k} \log^2 \lambda_i(\boldsymbol{D}_i^{-\frac{1}{2}}\boldsymbol{D}_j\boldsymbol{D}_i^{-\frac{1}{2}})\right]^{\frac{1}{2}} \tag{163}$$

where $\lambda_i(\boldsymbol{X})$ refers to the – positive – eigenvalues of the square matrix $\boldsymbol{X}$. The reader is referred to [344] for a more detailed derivation. It is worth noting that the distance $\boldsymbol{d}_{\mathcal{P}(k)}(\boldsymbol{D}_i, \boldsymbol{D}_j)$ goes to infinity if any of the matrices $\boldsymbol{D}_i$ or $\boldsymbol{D}_j$ approaches the boundary of the $\mathcal{P}(k)$ cone (i.e. if it is "close" to be singular positive semi-definite). Following the work of Lee et al. [103], the minimal geodesic curve $\gamma_{i\to j} : [0,1] \to \mathcal{P}(k)$ connecting two arbitrary points $\boldsymbol{D}_i, \boldsymbol{D}_j \in \mathcal{P}(k)$

[2] The manifold dimension refers to the number of entries needed to parameterize the system. A total of $k(k+1)/2$ entries are necessary to parameterize a $k \times k$ symmetric matrix.

of the cone of physical consistency (c.f. Fig.44ii), can be defined as

$$\gamma(t) = \boldsymbol{D}_i^{\frac{1}{2}} \left(\boldsymbol{D}_i^{-\frac{1}{2}} \boldsymbol{D}_j \boldsymbol{D}_i^{-\frac{1}{2}} \right)^t \boldsymbol{D}_i^{\frac{1}{2}} \tag{164}$$

6.2.2. Leveraging the Riemannian Structure of the Inertial Parameters Physicality Manifold

As discussed in chapter 2.3, the physicality conditions related to the inertial parameters of a robot link j can be formulated as a positivity definition constraint on the corresponding pseudo-inertia matrix $\boldsymbol{D}'_{Lj}(\boldsymbol{\chi})$, yielding

$$\boldsymbol{D}'_{Lj}(\boldsymbol{\chi}) = \begin{bmatrix} \underbrace{\frac{1}{2}\operatorname{tr}(\boldsymbol{L}_j)\mathbf{1}_{3\times 3} - \boldsymbol{L}_j}_{\tilde{\boldsymbol{L}}_j} & \boldsymbol{m}\boldsymbol{p}_j \\ \boldsymbol{m}\boldsymbol{p}_j^\top & M_j \end{bmatrix} \succ 0. \tag{165}$$

As originally noted by Lee et al. [294, 104, 103, 311], the space of physically consistent inertial parameters of a given robot link, endowed with an affine-invariant metric can be considered as a Riemannian manifold of dimension 4, denoted as $\mathcal{P}(4)$. The possibility of specifying a Riemannian distance $\boldsymbol{d}_{\mathcal{P}(k)}(\boldsymbol{D}'_{Lj}(\boldsymbol{\chi}_1), \boldsymbol{D}'_{Lj}(\boldsymbol{\chi}_2))$ between two points $\boldsymbol{D}'_{Lj}(\boldsymbol{\chi}_1)$ and $\boldsymbol{D}'_{Lj}(\boldsymbol{\chi}_2)$ of this manifold – via eq. (163) – enables enhanced relaxation techniques within inertial parameter identification algorithms, as specifically discussed in [104]. In this context, it is worth noting that for an n-link robotic system, the distance metric $\boldsymbol{d}_{\mathcal{P}^n(4)}$ can be obtained by summation

$$\boldsymbol{d}_{\mathcal{P}^n(4)}(\boldsymbol{\chi}_1, \boldsymbol{\chi}_2)^2 = \sum_{i=1}^{n} \boldsymbol{d}_{\mathcal{P}(4)}(\boldsymbol{D}'_{Li}(\boldsymbol{\chi}_1), \boldsymbol{D}'_{Li}(\boldsymbol{\chi}_2))^2. \tag{166}$$

Proper relaxation of an identification problem turns out to be of crucial importance in practice, since most inertial parameter identification problems tend to be ill-conditioned. This is especially true in the case of hyper-redundant floating base mechanisms – such as humanoid robots – under low excitability conditions. The main drawback of this technique stems from the non-linearity of the Riemannian metric, thereby making the use of computationally expensive optimization-on-manifold techniques inevitable. Nevertheless, it is worth emphasizing that a suitable linearization of the Riemannian distance enables implementation of geometric relaxation within classical identification approaches such as IDIM-LS. Lee et al. [103, 311] in particular proposed two approximation of the geodesic distance between two sets of physically consistent parameters in $\mathcal{P}(4)$, referred to as the *entropic divergence* and *constant pullback metric*. The entropic divergence is a second order approximation of the Riemannian metric, valid for two infinitesimally close points of a manifold. The constant pullback metric is a constant Riemannian metric, which is evaluated at a given inertial parameter value χ_0 as

$$\boldsymbol{d}_0(\boldsymbol{\chi}_1, \boldsymbol{\chi}_2) = \boldsymbol{d}_0(\boldsymbol{D}'_{Lj}(\boldsymbol{\chi}_1), \boldsymbol{D}'_{Lj}(\boldsymbol{\chi}_2)) = \frac{1}{2}\operatorname{tr}\left(\boldsymbol{D}'^{-1}_{Lj}(\boldsymbol{\chi}_0)\boldsymbol{D}'^{2}_{Lj}(\boldsymbol{\chi}_1 - \boldsymbol{\chi}_2)\right) \tag{167}$$

eventually yielding SDP formulations similar to (64) for physically consistent identification.

6.3. Online Physically-Consistent Identification of Floating Base Robots

Halfway between classical – offline – and recursive formulations of the identification problem, we here propose a new physically consistent batch identification algorithm. The data sampled in real time by the robot is used to fill a reduced-size, potentially adjustable, buffered observation system. The identification process is executed in a loop within a dedicated thread, as soon as this buffer is full. As a result, the parameters are updated online, but at frequencies one to two orders of magnitude lower than that of the controller. In this context, it is important to underline that a use of the estimated parameters in the robot's control loop, especially for adaptive stabilization purposes, must take into account the potentially strong discontinuities between the different parameter estimates, in order to prevent potentially hazardous behaviors. In practice, it is therefore essential to both regularize and filter the parameter estimates passed to the controller, while guaranteeing their physicality. Hence the second contribution of this chapter, where a new interpolation method between the different parameter estimates, leveraging the underlying Riemannian structure of the physical consistency manifold is formulated, based on the work of Lee et al. Combined with geometric relaxation, the proposed method provides robots with a robust, physically consistent estimate of their inertial parameters at each control epoch. The final contribution of this work consists in leveraging the tactile feedback from an artificial skin covering a robot to hot-start the online identification process, both in the context of classical identification and adaptive manipulation.

6.3.1. Physical Consistency through Sequential Quadratic Programming with Geometric Relaxation

6.3.1.1 Alternative Parametrization for an Implicit Formulation of Physicality

In contrast to standard physically consistent identification methods such as optimization on manifold or Semi-Definite Programming (SDP) with LMI constraints, Sutanto et al. [297] proposed a set of alternative parametrizations, allowing to directly enforce physicality of the reconstructed dynamic parameters without the need for advanced constrained optimization techniques. Accordingly, it becomes possible to enforce the constraints in (59) by adopting a parametrization $\boldsymbol{\varrho}_j$, based a the Cholesky decomposition of $\boldsymbol{D}'_{Lj}(\boldsymbol{\chi})$

$$\boldsymbol{D}'_{Lj}(\boldsymbol{\chi}) = \underbrace{\begin{bmatrix} C_{j_{11}} & 0 & 0 & 0 \\ C_{j_{21}} & C_{j_{22}} & 0 & 0 \\ C_{j_{31}} & C_{j_{32}} & C_{j_{33}} & 0 \\ C_{j_{41}} & C_{j_{42}} & C_{j_{43}} & C_{j_{44}} \end{bmatrix}}_{\boldsymbol{C}_j} \underbrace{\begin{bmatrix} C_{j_{11}} & C_{j_{21}} & C_{j_{31}} & C_{j_{41}} \\ 0 & C_{j_{22}} & C_{j_{32}} & C_{j_{42}} \\ 0 & 0 & C_{j_{33}} & C_{j_{43}} \\ 0 & 0 & 0 & C_{j_{44}} \end{bmatrix}}_{\boldsymbol{C}_j^\top} + \epsilon \mathbf{1}_{4\times 4}, \tag{168}$$

$$\boldsymbol{\varrho}_j = [C_{j_{11}}\ C_{j_{21}}\ C_{j_{22}}\ C_{j_{31}}\ C_{j_{32}}\ C_{j_{33}}\ C_{j_{41}}\ C_{j_{42}}\ C_{j_{43}}\ C_{j_{44}}]^\top \in \mathbb{R}^{10} \tag{169}$$

where the diagonal terms $C_{j_{kk}} > 0,\ \forall k = \{1, \cdots, 4\}$ of $\boldsymbol{C}_j$ should be positive to guarantee the strict positivity of $\boldsymbol{D}'_{Lj}(\boldsymbol{\chi})$. The dynamics of each robot link j being parametrized by the vector $\boldsymbol{\chi}_j = [M_j,\ MX_j,\ MY_j,\ MZ_j,\ XX_j,\ XY_j,\ XZ_j,\ YY_j,\ YZ_j,\ ZZ_j]^\top \in \mathbb{R}^{10}$, the parameter physicality condition (165) can be expressed as

$$\boldsymbol{D}'_{Lj}(\boldsymbol{\chi}) = \begin{bmatrix} \frac{YY_j+ZZ_j-XX_j}{2} & -XY_j & -XZ_j & MX_j \\ -XY_j & \frac{XX_j-YY_j+ZZ_j}{2} & -YZ_j & MY_j \\ -XZ_j & -YZ_j & \frac{XX_j+YY_j-ZZ_j}{2} & MZ_j \\ MX_j & MY_j & MZ_j & M_j \end{bmatrix} \succ 0. \tag{170}$$

From (168)-(169), it then becomes possible to write

$$\boldsymbol{D}'_{Lj}(\boldsymbol{\varrho}) = \begin{bmatrix} C_{j_{11}}^2+\epsilon & C_{j_{11}}C_{j_{21}} & C_{j_{11}}C_{j_{31}} & C_{j_{11}}C_{j_{41}} \\ C_{j_{11}}C_{j_{21}} & C_{j_{21}}^2+C_{j_{22}}^2+\epsilon & C_{j_{21}}C_{j_{31}}+C_{j_{22}}C_{j_{32}} & C_{j_{21}}C_{j_{41}}+C_{j_{22}}C_{j_{42}} \\ C_{j_{11}}C_{j_{31}} & C_{j_{21}}C_{j_{31}}+C_{j_{22}}C_{j_{32}} & C_{j_{31}}^2+C_{j_{32}}^2+C_{j_{33}}^2+\epsilon & C_{j_{31}}C_{j_{41}}+C_{j_{32}}C_{j_{42}}+C_{j_{33}}C_{j_{43}} \\ C_{j_{11}}C_{j_{41}} & C_{j_{21}}C_{j_{41}}+C_{j_{22}}C_{j_{42}} & C_{j_{31}}C_{j_{41}}+C_{j_{32}}C_{j_{42}}+C_{j_{33}}C_{j_{43}} & C_{j_{41}}^2+C_{j_{42}}^2+C_{j_{43}}^2+C_{j_{44}}^2+\epsilon \end{bmatrix} \succ 0, \tag{171}$$

thereby yielding:

$$\boldsymbol{\chi}_j = \begin{bmatrix} M_j \\ MX_j \\ MY_j \\ MZ_j \\ XX_j \\ XY_j \\ XZ_j \\ YY_j \\ YZ_j \\ ZZ_j \end{bmatrix} = \begin{bmatrix} C_{j_{41}}^2+C_{j_{42}}^2+C_{j_{43}}^2+C_{j_{44}}^2+\epsilon \\ C_{j_{11}}C_{j_{41}} \\ C_{j_{21}}C_{j_{41}}+C_{j_{22}}C_{j_{42}} \\ C_{j_{31}}C_{j_{41}}+C_{j_{32}}C_{j_{42}}+C_{j_{33}}C_{j_{43}} \\ C_{j_{21}}^2+C_{j_{22}}^2+C_{j_{31}}^2+C_{j_{32}}^2+C_{j_{33}}^2+2\epsilon \\ -C_{j_{11}}C_{j_{21}} \\ -C_{j_{11}}C_{j_{31}} \\ C_{j_{11}}^2+C_{j_{31}}^2+C_{j_{32}}^2+C_{j_{33}}^2+2\epsilon \\ -C_{j_{21}}C_{j_{31}}-C_{j_{22}}C_{j_{32}} \\ C_{j_{11}}^2+C_{j_{21}}^2+C_{j_{22}}^2+2\epsilon \end{bmatrix}. \tag{172}$$

Since from (173) the standard inertial parameter vector $\boldsymbol{\chi}$ can be expressed as a function of the vector $\boldsymbol{\varrho}$ of square-root inertial robot parameters, defined as $\boldsymbol{\varrho} = [\boldsymbol{\varrho}_1^\top \, \boldsymbol{\varrho}_2^\top \cdots \boldsymbol{\varrho}_n^\top]^\top \in \mathbb{R}^p$, one eventually gets, by construction, a set of physically consistent standard inertial parameter estimates. Physicality is in this case formulated in an implicit manner, thereby allowing physically consistent identification using classic unconstrained optimization methods such as gradient descent. It is worth emphasizing that the positivity constraint on the diagonal terms of $\boldsymbol{C}_j$ is here enforced by adding a tailor-made fraction $\epsilon << 1$ of the identity matrix to (168). This method was successfully applied in the context of machine learning, for Deep Neural Network training purposes (see for instance [345, 346, 347] and the references therein) but was – to the best of our knowledge – never applied to real floating-base robotic systems.

6.3.1.2 Sequential Quadratic Programming for Online Physically Consistent Inertial Parameter Identification

In this work, we propose to make use of sequential batch quadratic programming in order to formally account for the positivity constraint on the diagonal terms of the $\boldsymbol{C}_j$ matrix. Although a direct identification of the components of the $\boldsymbol{\varrho}$ vector using standard algorithms, such as PC-IDIM-LS, proves unfeasible in practice due to the non-linearity of the observation system in the square-root inertial robot parameters $\boldsymbol{\varrho}$, the quadratic nature of this non-linearity can be leveraged in order to simplify the identification problem. As a matter of fact, introducing a matrix $\boldsymbol{A}_j(\boldsymbol{\varrho}) \in \mathbb{R}^{10\times10}$ depending linearly on $\boldsymbol{\varrho}$, by solving (168) for the elements of the standard parameter vector $\boldsymbol{\chi}$, allows one reformulating the quadratic dependence of the IDIM-LS observation system (15) in the square-root inertial parameters as $\boldsymbol{A}_j(\boldsymbol{\varrho})\boldsymbol{\varrho}_j = \boldsymbol{\chi}_j$, or alternatively $\mho\boldsymbol{A}(\boldsymbol{\varrho})\boldsymbol{\varrho} = \boldsymbol{\beta}$ where $\mho$ denotes the base parameter mapping introduced in (10):

$$\underbrace{\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & C_{j_{41}} & C_{j_{42}} & C_{j_{43}} & C_{j_{44}} \\ 0 & 0 & 0 & 0 & 0 & 0 & C_{j_{11}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & C_{j_{21}} & C_{j_{22}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & C_{j_{31}} & C_{j_{32}} & C_{j_{33}} & 0 \\ 0 & C_{j_{21}} & C_{j_{22}} & C_{j_{31}} & C_{j_{32}} & C_{j_{33}} & 0 & 0 & 0 & 0 \\ 0 & -C_{j_{11}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -C_{j_{11}} & 0 & 0 & 0 & 0 & 0 & 0 \\ C_{j_{11}} & 0 & 0 & C_{j_{31}} & C_{j_{32}} & C_{j_{33}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -C_{j_{21}} & -C_{j_{22}} & 0 & 0 & 0 & 0 & 0 \\ C_{j_{11}} & C_{j_{21}} & C_{j_{22}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}}_{\boldsymbol{A}_j(\boldsymbol{\varrho})} \underbrace{\begin{bmatrix} C_{j_{11}} \\ C_{j_{21}} \\ C_{j_{22}} \\ C_{j_{31}} \\ C_{j_{32}} \\ C_{j_{33}} \\ C_{j_{41}} \\ C_{j_{42}} \\ C_{j_{43}} \\ C_{j_{44}} \end{bmatrix}}_{\boldsymbol{\varrho}_j} = \underbrace{\begin{bmatrix} C_{j_{41}}^2 + C_{j_{42}}^2 + C_{j_{43}}^2 + C_{j_{44}}^2 \\ C_{j_{11}} C_{j_{41}} \\ C_{j_{21}} C_{j_{41}} + C_{j_{22}} C_{j_{42}} \\ C_{j_{31}} C_{j_{41}} + C_{j_{32}} C_{j_{42}} + C_{j_{33}} C_{j_{43}} \\ C_{j_{21}}^2 + C_{j_{22}}^2 + C_{j_{31}}^2 + C_{j_{32}}^2 + C_{j_{33}}^2 \\ -C_{j_{11}} C_{j_{21}} \\ -C_{j_{11}} C_{j_{31}} \\ C_{j_{11}}^2 + C_{j_{31}}^2 + C_{j_{32}}^2 + C_{j_{33}}^2 \\ -C_{j_{21}} C_{j_{31}} - C_{j_{22}} C_{j_{32}} \\ C_{j_{11}}^2 + C_{j_{21}}^2 + C_{j_{22}}^2 \end{bmatrix}}_{\boldsymbol{\chi}_j} . \tag{173}$$

It should be emphasized that the matrix $\boldsymbol{A}_j(\boldsymbol{\varrho})$ is invertible provided that $C_{j_{kk}} \neq 0,\ \forall k = \{1, \cdots, 4\}$, that its inverse can be formulated analytically as a function of $\boldsymbol{\varrho}$ and that it has a well defined sparsity pattern. Building the matrix $\boldsymbol{A}(\boldsymbol{\varrho}) \in \mathbb{R}^{p \times p}$ by diagonal block concatenation of the different $\boldsymbol{A}_j(\boldsymbol{\varrho})$ matrices then yields

$$\boldsymbol{y}_\tau = \boldsymbol{W}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q})\boldsymbol{A}(\boldsymbol{\varrho})\boldsymbol{\varrho} + \boldsymbol{\varepsilon} . \tag{174}$$

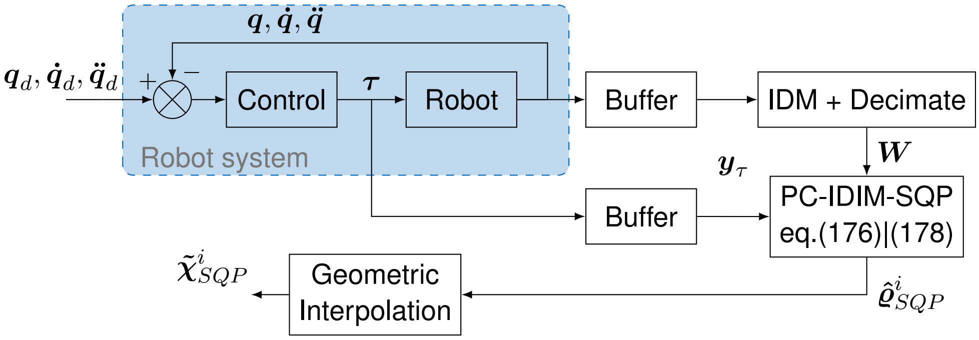
The key concept of the proposed approach consists in reformulating the identification problem (174) as an iterative process by making the approximation that the changes between two consecutive estimates $\hat{\boldsymbol{\varrho}}^{\boldsymbol{i-1}}$ and $\hat{\boldsymbol{\varrho}}^{\boldsymbol{i}}$ obtained at iterations $i-1$ and i, are small enough, thereby yielding

$$\boldsymbol{A}(\hat{\boldsymbol{\varrho}}^{\boldsymbol{i}})\hat{\boldsymbol{\varrho}}^{\boldsymbol{i}} \simeq \boldsymbol{A}(\hat{\boldsymbol{\varrho}}^{\boldsymbol{i-1}})\hat{\boldsymbol{\varrho}}^{\boldsymbol{i}} . \tag{175}$$

Under these conditions, the identification process can be formulated as a sequence of linear least squares problems, while the physicality constraints can be reduced to a positivity constraint on the terms of $\hat{\boldsymbol{\varrho}}^{\boldsymbol{i}}$ corresponding to the diagonal elements of the Cholesky factor $\hat{\boldsymbol{C}}$ of the associated pseudo-inertia matrix $\hat{\boldsymbol{D}}'_L(\boldsymbol{\chi})$. As a result, the estimate $\hat{\boldsymbol{\varrho}}^i$ at iteration i can be obtained by resolving the following QP:

$$\begin{aligned} \hat{\boldsymbol{\varrho}}^i = & \arg\min_{\boldsymbol{\varrho}} \left(\boldsymbol{W}\boldsymbol{A}(\boldsymbol{\varrho}^{i-1})\boldsymbol{\varrho} - \boldsymbol{y}_\tau\right)^\top \boldsymbol{\Sigma}^{-1} \left(\boldsymbol{W}\boldsymbol{A}(\boldsymbol{\varrho}^{i-1})\boldsymbol{\varrho} - \boldsymbol{y}_\tau\right) + \underbrace{\lambda \boldsymbol{d}\left(\boldsymbol{A}(\boldsymbol{\varrho}^{i-1})\boldsymbol{\varrho} - \boldsymbol{\chi}_0\right)^2}_{\text{Geometric relaxation}} \\ s.t. \quad & \boldsymbol{S}\boldsymbol{\varrho} > \boldsymbol{0}_{4n} \\ & \boldsymbol{B}\boldsymbol{A}(\boldsymbol{\varrho}^{i-1})\boldsymbol{\varrho} \leq \boldsymbol{b} \\ & \boldsymbol{\varrho}^{i-1} - \varpi \left|\boldsymbol{\varrho}^{i-1}\right| \leq \boldsymbol{\varrho} \leq \boldsymbol{\varrho}^{i-1} + \varpi \left|\boldsymbol{\varrho}^{i-1}\right| \end{aligned} \tag{176}$$

where $\boldsymbol{\chi}_0$ refers to the initial value of the set of standard parameters, $\boldsymbol{S} \in \mathbb{R}^{4n \times p}$ is a selection matrix for the diagonal terms of the $\hat{\boldsymbol{C}}_j$ Cholesky factors, $\lambda \in \mathbb{R}_+$ is a relaxation gain and $\varpi \in \mathbb{R}_+$ is a tuning factor. The matrix $\boldsymbol{B} \in \mathbb{R}^{N_C \times p}$ and the vector $\boldsymbol{b} \in \mathbb{R}^{N_C}$ denote a set of N_C additional constraints acting on the parameter estimate. It was decided to make use of the geometric relaxation method developed by Lee et al. [104, 103] to avoid data rank deficiencies of the identification system, resulting from improper excitation. The function



$\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$
$\boldsymbol{q}_d, \dot{\boldsymbol{q}}_d, \ddot{\boldsymbol{q}}_d$
+
−
Control
$\boldsymbol{\tau}$
Robot
Robot system
Buffer
IDM + Decimate
$\boldsymbol{W}$
$\boldsymbol{y}_{\boldsymbol{\tau}}$
PC-IDIM-SQP
eq.(176)|(178)
Buffer
$\tilde{\boldsymbol{\chi}}^i_{SQP}$
Geometric
Interpolation
$\hat{\boldsymbol{\varrho}}^i_{SQP}$

Figure 45 Block diagram of the PC-IDIM-SQP identification method.

$\boldsymbol{d}(\cdot,\cdot)$ refers to the constant pullback[3] approximation of the Riemannian affine-invariant distance between two points on the physical consistency manifold $\mathcal{P}(4)^n$. The elements of the covariance matrix $\boldsymbol{\Sigma}$ can be estimated in the very same manner as in IDIM-WLS, following (17). The resulting algorithm can be initialized using the elements of the Cholesky decomposition of the matrices $\boldsymbol{D}'_{Lj}(\boldsymbol{\chi}_0)$, computed for each robot link j using an initial estimate $\boldsymbol{\chi}_0$ of the inertial parameters (obtained, for instance, from CAD data). It is worth noting that the recursive nature of this algorithm makes it particularly suitable for an IRLS implementation. In this case, similarly to (18) one has to define a weight vector $\boldsymbol{\upsilon}^i \in \mathbb{R}^{n \cdot N}$ and the weight matrix $\boldsymbol{\Upsilon}^i = [\boldsymbol{\upsilon}^i, \cdots, \boldsymbol{\upsilon}^i] \in \mathbb{R}^{(n \cdot N) \times b}$ in order to penalize the problem outliers, such that at iteration i

$$\begin{aligned} \boldsymbol{W}^{*i} &= \boldsymbol{\Upsilon}^i \circ \boldsymbol{W} \\ \boldsymbol{y}_{\boldsymbol{\tau}}^{*i} &= \boldsymbol{\upsilon}^i \circ \boldsymbol{y}_{\boldsymbol{\tau}} \end{aligned} \tag{177}$$

where the operator $\circ$ refers to the Hadamard product. As a result, the recursive SQP identification problem (176) can be reformulated as

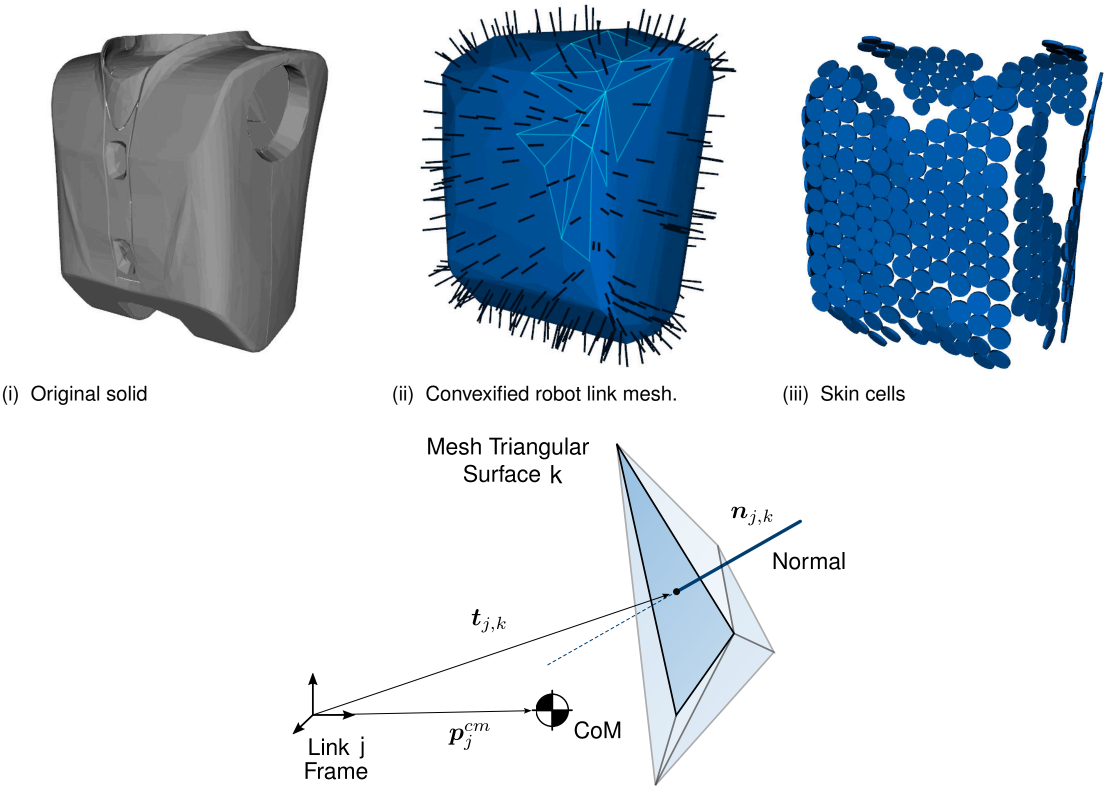
$$\begin{aligned} \hat{\boldsymbol{\varrho}}^i =& \arg\min_{\boldsymbol{\varrho}} \left(\boldsymbol{W}^{*i-1}\boldsymbol{A}(\boldsymbol{\varrho}^{i-1})\boldsymbol{\varrho} - \boldsymbol{y}_{\boldsymbol{\tau}}^{*i-1}\right)^\top \boldsymbol{\Sigma}^{-1} \left(\boldsymbol{W}^{*i-1}\boldsymbol{A}(\boldsymbol{\varrho}^{i-1})\boldsymbol{\varrho} - \boldsymbol{y}_{\boldsymbol{\tau}}^{*i-1}\right) \cdots \\ & \qquad\qquad\qquad \cdots + \underbrace{\lambda \boldsymbol{d}\left(\boldsymbol{A}(\boldsymbol{\varrho}^{i-1})\boldsymbol{\varrho} - \boldsymbol{\chi}_0\right)^2}_{\text{Geometric relaxation}} \\ s.t. \quad & \boldsymbol{S}\boldsymbol{\varrho} > \boldsymbol{0}_{4n} \\ & \boldsymbol{B}\boldsymbol{A}(\boldsymbol{\varrho}^{i-1})\boldsymbol{\varrho} \leq \boldsymbol{b} \\ & \boldsymbol{\varrho}^{i-1} - \varpi \left|\boldsymbol{\varrho}^{i-1}\right| \leq \boldsymbol{\varrho} \leq \boldsymbol{\varrho}^{i-1} + \varpi \left|\boldsymbol{\varrho}^{i-1}\right| \end{aligned} \tag{178}$$

where $\varpi \in \mathbb{R}_+$ is a tuning factor and where similar to chapter 2.2.1 the weight vector $\boldsymbol{\upsilon}^i$ is updated as

$$\boldsymbol{\upsilon}^i = \mathrm{Min}\left(\boldsymbol{\upsilon}^{i-1}, \Lambda(\boldsymbol{y}_{\boldsymbol{\tau}}^{*i-1} - \boldsymbol{W}^{*i-1}\boldsymbol{A}(\boldsymbol{\varrho}^{i-1})\hat{\boldsymbol{\varrho}}^i)\right) \tag{179}$$

the $\mathrm{Min}(\cdot)$ operator depicting the element wise $\min$ function and $\Lambda : \mathbb{R}^{n \cdot N} \to [0,1]^{n \cdot N}$ being a tailor made weight function (typically a window function). It is worth emphasizing that the term ϖ can be annealed as the number of iterations increases. This turns out to be of utmost importance since it allows enforcing convergence even under adversarial noise conditions.

[3] Unlike the entropic divergence relaxation, the constant pullback relaxation can be formulated linearly in the standard parameters and is therefore more convenient for implementation within a QP.



(i) Original solid

(ii) Convexified robot link mesh.

(iii) Skin cells

(iv) CoM bound constraint formulation.

Figure 46 Illustration of the CoM bounding process. The dark blue segments on subfigure (ii) denote the normals of each simplex while the cyan segments denote the borders of a selected subset of mesh simplexes.

This specific point will be further discussed within the MCS experiment section.

6.3.1.3 Constraining the Position of the Identified Link CoM

In this work, a set of convex bounding-box constraints are set to act on the CoM of each robot link as a security feature to prevent unsuitable solutions. Such constraints can for instance be implemented using sub-sampled convexified volumes of the different robot links, obtained following a process similar to that described in Chapter 2.5.4.3. Accordingly, each robot link j is associated with a convex triangular mesh containing N_T^j surfaces. The QP formulation of the proposed identification algorithm enables accounting for each planar surface of the considered mesh as an actual position bound acting on the CoM through the matrix $\boldsymbol{B}$ within eq. (176) and eq. (178), yielding

$$\begin{aligned}
\boldsymbol{B} &= \begin{bmatrix} \boldsymbol{B}_1 & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \ddots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{B}_n \end{bmatrix} \in \mathbb{R}^{N_C \times p} | \, \forall j = \{1 \cdots n\}\,, \; \boldsymbol{B}_j = \underbrace{\begin{bmatrix} 0 & \boldsymbol{n}_{j,1}^\top & \boldsymbol{0}_{1\times 6} \\ \vdots & \vdots & \vdots \\ 0 & \boldsymbol{n}_{j,N_T^j}^\top & \boldsymbol{0}_{1\times 6} \end{bmatrix}}_{\boldsymbol{\chi}_j} \in \mathbb{R}^{N_T^j \times 10} \\
\boldsymbol{b} &= \begin{bmatrix} \boldsymbol{b}_1 \\ \vdots \\ \boldsymbol{b}_n \end{bmatrix} \in \mathbb{R}^{N_C} | \, \forall j = \{1 \cdots n\}\,, \; b_j = M_j \begin{bmatrix} \boldsymbol{n}_{j,1}^\top \boldsymbol{t}_{j,1} \\ \vdots \\ \boldsymbol{n}_{j,N_T^j}^\top \boldsymbol{t}_{j,N_T^j} \end{bmatrix} \in \mathbb{R}^{N_T^j} \qquad (180)
\end{aligned}$$

In this expression, the term $\boldsymbol{n}_{j,k} \in \mathbb{R}^3$ denotes the normal vector of constraint surface k on robot link j (expressed in the link frame) depicted in dark blue within Fig. 46ii and Fig. 46. The term $\boldsymbol{t}_{j,k} \in \mathbb{R}^3$ denotes an arbitrary point on the constraint surface k. The link parameter vector is denoted as

$$\boldsymbol{\chi}_j = [M_j,\, MX_j,\, MY_j,\, MZ_j,\, XX_j,\, XY_j,\, XZ_j,\, YY_j,\, YZ_j,\, ZZ_j]^\top , \tag{181}$$

hence the sparse structure of the matrix $\boldsymbol{B}_j$, in (180), targeting the terms $MX_j,\ MY_j$ and MZ_j. Finally the term $\boldsymbol{p}_j^{cm} = [X_j\ Y_j\ Z_j]^\top = \frac{1}{M_j}[MX_j\ MY_j\ MZ_j]^\top$ denotes the position of the link CoM expressed in the link frame as depicted in Fig. 46iv. This work leverages the recursive nature of the proposed identification algorithm and set M_j at each iteration as the link mass identified at the previous iteration. This holds provided that the steps are small enough.

6.3.1.4 Inertial Parameter Smoothing using Geometric Interpolation

Although computationally efficient, online batch identification techniques often prove to be ill-conditioned, due in particular to the limited number of data samples in the considered batch, and to the fact that the executed behaviors do not necessarily excite every modality of the robot dynamics over the considered time horizon. In this context, besides existing relaxation techniques (described in chapter 2.3) allowing to steer the unexcited parameters towards a known reference – often obtained from CAD data – it might be desirable to implement some form of long term memory, allowing a robot to accumulate experience beyond the limited time horizon of the batch without, however, the need to solve a problem of constraining dimensions. In practice, this long term memory can take the form of a low-pass filter formulated at the parameter level. It is worth emphasizing that such parameter filtering approach might also prove relevant for other reasons, since the real-time control thread and the batch identification thread running on the robot are expected to spin at significantly different frequencies (by at least an order of magnitude). Therefore, it can be desirable in a context of adaptive stabilization, to filter the estimated parameter values sent to the controller in order to prevent abrupt changes that may potentially lead to unstable behaviors or tilting[4]. It should be noted that although the FIFO batch treatment of the identification problem – namely the fact of only renewing part of the identification batch at each execution of the identification routine – can be considered as a form of moving average low-pass filter, the fact that the batch used for identification is only renewed in a block-wise manner translates, in practice, into “jumps” within the estimated values. Nonetheless, it remains possible to smooth out these discontinuities using suitable interpolation techniques. Euclidean interpolation between two elements $\boldsymbol{D}_i$ and $\boldsymbol{D}_j$ of $\mathcal{P}(n)$ can be achieved as

$$\boldsymbol{D}_E(t) = t\boldsymbol{D}_i + (1-t)\boldsymbol{D}_j, \quad \forall t \in [0,1]\,. \tag{182}$$

This naturally yields the formulation of an exponential low-pass filter, where the filtered estimate $\boldsymbol{D}_{E_i}$ at iteration i is obtained from the filtered estimate $\boldsymbol{D}_{E_{i-1}}$ at iteration $i-1$ and the

[4] Especially in the context of floating-base mechanisms stabilization.

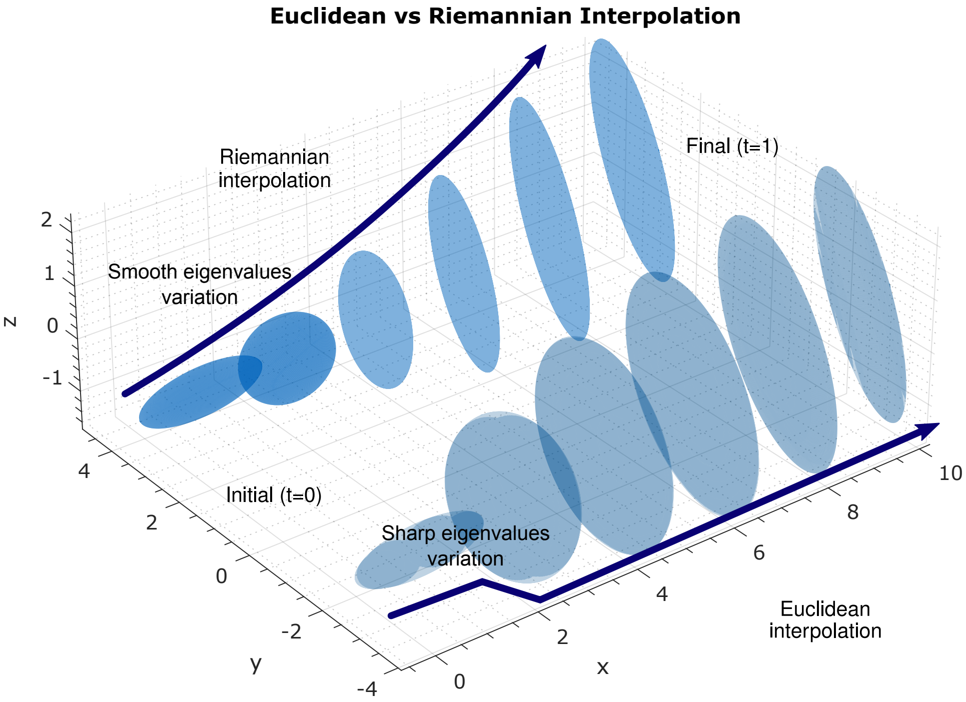


Figure 47 Difference between Euclidean and Riemannian ellipsoid interpolation. Notice the sharp volume changes occurring during the Euclidean interpolation process. This illustrates the interest of geodesic interpolation for physically consistent parameter identification purposes.

current unfiltered estimate $\boldsymbol{D}_i$ at iteration i as

$$\boldsymbol{D}_{E_i} = \alpha \boldsymbol{D}_{E_{i-1}} + (1-\alpha)\boldsymbol{D}_i, \quad \forall \alpha \in [0,1] \tag{183}$$

where the tuning factor α can be used to regulate how fast the set of identified inertial parameters is going to be updated (and therefore how fast the old parameter values will be forgotten). It is worth emphasizing that the resulting set of filtered inertial parameters will remain physically consistent. This is due to the convex nature of physicality cone. However, it appears that such process may also lead to improper variations of the eigenvalues of the interpolated pseudo-inertia matrices as highlighted in [341]. This is especially true in case the parameter estimate undergoes strong variations. Nevertheless in this context, it is still possible to leverage the Riemannian nature of the physicality manifold by executing interpolation along a geodesic curve of $\mathcal{P}(n)$ rather than along a Euclidean straight line. As stated in [348], Riemannian geodesic interpolation between two elements $\boldsymbol{D}_i$ and $\boldsymbol{D}_j$ of $\mathcal{P}(n)$ can be achieved following

$$\boldsymbol{D}_R(t) = \boldsymbol{D}_i \left(\boldsymbol{D}_i^{-1}\boldsymbol{D}_j\right)^t, \quad \forall t \in \mathbb{R}. \tag{184}$$

Note that this equation is simply a reformulation of the geodesic curve equation (164). In practice, Riemannian geodesic interpolation enables smoother variation of the eigenvalues of the pseudo-inertia matrix for control applications. This is for instance illustrated in Fig. 47, where intermediate steps of Euclidean and Riemannian interpolation between two 3D-ellipsoid configurations – depicting for instance the inertia tensor of a robot link – are shown. In this figure, it is possible to see that unlike the Riemannian interpolation the Euclidean interpolation process actually leads to a substantial increase of the ellipsoid volume between the initial and final configurations which might in turn have adversarial effects in the context of stabilization and control. As previously, a Riemannian geodesic exponential low-pass filter formulation can be derived from the interpolation equation (184) following

$$\boldsymbol{D}_{R_i} = \boldsymbol{D}_{R_{i-1}} \left(\boldsymbol{D}_{R_{i-1}}^{-1} \boldsymbol{D}_i \right)^{1-\alpha}, \quad \forall \alpha \in \mathbb{R} \tag{185}$$

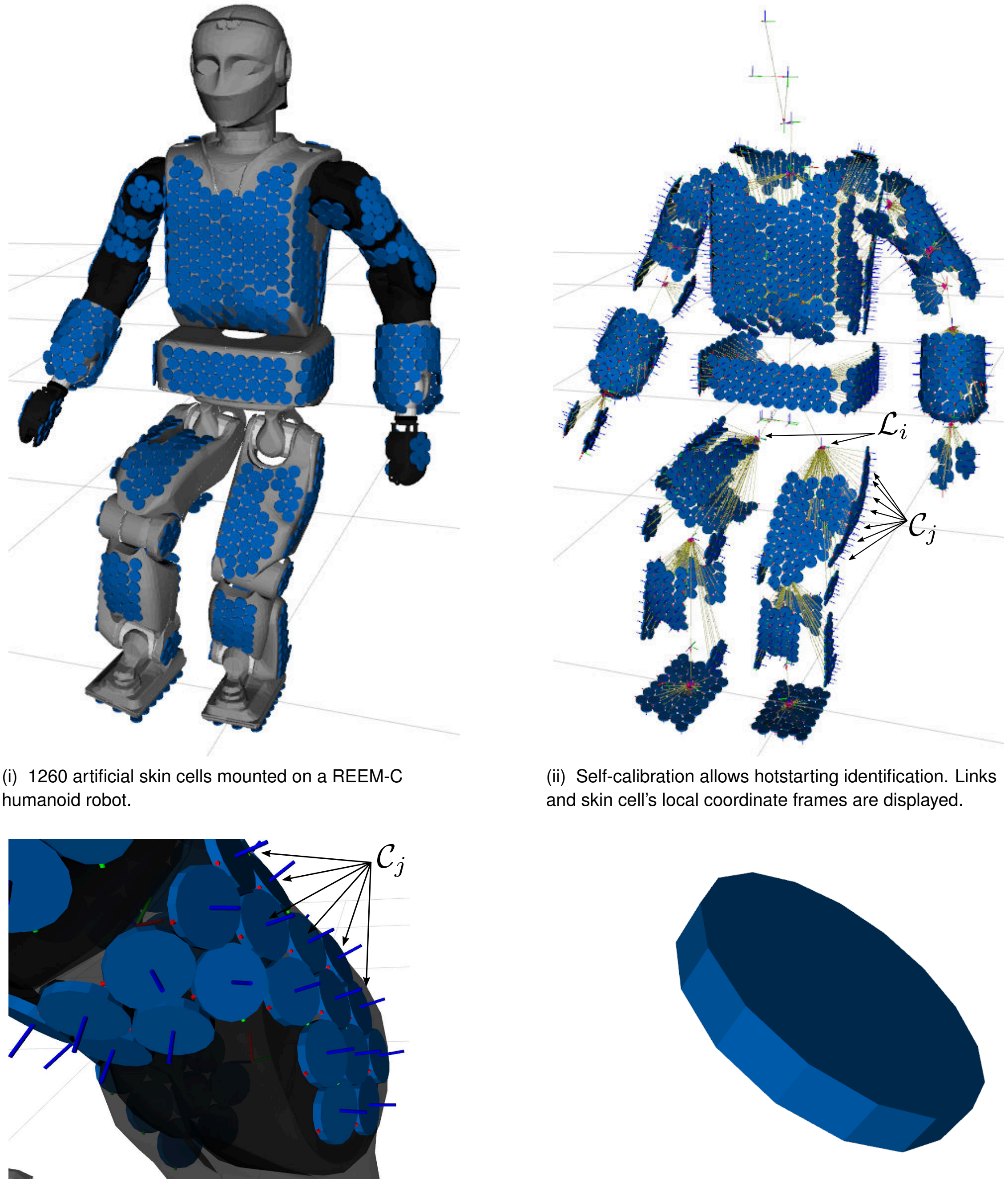
6.3.2. Leveraging a Self-Calibrating Multimodal Artificial Robot Skin for Identification Purposes

Manipulation tasks that are not limited to the end-effectors typically require the use of distributed sensors, covering part and even the entire body of a robot, thereby effectively forming an artificial skin. Although covering an entire humanoid robot with an artificial skin has multiple advantages, it unfortunately tends to significantly alter its dynamic properties as the number of cells increases. Nevertheless, in case the skin spatial distribution is known, for example through suitable self-calibration algorithm [6], the inertial properties of the coupled system can be estimated and used as an improved prior estimate of the system dynamics.

6.3.2.1 Hot-starting an Identification Algorithm using a Simplified Skin Model

Provided that a robot covered with an artificial skin is aware of the spatial arrangement and dynamic properties of this skin, it is possible to significantly refine its dynamic model before proceeding with more conventional identification. It is worth emphasizing that this new model can be used as an initial estimate or a new physically consistent relaxation value in the context of an inertial parameter identification process. As a matter of fact, the dynamic properties of each artificial skin cell can be fused with those of the robot link to which they are attached. We here consider as a first approximation that the assembly consisting of each cell's printed circuit board, its various electronic components, the elastomer layers and the connectors behaves locally as a homogeneous cylinder, whose position and orientation correspond to those of the cell, and are hence available after suitable calibration. Each cylinder has a height h, a radius r, a mass M_{cell} and an inertia $\boldsymbol{I}_{cell} = \begin{bmatrix} \frac{1}{12}M_{cell}(3r^2+h^2) & 0 & 0 \\ 0 & \frac{1}{12}M_{cell}(3r^2+h^2) & 0 \\ 0 & 0 & \frac{1}{2}M_{cell}r^2 \end{bmatrix}$ relative to its CoM[5] (c.f. Fig. 48iv). The cell's CoM is located in $\boldsymbol{p}_{cell}^{cm} = [X_{cell}, Y_{cell}, Z_{cell}]^\top$ relative to the cell's local reference frame $\mathcal{C}$. We here consider that the coordinate frame $\mathcal{C}_j$ attached to each skin cell j is related to the corresponding link i reference frame, denoted as $\mathcal{L}_i$, by a – known – homogeneous transform $\boldsymbol{H}_i^j = \begin{bmatrix} \boldsymbol{R}_i^j & \boldsymbol{p}_i^j \\ \boldsymbol{0} & 1 \end{bmatrix}$, determined after skin self-exploration and self-calibration procedure (c.f. Fig.48). As a result, the link inertial properties can be updated using a set of geometrical transformations. The mass M_i of a robot link i covered with $N_{cell_i} \in \mathbb{N}$ artificial skin cells can be trivially determined by summation $M_i =$

[5] This formula is valid if the considered reference CoM frame is aligned to the main axes of inertia of the cylinder, which in our case is verified, as the cell-attached reference frames $\mathcal{C}$ are aligned with the cylinder.



(i) 1260 artificial skin cells mounted on a REEM-C humanoid robot.

(ii) Self-calibration allows hotstarting identification. Links and skin cell's local coordinate frames are displayed.

(iii) The z axis of each cell-attached frame $\mathcal{C}_j$ is visible in dark blue.

(iv) A simplified cylinder dynamic model was selected for the computations.

Figure 48 Accounting for the skin dynamics to hotstart identification

$M_i + \sum_{j=1}^{N_{cell_i}} M_{cell_j}$. Similarly, the new location of the link CoM can be determined as

$$\boldsymbol{p}_i^{cm} = [X_i \; Y_i \; Z_i]^\top = \frac{M_i \boldsymbol{p}_i^{cm} + \sum_{j=1}^{N_{cell_i}} M_{cell_j} \left[\boldsymbol{p}_i^j + (\boldsymbol{R}_i^j)^\top \boldsymbol{p}_{cell_j}^{cm} \right]}{M_i + \sum_{j=1}^{N_{cell_i}} M_{cell_j}} \tag{186}$$

Update of the link inertia matrix actually requires a more careful treatment. Using the Huygens-Steiner parallel axis theorem (c.f. Appendix A.1.3), it is possible to express the inertia $\boldsymbol{I}_{cell}^j$

of every cell j into the corresponding cell's reference frame $\mathcal{C}_j$ rather than in the cell's CoM frame, as

$$\boldsymbol{L}_{cell}^{j} = \boldsymbol{I}_{cell}^{j} + M_{cell} \left[\boldsymbol{p}_{cell}^{cm}\right]_{\times}^{\top} \left[\boldsymbol{p}_{cell}^{cm}\right]_{\times} \tag{187}$$

$\boldsymbol{L}_{cell}^{j}$ can then be expressed relative to the corresponding link reference frame $\mathcal{L}_i$ as

$$\boldsymbol{L}_{i}^{j} = \boldsymbol{R}_{i}^{j} \boldsymbol{I}_{cell}^{j} \left(\boldsymbol{R}_{i}^{j}\right)^{\top} + M_{cell} \left[\boldsymbol{p}_{i}^{j}\right]_{\times}^{\top} \left[\boldsymbol{p}_{i}^{j}\right]_{\times} \tag{188}$$

The total link inertia update is eventually obtained by adding the different cells inertia to the link inertia, yielding

$$\boldsymbol{L}_{i} = \boldsymbol{L}_{i} + \sum_{j=1}^{N_{cell_i}} \boldsymbol{L}_{i}^{j} \tag{189}$$

These updated parameters are physically consistent and can be used as a relaxation to a classic identification problem or as a starting point for identification algorithms requiring an initial guess of the inertial parameter values.

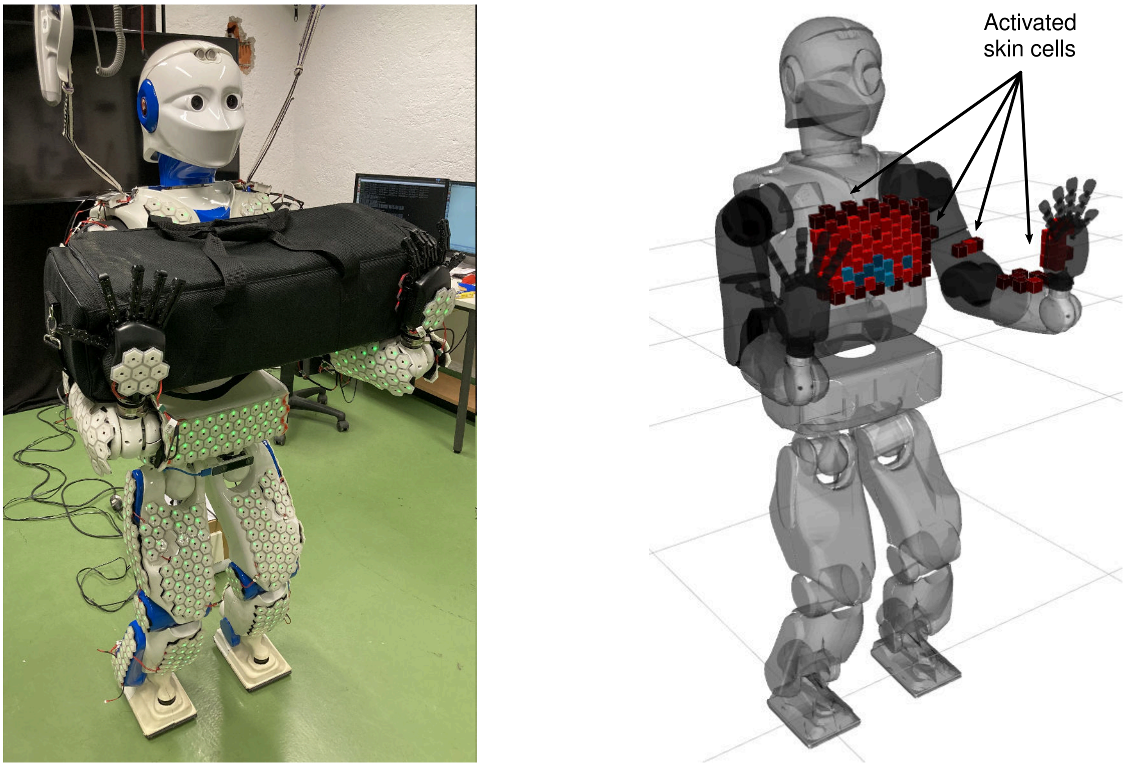
6.3.2.2 Inertial Parameter Identification in the Context of Full-Body Manipulation

One of the main obstacles to the real-world deployment of adaptive control strategies on humanoid robots – especially for stabilization and whole-body manipulation purposes – is the need for appropriate excitation data [340]. To identify the dynamics of its body segments, a robot must execute a set of exploratory movements[6] that reveal the contribution of each link in terms of the net contact wrench measured using its ankle force-torque sensors. For floating base mechanisms this usually proves to be a challenging and time-consuming process, which turns our to be a research field in its own right. The reader is referred to chapter 2 of this thesis for a more in-depth analysis of this topic.

Although the generation of permanent excitation is usually not a critical problem in the context of offline parameter identification, it raises substantial implementation and operational reliability concerns in the context of adaptive control, since in this case the identification must be executed under strict time constraints. In essence, it is undesirable for a robot handling an unknown payload to execute a full sequence of exploratory motions as soon as the dynamic properties of that payload change. In this context, the use of tactile feedback as an additional source of information, appears to be judicious insofar as it allows assessing the *occurrence*, *location* and *nature* of the dynamic coupling between a robot and its payload. As a matter of fact, the multi-modality of tactile data can be leveraged to distinguish, for instance, between proximity and effective contact of a robot and its payload or to evaluate the stability of this contact by detecting slippage thanks to the analysis of the vibratory modes of the accelerometer data or by monitoring the evolution of the number of cells involved in the interaction process [7]. In practice, a reliable knowledge of the coupling modalities between a robot and its payload opens the door to substantial simplifications of the identification problem insofar as it provides

[6] As pointed out by Venture et al, the nature of these movements depends on the type of parameters one wishes to identify.

[7] This specific aspect being out of the scope of this thesis, it is here considered as future works.

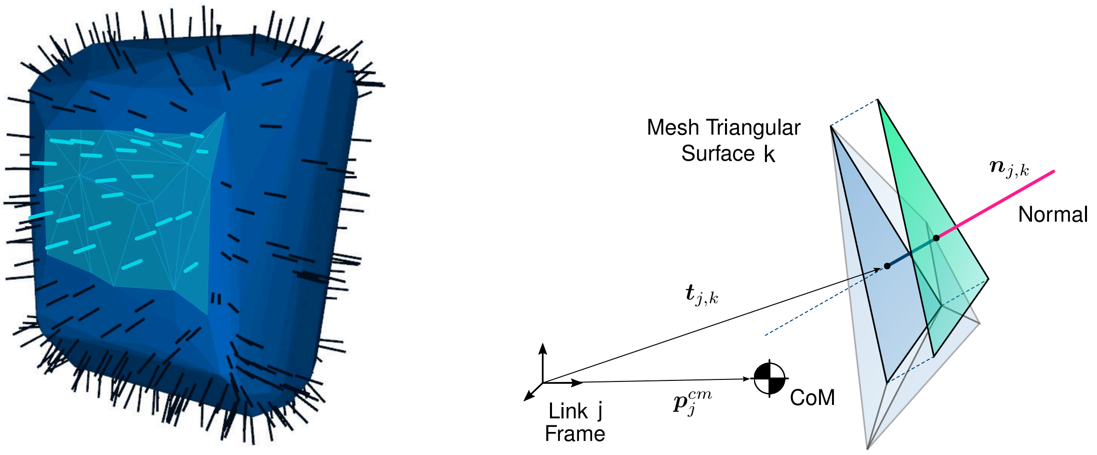


(i) The REEM-C humanoid robot manipulating a 9kg payload using its arms and chest.

(ii) The payload coupling can be assessed using the artificial skin covering the robot.

Figure 49 Whole-body manipulation scenario with artificial skin feedback.

indications as to the links whose inertial parameters are likely to have been modified during manipulation. It should be emphasized that in an adaptive stabilization context, considering a payload properly coupled to a robot as being an integral part of that robot allows to exploit its dynamic properties to improve performance. The first step is therefore to determine the links of the robot whose parameters may have been modified by stable contact with the payload. Selective excitation approaches such as that proposed by Venture et al. [123, 124] or Jovic et al. [20] based on sub-regressor matrices, may then be exploited to re-identify the considered bodies, although it should be noted that the control task executed by the robot can also prove, in itself, to be expressive enough to allow suitable data collection. Online adjustment of the filter tuning factor α in equation (185) can then be carried out based on the skin feedback in order to rapidly learn the coupled link-payload parameters. Finally, it should be emphasized that the precise knowledge of the coupling between the robot and its payload, and in particular the location of the different contact points, can be used to relax the CoM bound constraint acting on the link supporting interaction as exposed in Fig. 49. In practice, a tailor-made offset $\boldsymbol{t}_{j,k}^{off}$ can be added to the constraint, along the normal of the considered surface k as



(i) Location of the relaxed constraints.

(ii) Constraint relaxation.

Figure 50 Convex hull constraint relaxation triggered by the skin in the whole-body manipulation scenario depicted in Fig. 49

exposed in Fig. 50ii. This offset will be integrated within the constraint eq. (180) as

$$\begin{aligned} \boldsymbol{B} &= \begin{bmatrix} \boldsymbol{B}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{B}_n \end{bmatrix} \in \mathbb{R}^{N_C \times p} | \, \forall j = \{1 \cdots n\}, \; \boldsymbol{B}_j = \underbrace{\begin{bmatrix} 0 & \boldsymbol{n}_{j,1}^\top & \mathbf{0}_{1\times 6} \\ \vdots & \vdots & \vdots \\ 0 & \boldsymbol{n}^\top_{j,N_T^j} & \mathbf{0}_{1\times 6} \end{bmatrix}}_{\boldsymbol{\chi}_j} \in \mathbb{R}^{N_T^j \times 10} \\ \boldsymbol{b} &= \begin{bmatrix} \boldsymbol{b}_1 \\ \vdots \\ \boldsymbol{b}_n \end{bmatrix} \in \mathbb{R}^{N_C} | \, \forall j = \{1 \cdots n\}, \; b_j = M_j \begin{bmatrix} \boldsymbol{n}_{j,1}^\top \left(\boldsymbol{t}_{j,1} + \boldsymbol{t}_{j,1}^{off} \right) \\ \vdots \\ \boldsymbol{n}^\top_{j,N_T^j} \left(\boldsymbol{t}_{j,N_T^j} + \boldsymbol{t}^{off}_{j,N_T^j} \right) \end{bmatrix} \in \mathbb{R}^{N_T^j} \end{aligned} \tag{190}$$

It should be noted that in case a CAD model of the robot is not provided by its manufacturer, the artificial skin can still be used to constrain the identification process, as the position and orientation of each skin cell can be known in the considered link frame after a short calibration step [8]. In this context, the normal vector $\boldsymbol{n}_{j,k} \in \mathbb{R}^3$ of constraint surface can simply be the set to match the normal vector of each skin cell, or possibly each group of skin cell for improved efficiency.

6.3.3. Online Physically-Consistent Identification in Batch

Real-time inertial parameters identification has the potential to enable a certain degree of adaptability in the stabilization strategy of a humanoid robot. This proves to be especially relevant in a full-body manipulation scenario, where a robot is manipulating an unknown payload. To enable online identification at a rate f_{id}, the proposed algorithm is executed in batches. Note that the identification rate may in practice be one to two orders of magnitude lower than the control loop frequency, denoted as f_c. At every control iteration, a set of updated joint angles $\boldsymbol{q}$, contact wrenches $\boldsymbol{w}_l, \boldsymbol{w}_r$ and IMU measurements $\dot{\boldsymbol{v}}, \boldsymbol{\omega}$, are appended to form a set of dedicated fixed-size buffers – also referred to as mini-batches – each containing N_{mb} inputs

[8] The detail of this process being out of the scope of this thesis, the reader is referred to [6] and the references therein for additional information.

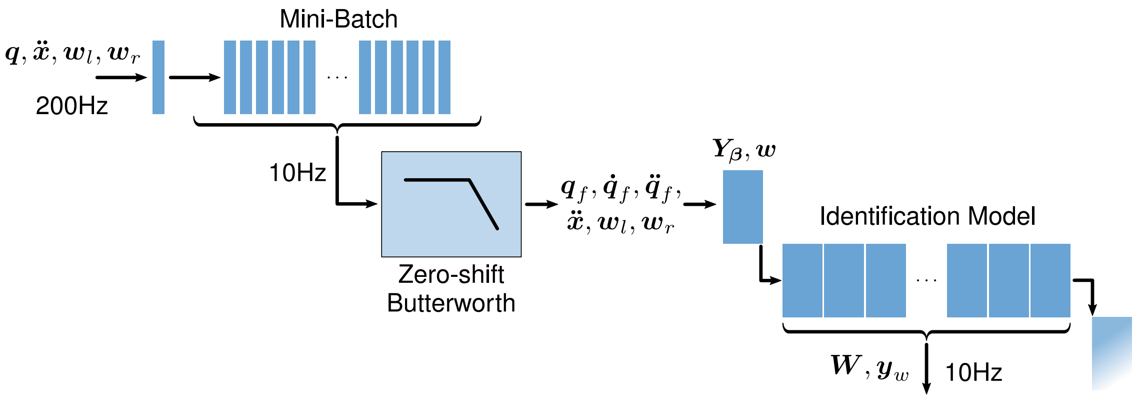

Mini-Batch
$\boldsymbol{q}, \ddot{\boldsymbol{x}}, \boldsymbol{w}_l, \boldsymbol{w}_r$
200Hz
...
10Hz
Zero-shift
Butterworth
$\boldsymbol{q}_f, \dot{\boldsymbol{q}}_f, \ddot{\boldsymbol{q}}_f,$
$\ddot{\boldsymbol{x}}, \boldsymbol{w}_l, \boldsymbol{w}_r$
$\boldsymbol{Y}_{\boldsymbol{\beta}}, \boldsymbol{w}$
Identification Model
...
$\boldsymbol{W}, \boldsymbol{y}_w$
10Hz

Figure 51 Details of the batch identification process.

(c.f. Fig. 51). The mini-batch containing the joint position values undergoes zero-shift band-pass filtering using a dedicated third-order forward-backward Butterworth filter. The resulting filtered values are then differentiated twice, yielding the joint velocity and joint acceleration mini-batches. It should be emphasized that since the forward-backward filter makes use of multiple filtering passes in order to cancel the phase shift induced by classic online digital filters [349], it cannot be used on a continuous data stream but rather on a batch of data, hence the proposed formulation. In this context, it is worth noting that the filtering artifacts arising at the ends of the measurement buffer can be easily discarded by considering only the central part of the buffer in the computations. The size of the buffers mentioned in this thesis naturally takes into account this elimination process. Unlike BIRDy, data decimation is here not considered. This is justified, by the rather low frequency of the position control loop – typically f_c =200Hz – compared to a torque control loop usually running at rates of several kHz, and where decimation filtering can be used to alleviate the effects of undesired ripples in the motor torque signal. After the filtering and numerical time derivation steps, the mini-batch data is used to build an updated robot observation system as a function of the base parameter vector. Each element of this system can be computed as

$$\begin{bmatrix} \mathbf{0}_{6\times 1} \\ \boldsymbol{\tau} \end{bmatrix} = \underbrace{\begin{bmatrix} \boldsymbol{Y}_{b\boldsymbol{\beta}} \\ \boldsymbol{Y}_{q\boldsymbol{\beta}} \end{bmatrix}}_{\boldsymbol{Y}_{\boldsymbol{\beta}}} \boldsymbol{\beta} + \sum_{i=1}^{c} \begin{bmatrix} \boldsymbol{J}_{b_i}^{\top} \\ \boldsymbol{J}_{q_i}^{\top} \end{bmatrix} \boldsymbol{w}_i \, . \tag{191}$$

In this equation, $c \in$ denotes the number of contact points between the robot and its environment, $\boldsymbol{\tau} \in \mathbb{R}^n$ is the vector of control torques, applied by the different robot joints, $\boldsymbol{w}_i$ is the i^{th} contact wrench between the robot and its environment and the matrices $\boldsymbol{J}_{b_i} \in \mathbb{R}^{6\times 6}$ $\boldsymbol{J}_{q_i} \in \mathbb{R}^{6\times n}$ are the Jacobians mapping the contact wrenches to the floating base and to the different robot links. Similar to Venture et al. [123] only the regressor rows related to the floating base of the robot are considered as the reconstruction of joint torque signal is irrelevant in stabilization of a position controlled mechanism. The observation system $(\boldsymbol{W}, \boldsymbol{y}_w)$ is therefore constructed with the blocks $(\boldsymbol{Y_{b\beta}}, \boldsymbol{J}_{b_i}^{\top}\boldsymbol{w}_i)$ of the previous equation (c.f. Fig. 51)

$$\mathbf{0}_{6\times 1} = \boldsymbol{Y_{b\beta}}\boldsymbol{\beta} + \sum_{i=1}^{c} \boldsymbol{J}_{b_i}^{\top}\boldsymbol{w}_i\,. \tag{192}$$

The main identification batch contains a set of N mini-batches and therefore $N_{batch} = N{\cdot}N_{mb}$ distinct observation systems, corresponding to as many control epochs. It therefore entirely renewed every $(N_{batch} + 2N \cdot N_{border})/f_c$ seconds when executed on a robot with a control loop frequency f_c. The scalar term N_{border} of the previous expression refers to the border elements of each mini-batch, that have to be removed to discard possible filtering artifacts. The identification algorithm is finally executed on the full data batch $(\boldsymbol{W}, \boldsymbol{y}_w)$. Modern humanoid robots having multiple embedded computers, it is possible – and even recommended – to execute the identification process and the robot controller on separated processors, to avoid inducing potentially harmful time jitters within the low-level real-time control and stabilization loops. In practice, although the control and identification processes operate at different frequencies, they are connected to each-other through suitable non-blocking data-queues. The updated parameters are low-pass filtered using the process described in section 6.3.1.4 of this chapter, before being sent to the control thread at the control frequency f_c, such that each new control iteration is executed with a set of smoothly updated parameters.

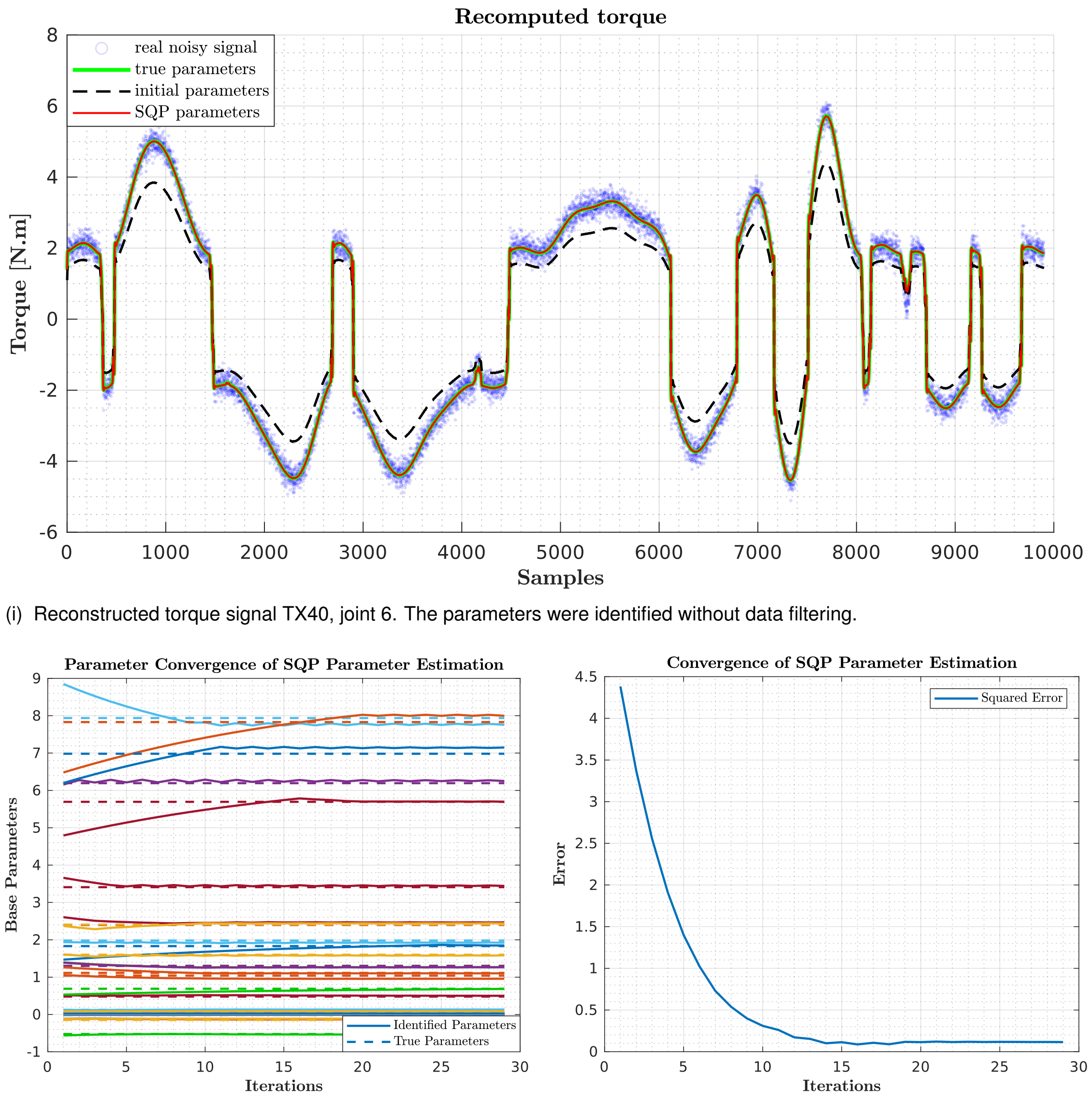
6.4. Validation of PC-IDIM-SQP in Simulation

This section investigates the convergence and noise robustness properties of the IDIM-SQP identification algorithm through a set of Monte-Carlo Simulation experiments (MCS) executed within the BIRDy identification benchmark presented in chapter 5.

6.4.1. Monte-Carlo Simulation (MCS) using the BIRDy Identification Framework

During these MCS experiments, the performance and robustness properties of *three* variants of the proposed algorithm were evaluated, namely the PC-IDIM-SQP-OLS, PC-IDIM-SQP-WLS and PC-IDIM-SQP-IRLS, and compared to a selected set of methods already implemented in BIRDy. Each algorithm was tested with and without data filtering on a TX40 industrial robot model, under the very same experimental conditions as those presented in chapter 5. More specifically, the experiment data of MCS-TX40-4-1, MCS-TX40-4-2, MCS-TX40-4-3, MCS-TX40-4-4, and MCS-TX40-4-5 were used (c.f. Table 1). It should be emphasized that the geometric relaxation was left inactivated during these experiments to better highlight the performance of each algorithm. This is also justified by the fact that the considered identification methods are tested on permanently exciting trajectories. The main tuning factor of the proposed algorithm is the gain ϖ in eq. (176) and eq. (178). The best results were obtained by annealing ϖ as the algorithm proceeds, following an exponentially decreasing law of the form

$$\varpi_i = \lambda e^{-\mu i}, \quad \forall \lambda, \mu \in \mathbb{R}_+ \tag{193}$$



(i) Reconstructed torque signal TX40, joint 6. The parameters were identified without data filtering.

(ii) IDIM-SQP-WLS base parameter convergence.

(iii) IDIM-SQP-WLS error function.

Figure 52 Convergence results of IDIM-SQP-WLS without data filtering.

where i refers to the iteration number of the algorithm. As for the results obtained in chapter 5, the computations were realized using Matlab R2020a on a AMD-Threadripper 1920X workstation with 32GB of RAM. The qpOASES solver [216] was used as it offers a mature Matlab interface with support of double-sided constraints. The reader is referred to chapter 5 for additional details on the BIRDy identification framework and on the figures of merit (FoM) selected to compare the performance of the different identification algorithms.

6.4.2. Monte Carlo Simulation Results and Discussion

After suitable tuning of the annealing factor ϖ, convergence of the different variations of PC-IDIM-SQP is observed for every sample of the different MCS sets. It takes PC-IDIM-SQP an average of 15 iterations to achieve convergence under the standard joint noise conditions of experiment MCS-TX40-4-2 (c.f. Fig. 52ii and Fig. 52iii). The obtained set of parameters allow

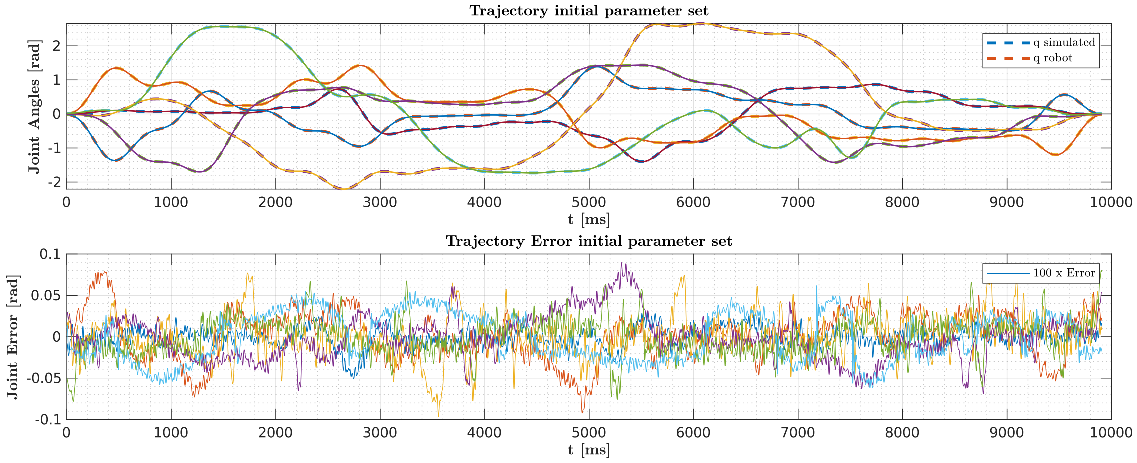
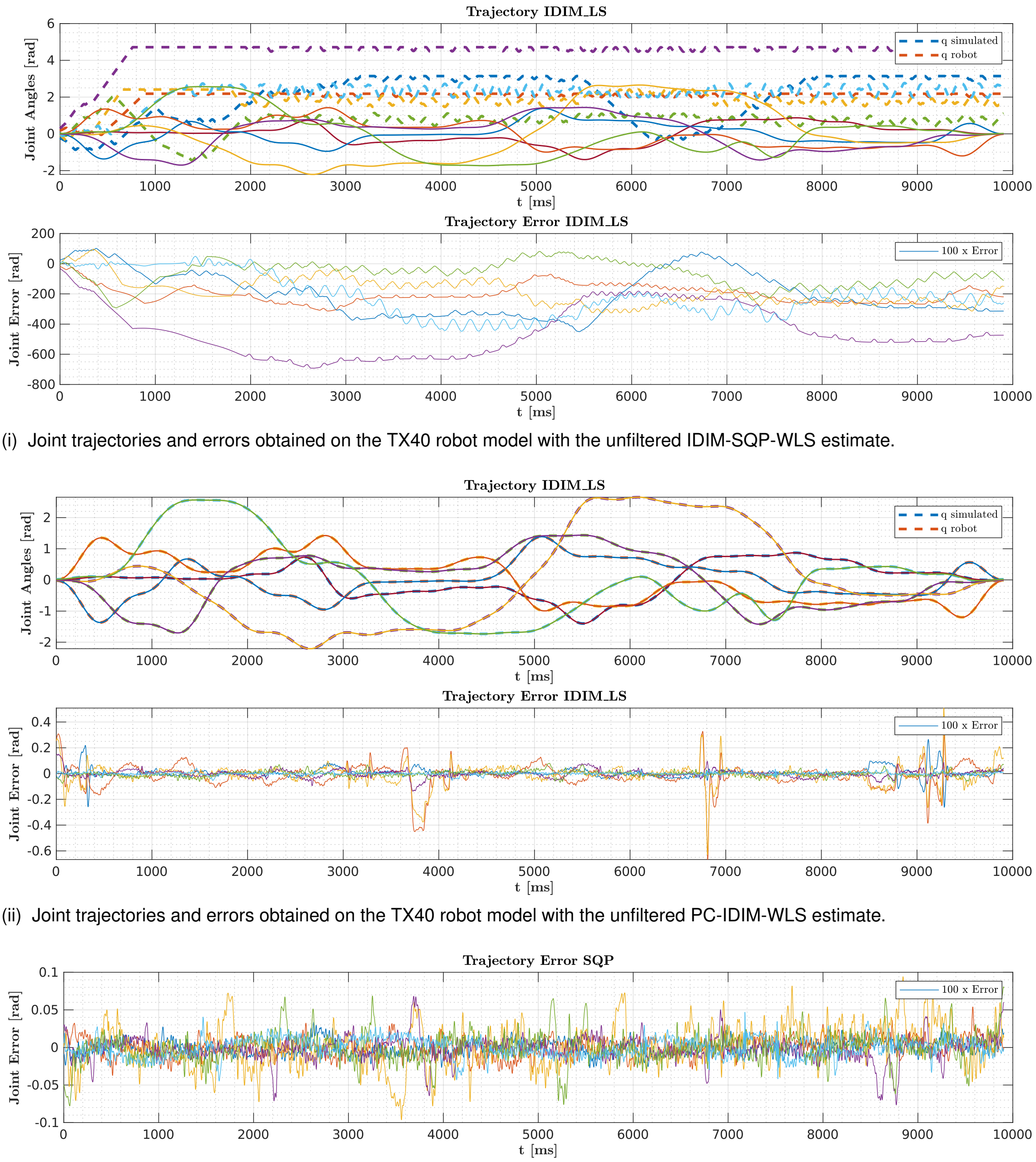


Figure 53 Joint trajectories obtained on the TX40 robot model with the unfiltered PC-IDIM-WLS estimate.

accurate reconstruction of the joint torque signal, as exposed in Fig. 52i. As the computation time is highly dependent on the batch size, the latter will therefore have to be carefully adjusted for online execution purposes. A quick physicality check using a Cholesky decomposition of the pseudo inertia matrix generated with the identified parameter sets obtained during the different MCS scenarios confirms that – in contrast to the case of unconstrained methods such as IDIM-LS – the identified parameter sets are physically consistent for every single experiment execution within the MCS. It is worth pointing that physical consistency is also obtained without data filtering, and under high joint position and torque noise conditions, such as those encountered in the MCS-TX40-4-5 experiments of chapter 5 (c.f. table 1). These results are valid for IDIM-SQP-OLS, -WLS and -IRLS and moreover comparable to those obtained with methods based on SDP with LMI constraints. The results in table 7 also suggest that the different IDIM-SQP methods have excellent noise robustness properties, both at the joint position level and at the joint torque level. In particular, the results obtained with the MCS-TX40-4-5 scenario – at the highest noise levels – clearly demonstrate that IDIM-SQP estimates, even without filtering, allows an improvement of the control performance compared to the initial parameter estimate (c.f. Fig. 54iii) while the unconstrained least squares methods such as IDIM-OLS, -WLS or -IRLS result – if unfiltered – in a strong trajectory divergence (c.f. Fig. 54). It is worth noting that the trajectory tracking properties of IDIM-SQP also appear to be improved compared to SDP identification methods (c.f. Fig. 54ii).



(i) Joint trajectories and errors obtained on the TX40 robot model with the unfiltered IDIM-SQP-WLS estimate.

(ii) Joint trajectories and errors obtained on the TX40 robot model with the unfiltered PC-IDIM-WLS estimate.

(iii) Joint trajectory errors obtained on the TX40 robot model with the unfiltered IDIM-SQP-WLS estimate.

Figure 54 Joint trajectories obtained on the TX40 robot model with the initial parameters, the unfiltered IDIM-WLS estimate, PC-IDIM-WLS and IDIM-SQP-WLS.

MCS-TX40-4-5	$E(d_{\boldsymbol{q}}), \sigma_{\boldsymbol{q}}$	$E(d_{\boldsymbol{\tau}}), \sigma_{\boldsymbol{\tau}}$	$E(d_t), \sigma_t$	$E(d_{N_{it}}), \sigma_{N_{it}}$	$E(d_{N_{sim}}), \sigma_{N_{sim}}$
IDIM-OLS	442% (969%)	99.82% (17.05%)	0.54s (5.30%)	1.00 (0.00%)	N.A.
IDIM-OLS filt.	420% (955%)	99.98% (18.18%)	0.56s (5.09%)	1.00 (0.00%)	N.A.
IDIM-WLS	241% (647%)	99.82% (16.99%)	0.58s (4.34%)	1.00 (0.00%)	N.A.
IDIM-WLS filt.	0.033% (0.14%)	99.98% (18.18%)	0.58s (4.40%)	1.00 (0.00%)	N.A.
IDIM-TLS	579% (1356%)	NaN% (NaN%)	0.55s (5.19%)	1.00 (0.00%)	N.A.
IDIM-TLS filt.	579% (1356%)	NaN% (NaN%)	0.56s (6.15%)	1.00 (0.00%)	N.A.
IRLS	229% (615%)	99.82% (16.99%)	0.95s (6.24%)	1.00 (0.00%)	N.A.
IRLS filt.	0.033% (0.14%)	99.98% (18.18%)	0.97s (6.61%)	1.00 (0.00%)	N.A.
ML	0.043% (0.17%)	100.10% (18.86%)	11.80s (46.24%)	1.00 (0.00%)	N.A.
ML filt.	0.035% (0.16%)	100.09% (18.79%)	8.01s (38.59%)	1.00 (0.00%)	N.A.
IDIM-IV	334% (918%)	100.05% (17.81%)	5.97s (17.02%)	9.53 (16.58%)	9.53 (16.58%)
DIDIM	0.161% (0.30%)	99.27% (9.37%)	1.76s (21.69%)	3.54 (16.82%)	3.54 (16.82%)
CLIE	0.038% (0.17%)	100% (18.73%)	35.26s (43.55%)	3.17 (57.12%)	214 (46.40%)
CLOE	0.037% (0.17%)	100% (18.62%)	21.92s (62.45%)	5.13 (69.78%)	309 (63.44%)
EKF	0.289% (0.90%)	103% (29.82%)	2.48s (6.01%)	1.00 (0.00%)	N.A.
SREKF	0.289% (0.90%)	103% (29.82%)	4.92s (3.90%)	1.00 (0.00%)	N.A.
UKF	0.290% (0.92%)	103% (30.02%)	12.24s (2.92%)	1.00 (0.00%)	N.A.
SRUKF	0.294% (0.93%)	103% (30.33%)	5.75s (4.51%)	1.00 (0.00%)	N.A.
CDKF	0.288% (0.91%)	103% (29.89%)	10.02s (4.52%)	1.00 (0.00%)	N.A.
SRCDKF	0.290% (0.91%)	103% (30.01%)	6.63s (3.95%)	1.00 (0.00%)	N.A.
ANN	448% (1158%)	99.83% (17.20%)	0.92s (4.97%)	1.00 (0.00%)	N.A.
ANN filt.	459% (1084%)	100% (18.46%)	0.93s (5.21%)	1.00 (0.00%)	N.A.
HTRNN	298% (661%)	133.44% (73.31%)	0.55s (3.50%)	1.00 (0.00%)	N.A.
HTRNN filt.	424% (929%)	99.99% (18.23%)	0.55s (3.08%)	1.00 (0.00%)	N.A.
PC-IDIM-OLS	0.064% (0.22%)	99.83% (17.20%)	2.05s (4.28%)	1.00 (0.00%)	N.A.
PC-IDIM-OLS filt.	0.036% (0.14%)	100.00% (18.25%)	2.06s (3.71%)	1.00 (0.00%)	N.A.
PC-IDIM-WLS	0.060% (0.22%)	99.82% (17.15%)	2.07s (4.35%)	1.00 (0.00%)	N.A.
PC-IDIM-WLS filt.	0.033% (0.14%)	99.99% (18.22%)	2.15s (5.76%)	1.00 (0.00%)	N.A.
PC-IDIM-IRLS	0.060% (0.22%)	99.83% (17.18%)	7.75s (10.48%)	1.00 (0.00%)	N.A.
PC-IDIM-IRLS filt.	0.033% (0.14%)	99.99% (18.22%)	7.25s (12.75%)	1.00 (0.00%)	N.A.
PC-IDIM-IV	0.064% (0.24%)	100% (19.09%)	2.34s (3.98%)	3.00 (0.00%)	3.00 (0.00%)
PC-DIDIM	0.033% (0.15%)	100% (18.44%)	3.71s (12.85%)	3.00 (0.00%)	3.00 (0.00%)
PC-SQP-OLS	0.045% (0.18%)	99.79% (17.01%)	1.00s (5.69%)	30.00 (0.00%)	N.A.
PC-SQP-OLS filt.	0.032% (0.14%)	99.99% (18.24%)	1.03s (6.43%)	30.00 (0.00%)	N.A.
PC-SQP-WLS	0.042% (0.18%)	99.87% (17.52%)	1.10s (4.38%)	30.00 (0.00%)	N.A.
PC-SQP-WLS filt.	0.033% (0.15%)	100.01% (18.36%)	1.13s (4.84%)	30.00 (0.00%)	N.A.
PC-SQP-IRLS	0.044% (0.18%)	99.84% (17.34%)	2.13s (2.89%)	30.00 (0.00%)	N.A.
PC-SQP-IRLS filt.	0.032% (0.14%)	99.99% (18.24%)	3.23s (5.25%)	50.00 (0.00%)	N.A.

Table 7 Various FoM computed for the TX40 robot model, with a control frequency of $5kHz$, a sampling frequency of $1kHz$ and a decimation frequency of $100Hz$. These results are obtained using the average values $E(d)$ and standard deviations σ of the 1300 parameter estimates obtained during the MCS over a set of 10s experiments, with $\boldsymbol{\sigma_\tau} = 1 \cdot 10^{-1} N.m$ and $\boldsymbol{\sigma_q} = 1 \cdot 10^{-2} rad$. Data is filtered using a Butterworth forward-backward filter with a cutoff frequency of $50Hz$.

6.5. Identification Experiments on a Humanoid Robot

This section discusses the detail of the proposed implementation on a humanoid robot. The IDIM-SQP algorithm is implemented on a real humanoid robot, first for offline identification purposes, and then for online identification in the context of whole-body manipulation. The tactile feedback from an artificial skin covering the robot is exploited to simplify the identification problem.

6.5.1. Experimental Setup

The proposed experiment setup consists of a REEM-C humanoid robot, covered with 1260 cells of our multimodal artificial skin. The robot is equipped with two i7 Linux computers, each running a real-time patched *Ubuntu 16.04* Linux distribution. One of these computers, referred to as the *control computer* is entirely dedicated to time-sensitive control tasks while the other one, referred to as the *media computer*, can be used for advanced data processing tasks[9]. The control and media computers are allocated respectively 8Gb and 16Gb of RAM. It should be emphasized that hyper-threading is disabled on the control computer in order to mitigate the inevitable control loop frequency jitter induced by the operating system.

The complete control and identification frameworks are coded in C++, making extensive use of the Eigen 3.4 linear algebra library and of the ROS kinetic robot middle-ware environment. The low-level control framework, including the robot stabilizer, and trajectory generator, are running on the control computer at a steady frequency f_c of 200Hz, providing joint position control references to the robot's hardware interface. The Identification loop is executed on the media computer and performs regular updates of the control parameters through dedicated ROS data queues. Note that although the identification process is executed asynchronously, the parameter interpolation routine described in section 6.3.1.4 is executed at the controller's frequency f_c, thereby providing a parameter update at each new control iteration. After benchmarking multiple QP solvers[10], it was decided to make use of the qpMAD C++ optimization library from Alexander Sherikov [350], based on the Goldfarb-Idnani dual active set algorithm [354]. The robot regression matrix is generated following the approach of [355, 356, 109], detailed in Appendix A.3. This approach was implemented as an add-on to the RBDL C++ library [322] for efficient online numerical computation. The robot position and velocity relative to the world frame are measured using an *Intel Realsense* T265 odometry camera [357]. This device acts by fusing the data from two fisheye cameras with that of an embedded IMU into an odometry vector containing the position and orientation of the camera frame relative to its starting point, fixed in world frame, at a frequency of 200 Hz. In this manner the IMU drift can be corrected. The Realsense T265 also provides estimates of the linear and angular velocities in the current camera frame, at the same rate of 200 Hz.

[9] In our case, the media computer is used for parameter identification as well as skin data processing.

[10] Benchmarking of the QP solvers was executed on qpOASES[221], qpMAD[350], Quadprog++, OSQP[351] and QLD[352]. The results of this benchmark are consistent with [353].

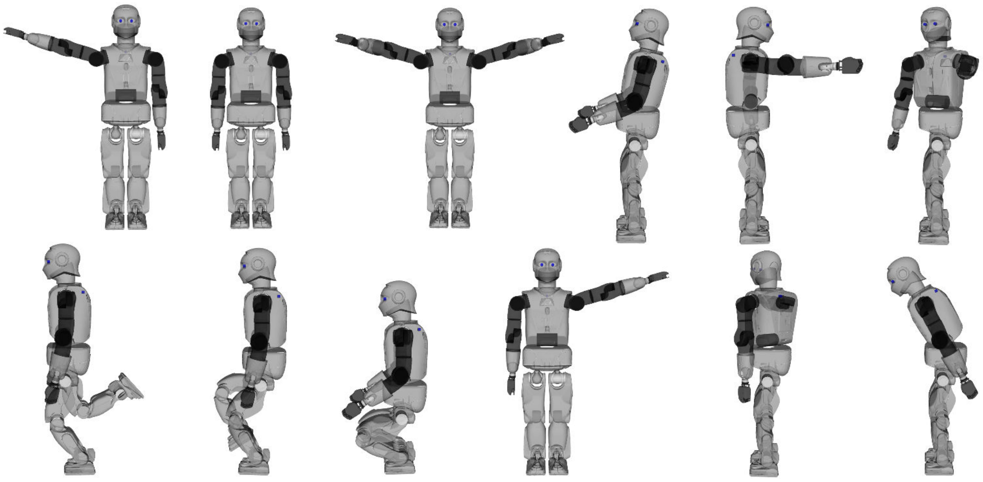


Figure 55 Sample of the different excitation postures executed by REEM-C for offline identification purposes.

6.5.2. Offline Identification of a Humanoid Robot using PC-IDIM-SQP

Besides validating the proposed identification approach on a floating base robot and comparing it to the more conventional IDIM-WLS, the objective of this first experiment is to generate a set of inertial parameter estimates that are sufficiently precise to be used as reference in for an online identification experiment in a whole-body manipulation scenario. During this first experiment, offline identification of the robot dynamics is carried out using a similar process as Venture et al. [123, 124]. Suitable excitation data is collected, similar to Jovic et al. [20], by letting the robot covered with artificial skin execute a set of walking and squatting motions, along with a set of tailor-made upper-body motion. The data is collected at a rate of 200Hz over a time horizon of 100s, eventually resulting in a unique data buffer containing 20000 epochs (batch border removal process included). It should be noted that the artificial skin layer covering the robot feet can here be used to reduce the impacts occurring during contact transition following the approach of Guadarrama et al. [358]. Such smooth contact transition actually prevents from exciting un-modeled flexibility modes of the robot, eventually resulting in higher quality estimates.

Structural rank deficiencies are addressed in a classical fashion, by generating and identifying a set of base parameters. The robot model consists in a total of 31 robot links connected to each-other through 1-DoF revolute joints. Each robot link j is allocated 10 inertial parameters, namely $\boldsymbol{\chi}_j = [M_j,\, MX_j,\, MY_j,\, MZ_j,\, XX_j,\, XY_j,\, XZ_j,\, YY_j,\, YZ_j,\, ZZ_j]^\top$. The absence of friction terms is here justified by the very nature of the floating base dynamics identification process. As a matter of fact, these parameters are structurally removed identification process. The robot therefore has a total of 310 standard inertial parameters, eventually reduced to 220 base parameters, namely 7 base parameters per link and 10 for the floating base. In IDIM-SQP-WLS, the physical consistency is enforced by construction and the Cholesky decomposition of the resulting link pseudo-inertia matrix might only fail in case its diagonal elements are too close to zero. This issue can be addressed by adding a small threshold $0 < \varepsilon < 1$ within the positivity constraint in eq. (176), yielding

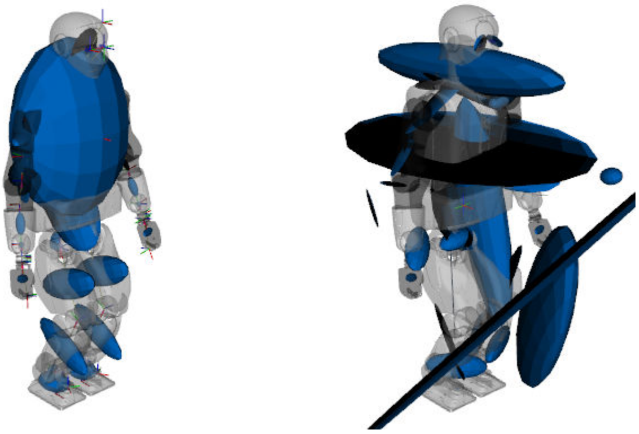
$$\boldsymbol{S\varrho} > \mathbf{0}_{4n} \Rightarrow \boldsymbol{S\varrho} > \varepsilon \cdot \mathbf{1}_{4n} > \mathbf{0}_{4n}. \tag{194}$$

During this work, the value ε was set to $\varepsilon = 10^{-5}$ to prevent infeasibility in the case of low-inertia or low-mass robot bodies. Riemannian geometric relaxation is finally used within the identification system to address the inevitable data rank deficiency issues arising from improper excitation. During this experiment, the relaxation gain λ of eq. (176) is set to 100 and the convergence gains λ and μ of eq. (193) are set to 0.25 and 10 respectively. The cross-validation experiment was executed on a new dedicated set of squats and upper-body robot motions. The IDIM-SQP-WLS algorithm was set to run for a maximum of 30 iterations.

6.5.3. Online Identification with Tactile Feedback

The objective of this second experiment is to demonstrate the capability of the proposed algorithm to be executed online, with in mind, a set of potential applications in the context of adaptive stabilization and control. During this experiment, the humanoid robot performs a whole-body manipulation task with a time-varying payload. The experiment setup is presented in Fig. 49. The robot is initially operated without payload and therefore makes use of the parameter set estimated during the offline identification experiment. The robot is then tasked to grasp and manipulate a travel bag containing a 8 kg payload. The contact with the payload occurs on the arms and torso: namely the robot holds the travel bag with the arm and fix it against its torso. The skin is here used to estimate the location and modalities of the robot-payload interaction. Consequently, a set of simplified excitation behaviors are locally generated over a short time horizon in order to enable online identification of the manipulated payload using the IDIM-SQP algorithm. During this experiment, the identification is triggered as soon as the robot detects a significant inconsistency between the expected and the measured ground reaction wrench. Considering that the manipulated payload is rigidly coupled to a specific robot link is of significant interest for both computational complexity and control reasons. On the one hand, this enables significant simplification of the identification problem, by only requiring re-identification of a single robot link[11] rather than the whole articulated system. On the other hand, the updated parameters of the system consisting of the robot link coupled to the payload can be seamlessly integrated within a model-predictive control strategy, where the robot can actively use the manipulated payload – now seen as part of its own body – for improved control performance. In practice, the way in which the payload dynamics is distributed at its different contact points with the robot body can prove challenging to capture, especially considering the wide variety of possible gripping modes achievable with a humanoid robot. As a matter of fact, not every robot-payload gripping mode allows such a simplification, not to mention that the field of grasping taxonomy is a research topic in its own right with multiple contributions over the last two decades [359]. For simplification purposes, this thesis will therefore only focus on the simple case where the payload can be

[11] In this context other links' inertial parameters can be forced to their initial values using a set of suitable equality constraints within eq. (176).



(i) Initial parameters estimate of the system “robot + skin”.

(ii) Unconstrained IDIM-WLS parameters estimate.

(iii) Constrained IDIM-SQP-WLS parameters estimate.

Figure 56 Visualization of the offline-estimated set of inertial parameters of the REEM-C humanoid robot.

considered as being dynamically coupled with *only one of the robot links*, namely the root of the kinematic sub-tree within which the gripping occurs. Transposed to hand manipulation, this amounts to only considering fully constrained manipulations involving the palm. In the case of the considered robot, the root of the kinematic sub-tree *arms + torso* is the torso. Therefore, the manipulated payload will be considered as part of the robot torso within the computations. The local excitation behavior is generated using the joints linking the interacting robot body to the floating base. Therefore during the considered experiment, the two torso joints underwent sinusoidal excitation over a short time horizon of 4s. Batch identification was then executed online on the resulting measurement buffer. As previously, the relaxation gain λ of eq. (176) was set to 100 and the number N_{border} of samples discarded on the higher and lower borders was set to 50. It should be emphasized that the relaxation process is now executed towards the set of physically consistent inertial parameters estimated during the previous offline identification experiment rather than towards CAD values.

6.5.4. Experiment Results and Discussion

The results of the offline identification experiments in terms of net contact wrench prediction are presented in Fig.57. The initial set of inertial parameters used for IDIM-SQP-WLS can be visualized in Fig.56i, while the sets of identified parameter for IDIM-WLS and IDIM-SQP-WLS can be visualized in Fig.56ii and Fig.56iii respectively. Note that in these figures, the parameters are represented as a set of ellipsoids, whose center is located in the estimated links' CoM and whose main axes are determined by the eigenvalues of the links' inertia matrix and mass. The cross-validation results in Fig.57 suggest that IDIM-SQP-WLS provides more accurate net contact wrench predictions than IDIM-WLS. Note in particular the better match at the x-torque and y-torque levels. As both IDIM-WLS and IDIM-SQP-WLS underwent the very same geometric relaxation process, the observed difference in terms of wrench prediction results can therefore not be correlated to improper excitation but rather to physical consistency issues and estimated link CoM considerations. Note that physical consistency or

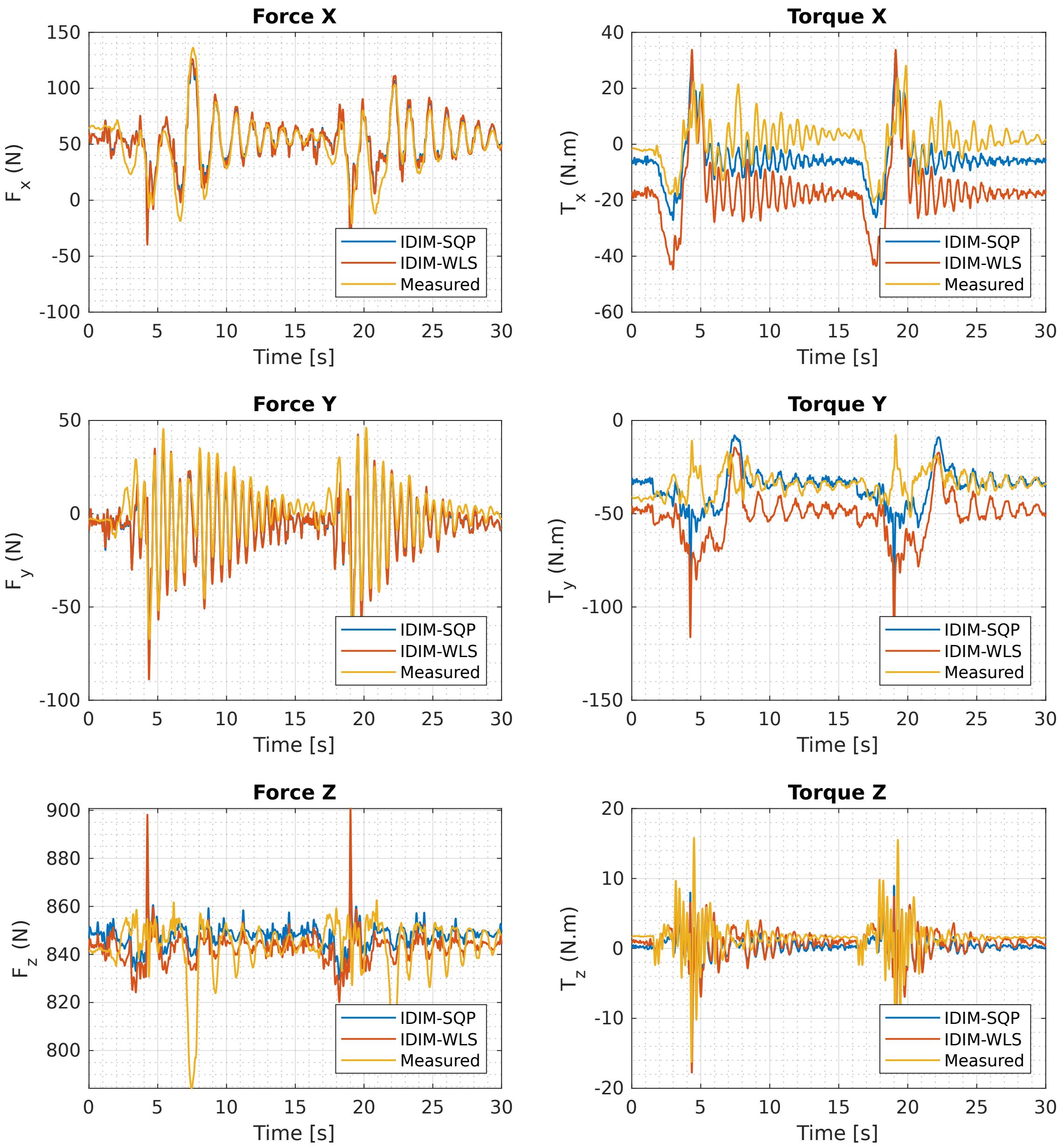


Figure 57 Cross validation experiment of the external wrench for IDIM-WLS and IDIM-SQP-WLS. In yellow, the measured values, in orange the IDIM-WLS values,and in blue the IDIM-SQP-WLS values.

inconsistency of a set of parameters can be verified in practice using a Cholesky decomposition of the pseudo-inertial matrix associated with every single robot link. This process can be executed at every iteration of the identification algorithm and used as a fail-safe feature.

It should be emphasized that although the match on the training trajectory might sometimes be better in the case of IDIM-WLS, the IDIM-SQP-WLS method eventually tends to provide the most accurate predictions on validation trajectories as the estimates are – by construction – set to be physically consistent. Moreover, as noticeable in Fig. 56, it should be emphasized that unlike IDIM-WLS, the constrained IDIM-SQP-WLS estimates comply with the robot geometrical features. These good general performances should nevertheless be tempered in practice by the occurrence of failure modes at the link CoM position constraint level, that may

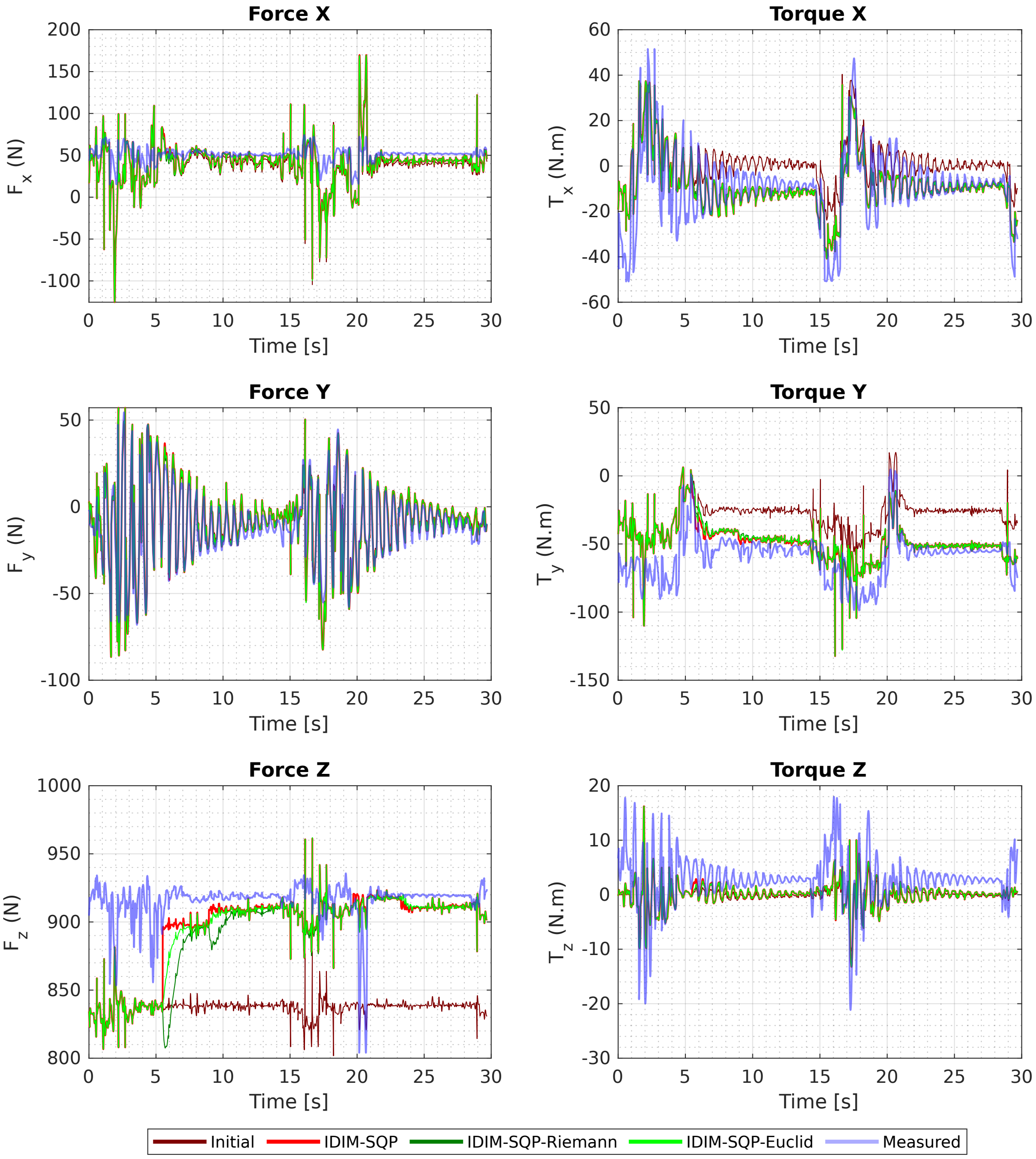
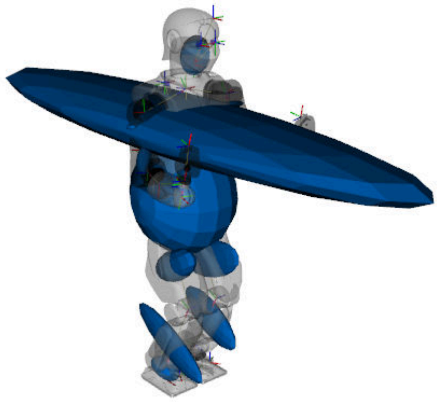
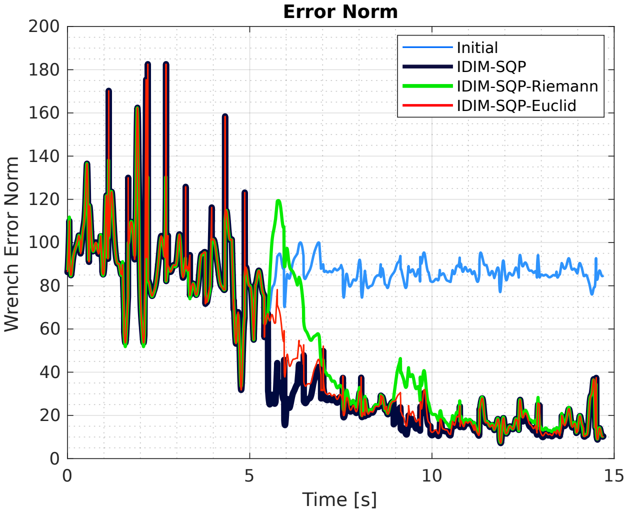


Figure 58 Online identification: note the parameter adaptation from t = 6s onwards.

sometimes lead to infeasibility. Such constraints failure modes were observed during strong parameter variations between two consecutive iterations of the algorithm. Nevertheless this issue could be mitigated by properly tuning the problem bound parameter ϖ within eq. (178). The experiment results of the online identification experiment are presented in Fig.58. The set of identified parameters can be visualized in Fig.59 along with the evolution of the wrench error norm during the identification process. The results suggest that the IDIM-SQP-WLS method is able to run online. Notice on Fig.58 the fast adaptation of the z component of net contact wrench to the 8kg payload manipulated by the robot. Convergence towards the new set of inertial parameters is achieved in 5s, thereby making the integration of IDIM-SQP-WLS into closed-loop adaptive stabilization strategies possible. This specific task is here considered as future works. Note that the convergence speed can be adjusted in practice through



(i) IDIM-SQP-WLS online identified parameters. Based on the skin feedback, only the torso link, coupled to the payload, was reidentified.



(ii) Wrench error norm. The blue curve was obtained with the initial parameter set, the dark blue curve with the updated set, the red curve with the Euclidean filtered parameters and the green curve with the Riemannian filtered parameters.

Figure 59 Online inertial parameter identification.

the parameter α within eq. (183) and eq. (185). It is worth highlighting the benefits related to the use of an artificial skin in terms in terms of generating appropriate equality constraints applied to the inertial parameters of the robot bodies that are not involved in the payload interaction process.

6.6. Summary

This chapter introduces a new physically consistent inertial parameter identification algorithm and explores the possible applications of a multimodal artificial skin in the context of inertial parameter identification of humanoid robots. Adopting a new parametrization of the physically consistent identification problem allows convenient implementation in the form of a sequence of QP, capable of running online on a robot's embedded computer. Additionally, the Riemannian nature of the physical consistency manifold is exploited for relaxation and interpolation purposes, thereby addressing the issues of data rank deficiency and parameter continuity. The artificial skin is being used to hot-start the identification process, set additional constraints to the identification process and finally provide real-time feedback on the nature of the mechanical coupling between a robot and its payload, thereby allowing significant simplifications of the whole the identification process. The performance of the proposed approach is demonstrated both in simulation, using the BIRDy identification benchmark, and through a set of experiments on a humanoid robot. The MCS results suggest that the proposed IDIM-SQP approach has enhanced noise robustness compared to the other methods, while guaranteeing physicality of the estimates. Preliminary experiment results performed on a REEM-C humanoid robot show its ability to run online within a robot control framework.

Chapter 7

General Conclusion

7.1. Summary and Conclusions

Adaptability is the key to the upcoming large-scale democratization of complex robotic systems in society. Its implementation essentially relies on the ability of a system to assess the nature of its environment and to understand its properties. Only then can coherent control actions be generated. Multiple researches suggest that advanced multimodal sensory capabilities are essential to enable robots to physically interact and even work alongside humans on a daily basis. More specifically, it appears that physical interaction between humans and robots should be based on human-like sensory modalities, and in particular on a reliable sense of touch. In this context, the development of a large-scale multimodal artificial skin, which can potentially equip any robot, seems to be a very promising research direction.However, its implementation remains problematic in practice. This is mainly due, on the one hand, to the substantial amount of data that can potentially be generated by this class of highly distributed systems and, on the other hand, to the fact that it modifies the dynamic properties of the robot on which it is mounted. This thesis addresses some aspects of these problematics from the perspective of control, estimation and identification.

7.1.1. Whole-Body Human-Robot Physical Interaction

This chapter addresses the issue of whole-body compliance with force propagation for wheeled mobile manipulators using the multimodal tactile feedback of an artificial robot skin. A new control paradigm is proposed, leveraging the contact and precontact modalities of the skin. This control strategy is based on the reactive acceleration error between each of the robot's rigid links, and a corresponding set of virtual mass-spring-damper systems, moving with a predefined compliant dynamics. The use of Quadratic programming allows determining a set of suitable control torques references, to track as closely as possible the desired reactive behavior while intrinsically complying with the robot physical constraints. In addition to its intuitiveness, formulating the controller in Cartesian space, allows to conveniently capture the dynamic residuals between the desired and the actual reactive motions of each limb, and to propagate them onto a parent kinematic chain. In this manner, the robot remains compliant under external tactile perturbations, even when the arms reach kinematic or dynamic constraints. The artificial skin here plays a major role since it provides information on the magnitude, direction and localization of tactile interactions. The experiments performed on a dual arm mobile manipulator demonstrate the reliability and efficiency of the proposed approach as well as its applicability to human-robot interaction scenarios.

7.1.2. Estimation of High-Order Motion Derivative Using Distributed Inertial Feedback

This chapter describes a new derivation-free estimation method for second-order kinematics on floating-base robots. Relying on highly-redundant distributed acceleration measurement from an artificial sensory skin covering the robot, the proposed method is two-fold. The first step consists in fusing the skin accelerations link-wise using a dedicated Maximum-Likelihood observer. The second step then involves fusing these measurements with that of the encoders and floating-base IMU. The results suggest that the motion derivative estimates have equivalent noise levels to those of the reference signals while maintaining a low lag. In addition, this chapter addresses the issues of measurement robustness and data flow mitigation on redundant inertial sensor arrays by introducing a new online sensor selection algorithm. As a matter of fact, direct physical interaction with an acceleration sensor result in strong parasitic transients that can potentially jeopardize estimation performance. Sensor hyper-redundancy can also lead to network jamming, yielding nondeterministic latency. In this chapter, both issues are dealt with using a sensor selection approach, triggered by the precontact modality of the artificial skin. Relying on a projected gradient heuristics, the proposed sensor selection algorithm evaluates sensor configurations that provide the best measurement signal-to-noise ratio (SNR) and is capable of converging in less than ten iterations, which makes it well suited for online implementation on a robotic system.

7.1.3. Quantitative Performance Assessment of State-of-the-art Identification Algorithms

In this work, a sample of the most popular approaches to inertial parameter identification for fixed-based robotic systems is evaluated and benchmarked. These methods are IDIM-OLS, IDIM-WLS, IDIM-IRLS, IDIM-TLS, ML, IDIM-IV, DIDIM, CLIE, CLOE, EKF, SREKF, UKF, SRUKF, CDKF, SRCDKF, AdaNN, HTRNN, PC-IDIM-OLS, PC-IDIM-WLS, PC-IDIM-IRLS, PC-DIDIM and PC-IDIM-IV. First presented and discussed in a theoretical manner, each method is then implemented and evaluated experimentally within a dedicated framework, named BIRDy, which was specifically developed for this purpose. BIRDy features a complete identification pipeline, allowing one to generate the kinematic and dynamic models of a given robot, to compute a trajectory that excites its dynamic parameters, to simulate the system's behavior along this trajectory, collect experimental data under well-defined conditions, proceed to parameter identification using a pool of dedicated algorithms and eventually to compare the identification performances using a set of suitable metrics. In this work, BIRDy was used to perform Monte Carlo simulations on two models of 6-DoF industrial robot manipulators, namely the Staubli TX40 and the Mitsubishi RV2SQ. Experiments were also carried out on the real robots, thereby providing helpful insight on the influence of multiple factors over the performance of identification, including prior knowledge of the control architecture, sampling frequency, or friction model. The results make it possible to infer a set of general guidelines based on quantitative arguments regarding the applicability of a given identification method to a particular experimental context.

7.1.4. Online Physically-Consistent Inertial Parameter Identification with Geometric Relaxation and Local Excitation

This chapter introduces a new physically consistent batch identification algorithm capable of running both offline and online on fixed and floating base robots. The issue of parameter physical consistency is here addressed using an alternative parametrization based on a Cholesky decomposition of the pseudo-inertia matrix, generally used to assess physicality during an identification process. As a matter of fact, the resulting problem structure allows reformulating the classical SDP identification approach, based on linear matrix inequalities (LMI) as a sequence of quadratic programs (QP). Evaluation of this method on a model of Staubli TX40 robot using the BIRDy identification benchmark demonstrates excellent robustness to high noise levels in the joint position signal as well as well as a preservation of the physical consistency of the estimates. Preliminary results obtained with a REEM-C humanoid robot covered with artificial skin indicate that this method is suitable for online batch implementation, with potential applications to adaptive stabilization and whole-body manipulation. In this context, it is worth emphasizing that the tactile feedback of an artificial skin can be leveraged in order to provide relevant information regarding the nature of the coupling between the robot and its payload, thereby simplifying the identification problem, in particular through the online generation of local excitation behaviors. To avoid abrupt parameter changes induced by the batch nature of the identification algorithm, the identified values are interpolated along a geodesic curve of the physicality Riemannian manifold, eventually resulting in a continuous stream of parameter estimates, that can potentially be made available to a robot controller at each time epoch for adaptive control purposes.

7.2. Outlook and Future Works

Considering the work developed in this thesis, future research should mainly focus on a deeper integration of tactile feedback in a more general robot control framework, for compliant control, identification and adaptive manipulation. It should be emphasized that in its current version, the artificial skin is essentially a technological demonstrator. As such, it has several well-known limitations that will be overcome in future design iterations. Besides improving the hardware design of the skin, a software stack will have to be developed, to allow the easy deployment of distributed robot sensors on much larger scales. Inertial sensor calibration issues should moreover be dealt with in a more robust manner.

In essence, future works in the field of human-robot interaction will mainly consist in further integrating the developed control paradigm within a stack-of-task architecture. It should also be noted that the use of arbitrary contact surfaces for stabilization purposes is also considered as a promising research direction, to the extent that the skin contacts can be integrated into eq. (110) and hence be considered within a wrench distribution algorithm. Further exploitation of the artificial skin multimodality, especially through acceleration and temperature signals, will also have to be considered.

Future work in the field of joint high-order motion derivative estimation will consist, on the one hand, of extending the developed paradigm to an entire humanoid robot, and, on the other hand, of evaluating the propensity of the estimated acceleration signal to be used in a high-gain control loop. This will require a set of modifications at the skin firmware level, allowing in particular the *event mode* and the *clock mode* of the skin, to coexist on the same network and enabling seamless transition between them at run time and at the cell level. On the hardware point of view, the mechanical coupling between the skin and the robot to which it is attached should also be improved so that the accelerometer calibration remains valid even after intensive use.

Future works in the field of parameter identification will first be to migrate the BIRDy framework as a set of C++ ROS packages, made available to the community for online implementation and deployment on live robotic systems. It is worth emphasizing that this was already partially done as part of chapter 6 of this thesis, where new functionalities, such as robot description using the URDF file format, the consideration of floating base mechanisms and the use of an efficient rigid body dynamics library for identification model computation was proposed. In this context, it should moreover be emphasized that the use of an artificial skin for shape recognition and adaptive manipulation may reasonably be considered. The feedback from the artificial skin in terms of contact or even pre-contact could in this case be combined with that from a lidar or RGBD camera mounted on the robot in order to match a convex geometric shape, such as an ellipsoid, to the resulting point cloud of the manipulated object. This would allow in particular to constrain some aspects of the parametric identification process, such as the position of the center of mass of the manipulated object via the use of approaches similar to Ayusawa et al. [101, 102] or Wensing et al. [325]. Finally, the integration of the proposed identification approach within an adatptive stabilization loop or an adaptive trajectory generator is to be considered.

Appendix A

Robot Dynamics

This appendix chapter reviews important details related to the mathematical formulation of the dynamics of a robot.

A.1. Properties of the Inertia tensor

A.1.1. Definitions

Formulating the dynamic equations of a robot manipulator requires a precise knowledge of the inertia tensors $\boldsymbol{I}_j$ of every link j (expressed in the link Center-of-Mass):

$$\boldsymbol{I}_j = \begin{bmatrix} I_{xx_j} & I_{xy_j} & I_{xz_j} \\ I_{yx_j} & I_{yy_j} & I_{yz_j} \\ I_{zx_j} & I_{zy_j} & I_{zz_j} \end{bmatrix}, j \in \{1 \cdots n\} \tag{195}$$

where for a robot link j, I_{xx_j}, I_{yy_j} and I_{zz_j} depict the *principal moments of inertia* and where $I_{xz_j} = I_{zx_j}, I_{xy_j} = I_{yx_j}$ and $I_{yz_j} = I_{zy_j}$ are the so called *products of inertia*. Usually not provided by the robot manufacturers, these terms can nevertheless be calculated as [9]:

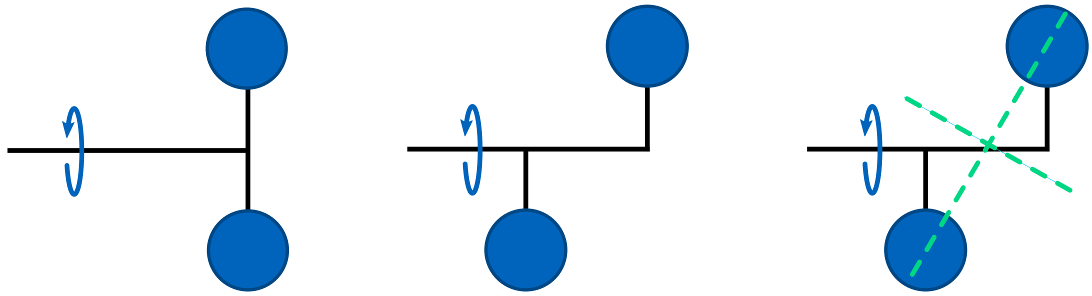
$$\begin{aligned} I_{xx_j} &= \iiint \rho_j \left(y_j^2 + z_j^2\right) dx\, dy\, dz & \qquad I_{xy_i} &= -\iiint \rho_j x_j y_j \, dx\, dy\, dz \\ I_{yy_j} &= \iiint \rho_j \left(x_j^2 + z_j^2\right) dx\, dy\, dz & \qquad I_{xz_i} &= -\iiint \rho_j x_j z_j \, dx\, dy\, dz \\ I_{zz_j} &= \iiint \rho_j \left(x_j^2 + y_j^2\right) dx\, dy\, dz & \qquad I_{yz_i} &= -\iiint \rho_j y_j z_j \, dx\, dy\, dz \end{aligned}$$

A.1.2. Main axes of inertia and elements of solid material symmetry

Because of its symmetrical nature it is always possible to choose an axis system such that the matrix $\breve{\boldsymbol{I}}$ representing $\boldsymbol{I}_j$ is diagonal (c.f. Fig.60iii). Such axes are called **main axes of inertia**. The corresponding moments of inertia are called **principal moments of inertia**. In practice, $\boldsymbol{I}_j$ can be written as:

$$\boldsymbol{I}_j = \boldsymbol{R}_j \breve{\boldsymbol{I}}_j \boldsymbol{R}_j^\top \tag{196}$$

where where $\breve{\boldsymbol{I}}_j = \mathrm{diag}(\breve{I}_{xxj}, \breve{I}_{yyj}, \breve{I}_{zzj})$ and $\boldsymbol{R}_j \in SO(3)$ is the rotation matrix relating the frame $\mathcal{I}_j$ of the principal axes of inertia, to the link's Center-of-Mass frame $\mathcal{C}_j$.



(i) Balanced rotating body

(ii) Unbalanced rotating body

(iii) Main axes of inertia

Figure 60 The unbalanced body has non-zero off-diagonal components, which represents “wobbliness” of rotation

A.1.3. The Huygens-Steiner Parallel Axis Theorem

Also known as the **parallel axis theorem**. Provided that the center-of-mass frame and the origin frame of a robot link j have the same orientation, the inertia matrix $\boldsymbol{I}_j$ at the center-of-mass frame can be related to the inertia matrix $\boldsymbol{L}_j$ at the link origin frame (i.e. DH frame) using the following formula:

$$\boldsymbol{L}_j = \boldsymbol{I}_j + M_j \left[\boldsymbol{p}_j^{cm_j}\right]_\times^\top \left[\boldsymbol{p}_j^{cm_j}\right]_\times \tag{197}$$

where M_j is the mass of link j, $\boldsymbol{p}_j^{cm_j} = [X_j, Y_j, Z_j]^\top$ is the coordinate vector of the link center-of-mass related to the link origin frame (i.e. DH frame) and expressed in the base coordinate frame and $\forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^3 \; [\boldsymbol{u}]_\times \boldsymbol{v} = \boldsymbol{u} \times \boldsymbol{v}$. It is worth noting that (197) is in fact the Schur complement (c.f. B.2) of a matrix $\boldsymbol{D}_j(\boldsymbol{\chi}) \in \mathbb{S}^{6\times 6}$ defined as

$$\begin{aligned}
\boldsymbol{D}_j(\boldsymbol{\chi}) &= \begin{bmatrix} \boldsymbol{I}_j + M_j \left[\boldsymbol{p}_j^{cm_j}\right]_\times^\top \left[\boldsymbol{p}_j^{cm_j}\right]_\times & M_j \left[\boldsymbol{p}_j^{cm_j}\right]_\times^\top \\ M_j \left[\boldsymbol{p}_j^{cm_j}\right]_\times & M_j \mathbf{1}_{3\times 3} \end{bmatrix} && \text{(198a)} \\
&= \begin{bmatrix} \boldsymbol{L}_j & \left[\boldsymbol{mp}_j^{cm_j}\right]_\times^\top \\ \left[\boldsymbol{mp}_j^{cm_j}\right]_\times & M_j \mathbf{1}_{3\times 3} \end{bmatrix} && \text{(198b)}
\end{aligned}$$

where $\mathbf{0}_{3\times 3}, \mathbf{1}_{3\times 3}$ respectively denote the 3×3 zero and identity matrices and where $\boldsymbol{mp}_j^{cm_j} = [M_j X_j, M_j Y_j, M_j Z_j]^\top$. It is also worth noting that $\boldsymbol{L}_j$ can be constructed from the standard parameter vector $\boldsymbol{\chi}_j$ as:

$$\forall \boldsymbol{u} \in \mathbb{R}^3, \; \boldsymbol{L}_j \boldsymbol{u} = \begin{bmatrix} XX_j & XY_j, & XZ_j \\ XY_j & YY_j, & YZ_j \\ XZ_j & YZ_j, & ZZ_j \end{bmatrix} \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix} = \underbrace{\begin{bmatrix} u_x & u_y & u_z & 0 & 0 & 0 \\ 0 & u_x & 0 & u_y & u_z & 0 \\ 0 & 0 & u_x & 0 & u_y & u_z \end{bmatrix}}_{[\boldsymbol{u}]_\bullet} \underbrace{\begin{bmatrix} XX_j \\ XY_j \\ XZ_j \\ YY_j \\ YZ_j \\ ZZ_j \end{bmatrix}}_{\boldsymbol{l}_j} \tag{199}$$

A.2. Closed-form Symbolic Formulation of Robot Dynamics

A.2.1. Robot Dynamic Equations

The dynamic model of a robot provides a mapping between the generalized forces $\boldsymbol{\tau}$ acting on its joints (e.g. gravity, contact wrenches, control torques...) and the resulting accelerations they produce. It consists in a system of *second order differential equations* of the form:

$$\Phi(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) = \boldsymbol{\tau} \tag{200}$$

As exposed in [360], these equations can be established ether by using the *Newton-Euler* approach (balance of forces/torques) or by using the *Euler-Lagrange* method (energy-based approach). In Euler-Lagrange formalism, we define the Lagrange parameter $\mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}}, t)$ as the difference between the robot *total kinetic energy* $K_T(\dot{\boldsymbol{q}}, t) = \sum_{i=1}^{n} K_i(\dot{\boldsymbol{q}}, t)$ and its *total potential energy* $P_T(\boldsymbol{q}, t) = \sum_{i=1}^{n} P_i(\boldsymbol{q}, t)$ (defined as the sum of every link contributions), yielding

$$\mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}}, t) = K_T(\dot{\boldsymbol{q}}, t) - P_T(\boldsymbol{q}, t) \tag{201}$$

According to the *Hamilton's principle of least action*, any robot motion between two configurations $\boldsymbol{q}(t_1)$ and $\boldsymbol{q}(t_2)$ have to be done minimizing the action parameter $\mathcal{S}$ defined as

$$\mathcal{S}(\boldsymbol{q}) = \int_{t_1}^{t_2} \mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}}, t) dt \tag{202}$$

eventually resulting in the so called Lagrange's equation, describing the Robot's dynamics:

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\boldsymbol{q}}}\right) - \frac{\partial \mathcal{L}}{\partial \boldsymbol{q}} = \boldsymbol{\tau} \tag{203}$$

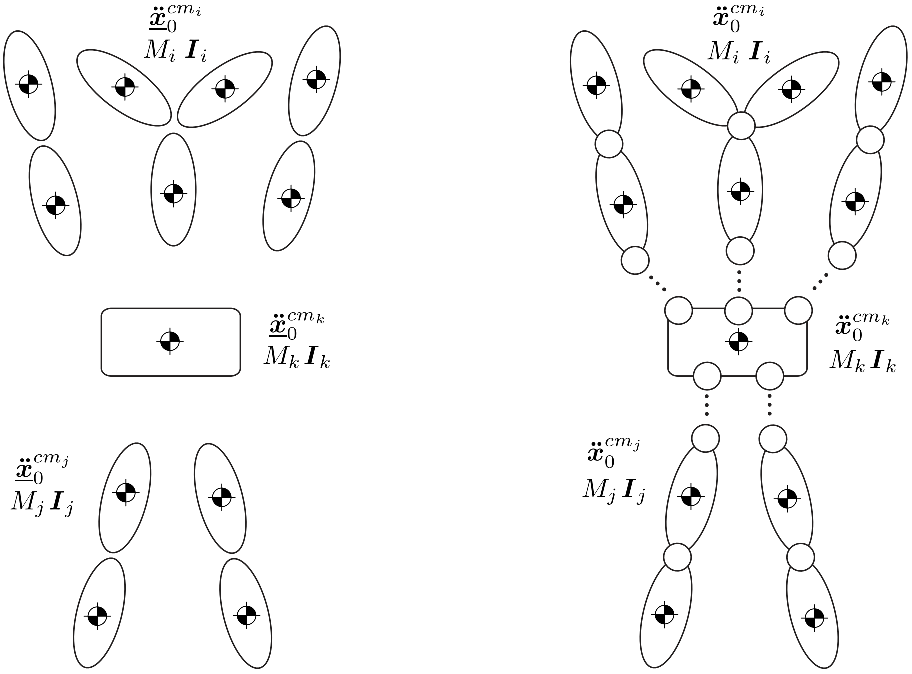
where for a n-DoF mechanism, $\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\tau} \in \mathbb{R}^n$ are n-dimensional vectors representing respectively *generalized positions*, *generalized velocities* and *generalized joint forces* such as the control torque $\boldsymbol{\tau}_{ctrl} \in \mathbb{R}^n$, the external loads $\boldsymbol{\tau}_{ext} \in \mathbb{R}^n$, and non-conservative generalized forces such as friction. As the potential energy of a robot manipulator does not depends on $\dot{\boldsymbol{q}}$, eq.(203) can thus be reformulated as:

$$\frac{d}{dt}\left(\frac{\partial K_T}{\partial \dot{\boldsymbol{q}}}\right) - \frac{\partial K_T}{\partial \boldsymbol{q}} + \frac{\partial P_T}{\partial \boldsymbol{q}} = \boldsymbol{\tau} \tag{204}$$

Computer implementation of such an equation has of course to be done using matrix formalism, leading eventually to the following canonical expression, used in chapter 2

$$\boldsymbol{M}(\boldsymbol{\chi}, \boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{\chi}, \boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{\chi}, \boldsymbol{q}) + \boldsymbol{\zeta}(\boldsymbol{\chi}, \dot{\boldsymbol{q}}) = \boldsymbol{\tau}_{ctrl} + \boldsymbol{\tau}_{ext} \tag{205}$$

The following subsection aims at clarifying the robot dynamic computations executed with the BIRDy robot identification benchmark, namely the computation of the terms $\boldsymbol{M}(\boldsymbol{\chi}, \boldsymbol{q}), \boldsymbol{C}(\boldsymbol{\chi}, \boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}$, $\boldsymbol{g}(\boldsymbol{\chi}, \boldsymbol{q})$ as well as the regression matrix used for identification purposes.


$\ddot{\underline{\boldsymbol{x}}}_0^{cm_i}$
$M_i\,\boldsymbol{I}_i$
$\ddot{\underline{\boldsymbol{x}}}_0^{cm_k}$
$M_k\boldsymbol{I}_k$
$\ddot{\underline{\boldsymbol{x}}}_0^{cm_j}$
$M_j\,\boldsymbol{I}_j$
$\ddot{\boldsymbol{x}}_0^{cm_i}$
$M_i\,\boldsymbol{I}_i$
$\ddot{\boldsymbol{x}}_0^{cm_k}$
$M_k\boldsymbol{I}_k$
$\ddot{\boldsymbol{x}}_0^{cm_j}$
$M_j\,\boldsymbol{I}_j$

(i) Unconstrained set or rigid bodies

(ii) Articulated (constrained) bodies

A.2.2. The Gauss-Newton Method

The Gauss principle of dynamics [361] states that:

- The acceleration $\ddot{\boldsymbol{x}}_0^{cm_j} = \left[\dot{\boldsymbol{v}}_0^{cm_j\top}, \dot{\boldsymbol{\omega}}_0^{cm_j\top}\right]^\top$ of a set of solids subject to a set of constraints, deviates the **least possible** (in the L2 norm sense) from the acceleration $\ddot{\underline{\boldsymbol{x}}}_0^{cm_j} = \left[\dot{\underline{\boldsymbol{v}}}_0^{cm_j\top}, \dot{\underline{\boldsymbol{\omega}}}_0^{cm_j\top}\right]^\top$ that it would have had without these constraints.
- The deviation is measured using the total kinetic energy as a cost function:

$$\mathcal{D} = \sum_{j=1}^{n}\left[\frac{1}{2}\left(\dot{\boldsymbol{v}}_0^{cm_j} - \dot{\underline{\boldsymbol{v}}}_0^{cm_j}\right)^\top M_j \left(\dot{\boldsymbol{v}}_0^{cm_j} - \dot{\underline{\boldsymbol{v}}}_0^{cm_j}\right) + \frac{1}{2}\left(\dot{\boldsymbol{\omega}}_0^{cm_j} - \dot{\underline{\boldsymbol{\omega}}}_0^{cm_j}\right)^\top \boldsymbol{I}_0^j \left(\dot{\boldsymbol{\omega}}_0^{cm_j} - \dot{\underline{\boldsymbol{\omega}}}_0^{cm_j}\right)\right] \tag{206}$$

  where $\boldsymbol{I}_0^j$ is the inertia matrix of link j at its **center of mass** relative to the base reference frame. For the record, always remember that the inertia matrix $\boldsymbol{I}_j$ can be related to the inertia matrix $\boldsymbol{L}_j$ of link j at the link DH frame using the parallel axis theorem.
- Unconstrained rigid body equation of motion:

$$\boldsymbol{w}_0^{cm_j} = \begin{bmatrix} \boldsymbol{f}_0^{cm_j} \\ \boldsymbol{\mu}_0^{cm_j} \end{bmatrix} = \begin{bmatrix} M_j \dot{\underline{\boldsymbol{v}}}_0^{cm_j} \\ \boldsymbol{I}_0^j \dot{\underline{\boldsymbol{\omega}}}_0^{cm_j} - \left[\boldsymbol{I}_0^j \underline{\boldsymbol{\omega}}_0^{cm_j}\right]_\times \underline{\boldsymbol{\omega}}_0^{cm_j} \end{bmatrix} \tag{207}$$

- Constrained rigid body second-order kinematics:

$$\ddot{\boldsymbol{x}}_0^{cm_j} = \begin{bmatrix} \dot{\boldsymbol{v}}_0^{cm_j} \\ \dot{\boldsymbol{\omega}}_0^{cm_j} \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{J}}_v^{cm_j} \dot{\boldsymbol{q}} + \boldsymbol{J}_v^{cm_j} \ddot{\boldsymbol{q}} \\ \dot{\boldsymbol{J}}_\omega^{cm_j} \dot{\boldsymbol{q}} + \boldsymbol{J}_\omega^{cm_j} \ddot{\boldsymbol{q}} \end{bmatrix} \tag{208}$$

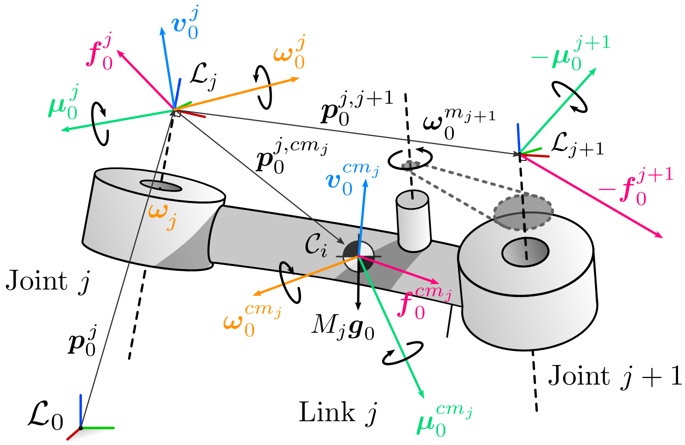


Figure 61 Illustration of the motions of a rigid robot with respect to a base reference frame $\mathcal{L}_0$. For each link j of the robot, we define a rigidly-attached body reference frame $\mathcal{L}_j$ and as well as a Center-of-Mass frame $\mathcal{C}_j$. $\mathcal{L}_j$ and $\mathcal{C}_j$ are chosen to have the same orientation in space.

- Constrained rigid body equation of motion is obtained by expressing the first-order optimality conditions over (206) (i.e. by derivation w.r.t the joint space variable $\ddot{\boldsymbol{q}}$ of the constrained system)

$$\frac{\partial \mathcal{D}}{\partial \ddot{\boldsymbol{q}}} := \mathbf{0}_{n \times 1} \tag{209}$$

The final step consists in identifying, in the obtained expression, the terms multiplied by $\ddot{\boldsymbol{q}}$ (corresponding to $\boldsymbol{M}(\boldsymbol{\chi}, \boldsymbol{q})$), the terms multiplied by $\dot{\boldsymbol{q}}$ (corresponding to $\boldsymbol{C}(\boldsymbol{\chi}, \boldsymbol{q}, \dot{\boldsymbol{q}})$) and the terms multiplied by nothing (corresponding to $\boldsymbol{g}(\boldsymbol{\chi}, \boldsymbol{q})$, yielding:

$$\boldsymbol{M}(\boldsymbol{\chi}, \boldsymbol{q}) = \sum_{j=1}^{n} \left[M_j \boldsymbol{J}_v^{cm_j \top} \boldsymbol{J}_v^{cm_j} + \boldsymbol{J}_\omega^{cm_j \top} \boldsymbol{R}_0^j \boldsymbol{I}_j \boldsymbol{R}_0^{j \top} \boldsymbol{J}_\omega^{cm_j} \right] \tag{210}$$

$$\begin{aligned} \boldsymbol{C}(\boldsymbol{\chi}, \boldsymbol{q}, \dot{\boldsymbol{q}}) &= \sum_{j=1}^{n} \Big[M_j \boldsymbol{J}_v^{cm_j \top} \dot{\boldsymbol{J}}_v^{cm_j} + \boldsymbol{J}_\omega^{cm_j \top} \boldsymbol{R}_0^j \boldsymbol{I}_j \boldsymbol{R}_0^{j \top} \dot{\boldsymbol{J}}_\omega^{cm_j} \cdots \\ & \cdots + \boldsymbol{J}_\omega^{cm_j \top} \left[\boldsymbol{J}_\omega^{cm_j} \dot{\boldsymbol{q}} \right]_\times \boldsymbol{R}_0^j \boldsymbol{I}_j \boldsymbol{R}_0^{j \top} \boldsymbol{J}_\omega^{cm_j} \Big] \end{aligned} \tag{211}$$

A.2.3. The Newton-Euler Method

Let us consider an intermediate link j of a n-DoF robot (c.f. Fig.61). The resultant of force and momentum applied to this link in its Center-of-Mass are denoted as $\boldsymbol{f}_0^{cm_j}$ and $\boldsymbol{\mu}_0^{cm_j}$. Accordingly, the linear and angular momentum of the link can be expressed as:

$$\boldsymbol{\nu}_v^{cm_j} = M_j \boldsymbol{v}_0^{cm_j} = M_j \boldsymbol{J}_v^{cm_j} \dot{\boldsymbol{q}} \tag{212a}$$

$$\boldsymbol{\nu}_\omega^{cm_j} = \boldsymbol{I}_0^j \boldsymbol{\omega}_0^{cm_j} = \boldsymbol{R}_0^j \boldsymbol{I}_j \boldsymbol{R}_0^{j \top} \boldsymbol{J}_\omega^{cm_j} \dot{\boldsymbol{q}} \tag{212b}$$

Taking the time derivative of the momentum equations yields:

$$\begin{aligned}
\dot{\boldsymbol{\nu}}_v^{cm_j} &= M_j \dot{\boldsymbol{v}}_0^{cm_j} \\
&= M_j \boldsymbol{J}_v^{cm_j} \ddot{\boldsymbol{q}} + M_j \dot{\boldsymbol{J}}_v^{cm_j} \dot{\boldsymbol{q}} \qquad \text{(213a)} \\
&= \boldsymbol{f}_0^{cm_j} \\
\dot{\boldsymbol{\nu}}_\omega^{cm_j} &= \boldsymbol{R}_0^j \boldsymbol{I}_j \boldsymbol{R}_0^{j\top} \dot{\boldsymbol{\omega}}_0^{cm_j} + \dot{\boldsymbol{R}}_0^j \boldsymbol{I}_j \boldsymbol{R}_0^{j\top} \boldsymbol{\omega}_0^{cm_j} + \cancel{\boldsymbol{R}_0^j \boldsymbol{I}_j \dot{\boldsymbol{R}}_0^{j\top} \boldsymbol{\omega}_0^{cm_j}} \\
&= \boldsymbol{R}_0^j \boldsymbol{I}_j \boldsymbol{R}_0^{j\top} \boldsymbol{J}_\omega^{cm_j} \ddot{\boldsymbol{q}} + \left[\boldsymbol{R}_0^j \boldsymbol{I}_j \boldsymbol{R}_0^{j\top} \dot{\boldsymbol{J}}_\omega^{cm_j} + \left[\boldsymbol{J}_\omega^{cm_j} \dot{\boldsymbol{q}} \right]_\times \boldsymbol{R}_0^j \boldsymbol{I}_j \boldsymbol{R}_0^{j\top} \boldsymbol{J}_\omega^{cm_j} \right] \dot{\boldsymbol{q}} \qquad \text{(213b)} \\
&= \boldsymbol{\mu}_0^{cm_j}
\end{aligned}$$

Noticing that (213b) and (214) are homogeneous to a force and a torque, it is possible to propagate these equations along the kinematic tree of the robot by simply premultiplying by the Jacobian transpose of the link. Summing up these equations for every robot joint eventually yields

$$\begin{aligned}
&\underbrace{\sum_{j=1}^{n} \left[M_j \boldsymbol{J}_v^{cm_j\top} \boldsymbol{J}_v^{cm_j} + \boldsymbol{J}_\omega^{cm_j\top} \boldsymbol{R}_0^j \boldsymbol{I}_j \boldsymbol{R}_0^{j\top} \boldsymbol{J}_\omega^{cm_j} \right] \ddot{\boldsymbol{q}} + \cdots}_{\boldsymbol{M}(\boldsymbol{\chi},\boldsymbol{q})} \\
&\underbrace{\sum_{j=1}^{n} \left[M_j \boldsymbol{J}_v^{cm_j\top} \dot{\boldsymbol{J}}_v^{cm_j} + \boldsymbol{J}_\omega^{cm_j\top} \boldsymbol{R}_0^j \boldsymbol{I}_j \boldsymbol{R}_0^{j\top} \dot{\boldsymbol{J}}_\omega^{cm_j} + \boldsymbol{J}_\omega^{cm_j\top} \left[\boldsymbol{J}_\omega^{cm_j} \dot{\boldsymbol{q}} \right]_\times \boldsymbol{R}_0^j \boldsymbol{I}_j \boldsymbol{R}_0^{j\top} \boldsymbol{J}_\omega^{cm_j} \right] \dot{\boldsymbol{q}} \cdots}_{\boldsymbol{C}(\boldsymbol{\chi},\boldsymbol{q},\dot{\boldsymbol{q}})} \qquad \text{(214)} \\
&= \sum_{j=1}^{n} \left[\boldsymbol{J}_v^{cm_j\top} \boldsymbol{f}_0^{cm_j} + \boldsymbol{J}_\omega^{cm_j\top} \boldsymbol{\mu}_0^{cm_j} \right]
\end{aligned}$$

This basically amounts to re-injecting (210) and (211) in (1), considering the gravitational effects as being part of $\boldsymbol{f}_0^{cm_j}$ while the friction and control efforts – in the case of revolute joints – are part of $\boldsymbol{\mu}_0^{cm_j}$.

A.2.4. Symbolic Computation of the Regressor Matrix

Although elegant, the previous expressions usually cannot be used "as is" to compute a robot regressor as pointed out in [309]. This is due to the fact that the Jacobian terms in these expressions have a nonlinear dependence on the position $\boldsymbol{p}_j^{cm_j} = [X_j Y_j Z_j]^\top$ of the Center-of-Mass of every link j relative to the link origin frame (a parameter that will later have to be estimated using dedicated identification techniques).

Let us consider the linear velocity $\boldsymbol{v}_0^j$ of a body-attached reference frame $\mathcal{L}_j$ (such as a DH frame or a URDF link frame) instead of that of the Center-of-Mass frame $\mathcal{C}_j$. $\boldsymbol{v}_0^j$ and $\boldsymbol{v}_0^{cm_j}$ can be related to each other following

$$\boldsymbol{v}_0^{cm_j} = \boldsymbol{v}_0^j + \left[\boldsymbol{\omega}_0^{cm_j} \right]_\times \boldsymbol{p}_j^{cm_j} = \left(\boldsymbol{J}_v^j - \left[\boldsymbol{p}_j^{cm_j} \right]_\times \boldsymbol{J}_\omega^{cm_j} \right) \dot{\boldsymbol{q}}. \qquad (215)$$

Since $\boldsymbol{v}_0^{cm_j} = \boldsymbol{J}_v^{cm_j}\dot{\boldsymbol{q}}$, we can relate the Jacobian matrix $\boldsymbol{J}_v^{cm_j}$ mapping the joint velocity $\dot{\boldsymbol{q}}$ to the linear velocity $\boldsymbol{v}_0^{cm_j}$ of $\mathcal{C}_j$, to the Jacobian matrix $\boldsymbol{J}_v^j$ mapping $\dot{\boldsymbol{q}}$ to the linear velocity $\boldsymbol{v}_0^j$ of $\mathcal{L}_j$ as

$$\boldsymbol{J}_v^{cm_j} = \boldsymbol{J}_v^j - \left[\boldsymbol{p}_j^{cm_j}\right]_\times \boldsymbol{J}_\omega^{cm_j} \tag{216}$$

Note that we have $\boldsymbol{J}_\omega^{cm_j} = \boldsymbol{J}_\omega^j$ as $\boldsymbol{\omega}_0^{cm_j} = \boldsymbol{\omega}_0^j$. Re-injecting (215) and (216) into (213) and premultiplying by the full CoM Jacobian transpose $\boldsymbol{J}_0^{cm_j \top}$, we can get:

$$\begin{aligned}
\boldsymbol{J}_v^{cm_j \top}\dot{\boldsymbol{\nu}}_v^{cm_j} &= M_j \boldsymbol{J}_v^{cm_j \top}\dot{\boldsymbol{v}}_0^{cm_j} && \text{(217a)}\\
&= M_j \left(\boldsymbol{J}_v^j - \left[\boldsymbol{p}_j^{cm_j}\right]_\times \boldsymbol{J}_\omega^{cm_j}\right)^\top \left(\dot{\boldsymbol{v}}_0^j + \left[\dot{\boldsymbol{\omega}}_0^{cm_j}\right]_\times \boldsymbol{p}_j^{cm_j} + \left[\boldsymbol{\omega}_0^{cm_j}\right]_\times \dot{\boldsymbol{p}}_j^{cm_j}\right)\\
\boldsymbol{J}_\omega^{cm_j \top}\dot{\boldsymbol{\nu}}_\omega^{cm_j} &= \boldsymbol{J}_\omega^{cm_j \top}\boldsymbol{R}_0^j \boldsymbol{I}_j \boldsymbol{R}_0^{j\top}\dot{\boldsymbol{\omega}}_0^{cm_j} + \boldsymbol{J}_\omega^{cm_j \top}\dot{\boldsymbol{R}}_0^j \boldsymbol{I}_j \boldsymbol{R}_0^{j\top}\boldsymbol{\omega}_0^{cm_j} && \text{(217b)}\\
&= \boldsymbol{J}_\omega^{cm_j \top}\boldsymbol{R}_0^j \boldsymbol{I}_j \boldsymbol{R}_0^{j\top}\dot{\boldsymbol{\omega}}_0^{cm_j} + \boldsymbol{J}_\omega^{cm_j \top}\left[\dot{\boldsymbol{\omega}}_0^{cm_j}\right]_\times \boldsymbol{R}_0^j \boldsymbol{I}_j \boldsymbol{R}_0^{j\top}\boldsymbol{\omega}_0^{cm_j}
\end{aligned}$$

The resultant force $\boldsymbol{f}_0^{cm_j}$ and torque $\boldsymbol{\mu}_0^{cm_j}$ acting on the link center of mass are a combination of the control efforts $\boldsymbol{f}_c, \boldsymbol{\mu}_c$ generated by the robot actuators, the actuator and transmission-chain inertia $\boldsymbol{f}_a, \boldsymbol{\mu}_a$, of the link own weight $\boldsymbol{f}_g = M_i \boldsymbol{R}_0^W \left[0, 0, g\right]^\top, \boldsymbol{\mu}_g = \boldsymbol{0}_{3\times1}$, of the friction effects $\boldsymbol{f}_\zeta, \boldsymbol{\mu}_\zeta$ and of the physical interactions between the robot and its environment, denoted by $\boldsymbol{f}_{ext}$ and $\boldsymbol{\mu}_{ext}$, yielding

$$\begin{cases}
\boldsymbol{f}_0^{cm_j} = \boldsymbol{f}_c + \boldsymbol{f}_a + M_i \boldsymbol{R}_0^W \left[0, 0, g\right]^\top + \boldsymbol{f}_\zeta + \boldsymbol{f}_{ext}\\
\boldsymbol{\mu}_0^{cm_j} = \boldsymbol{\mu}_c + \boldsymbol{\mu}_a + \boldsymbol{\mu}_\zeta + \boldsymbol{\mu}_{ext}
\end{cases} \tag{218}$$

Therefore we can write:

$$\begin{aligned}
&\sum_{j=1}^{n} M_j \boldsymbol{J}_v^{j\top}\dot{\boldsymbol{v}}_0^j + \sum_{j=1}^{n} M_j \left(\boldsymbol{J}_v^{j\top}\left[\dot{\boldsymbol{\omega}}_0^{cm_j}\right]_\times + \boldsymbol{J}_v^{j\top}\left[\boldsymbol{\omega}_0^{cm_j}\right]_\times \left[\boldsymbol{\omega}_0^{cm_j}\right]_\times - \boldsymbol{J}_\omega^{cm_j \top}\left[\dot{\boldsymbol{v}}_0^{cm_j}\right]_\times\right)\boldsymbol{p}_j^{cm_j} \cdots\\
&\cdots + \sum_{j=1}^{n} \boldsymbol{J}_\omega^{cm_j \top}\left[\boldsymbol{R}_0^j \boldsymbol{I}_j \boldsymbol{R}_0^{j\top}\dot{\boldsymbol{\omega}}_0^{cm_j} + \left[\boldsymbol{\omega}_0^{cm_j}\right]_\times \boldsymbol{R}_0^j \boldsymbol{I}_j \boldsymbol{R}_0^{j\top}\boldsymbol{\omega}_0^{cm_j}\right] \cdots && \text{(219)}\\
&\cdots + \sum_{j=1}^{n} M_j \left(\boldsymbol{J}_v^j - \left[\boldsymbol{p}_j^{cm_j}\right]_\times \boldsymbol{J}_\omega^{cm_j}\right)^\top \boldsymbol{R}_0^W \left[0, 0, g\right]^\top + \boldsymbol{\zeta}(\boldsymbol{\chi}, \dot{\boldsymbol{q}}) = \boldsymbol{\tau}_{ctrl} + \boldsymbol{\tau}_{ext}
\end{aligned}$$

Then using parallel axis theorem (197) and applying the inertia operator $[\cdot]_\bullet$ defined in (199) on the previous equation yields:

$$\begin{aligned}
&\sum_{j=1}^{n} M_j \boldsymbol{J}_v^{j\top}\left(\dot{\boldsymbol{v}}_0^j + \boldsymbol{R}_0^W \left[0, 0, g\right]^\top\right) \cdots && \text{(220)}\\
&\cdots + \sum_{j=1}^{n} M_j \left(\boldsymbol{J}_v^{j\top}\left[\dot{\boldsymbol{\omega}}_0^{cm_j}\right]_\times + \boldsymbol{J}_v^{j\top}\left[\boldsymbol{\omega}_0^{cm_j}\right]_\times \left[\boldsymbol{\omega}_0^{cm_j}\right]_\times - \boldsymbol{J}_\omega^{cm_j \top}\left[\dot{\boldsymbol{v}}_0^{cm_j} + \boldsymbol{R}_0^W \left[0, 0, g\right]^\top\right]_\times\right)\boldsymbol{p}_j^{cm_j} \cdots\\
&\cdots + \sum_{j=1}^{n} \boldsymbol{J}_\omega^{cm_j \top}\left(\boldsymbol{R}_0^j \left[\boldsymbol{R}_0^{j\top}\dot{\boldsymbol{\omega}}_0^{cm_j}\right]_\bullet + \left[\boldsymbol{\omega}_0^{cm_j}\right]_\times \boldsymbol{R}_0^j \left[\boldsymbol{R}_0^{j\top}\boldsymbol{\omega}_0^{cm_j}\right]_\bullet\right)\boldsymbol{l}_j + \boldsymbol{\zeta}(\boldsymbol{\chi}, \dot{\boldsymbol{q}}) = \boldsymbol{\tau}_{ctrl} + \boldsymbol{\tau}_{ext}
\end{aligned}$$

where $\boldsymbol{l}_j = \left[XX_j\ XY_j\ XZ_j\ YY_j\ YZ_j\ ZZ_j\right]^\top$. If the j^{th} component ζ_j of the friction force vector $\boldsymbol{\zeta}$ is expressed in the form of a linear function of the standard parameters, as it is for example the case in the Viscous-Coulomb friction model where $\forall Fc_j, Fv_j \in \mathbb{R}_+$

$$\zeta_j(\chi_j, \dot{q}_j) = \underbrace{Fc_j \cdot \mathrm{sign}(\dot{q}_j)}_{\text{Coulomb friction}} + \underbrace{Fv_j \cdot \dot{q}_j}_{\text{Viscous friction}} \tag{221}$$

the friction coefficients can be readily included within the link's standard dynamic parameter vector $\boldsymbol{\chi}_j \in \mathbb{R}^{13}$. In this form, all the terms of the dynamic equation (220) are expressed linearly in the standard inertial parameter vector $\boldsymbol{\chi}$. Ultimately, the regression equation can therefore be constructed as:

$$\boxed{\boldsymbol{\tau}_{ctrl} + \boldsymbol{\tau}_{ext} = \sum_{j=1}^{n} \boldsymbol{Y}_{\boldsymbol{\chi}_j}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q})\boldsymbol{\chi}_j = \sum_{j=1}^{n} \begin{bmatrix} \boldsymbol{Y}_{j1} & \boldsymbol{Y}_{j2} & \boldsymbol{Y}_{j3} & \boldsymbol{Y}_{j4} \end{bmatrix} \begin{bmatrix} \boldsymbol{l}_j \\ \boldsymbol{mp}_j \\ M_j \\ Ia_j \\ Fv_j \\ Fc_j \end{bmatrix}} \tag{222}$$

where:

$$\boldsymbol{Y}_{j1} = \boldsymbol{J}_{\omega}^{cm_j\top} \left(\boldsymbol{R}_0^j \boldsymbol{B}(\boldsymbol{R}_0^{j\top} \dot{\boldsymbol{\omega}}_0^{cm_j}) + \left[\boldsymbol{\omega}_0^{cm_j}\right]_\times \boldsymbol{R}_0^j \boldsymbol{B}(\boldsymbol{R}_0^{j\top} \boldsymbol{\omega}_0^{cm_j}) \right) \tag{223a}$$

$$\boldsymbol{Y}_{j2} = \boldsymbol{J}_v^{j\top} \left[\dot{\boldsymbol{\omega}}_0^{cm_j}\right]_\times + \boldsymbol{J}_v^{j\top} \left[\boldsymbol{\omega}_0^{cm_j}\right]_\times \left[\boldsymbol{\omega}_0^{cm_j}\right]_\times - \boldsymbol{J}_{\omega}^{cm_j\top} \left[\dot{\boldsymbol{v}}_0^{cm_j} + \boldsymbol{R}_0^W \left[0, 0, g\right]^\top\right]_\times \tag{223b}$$

$$\boldsymbol{Y}_{j3} = \boldsymbol{J}_v^{j\top} \left(\dot{\boldsymbol{v}}_0^j + \boldsymbol{R}_0^W \left[0, 0, g\right]^\top \right) \tag{223c}$$

$$\boldsymbol{Y}_{j4} = [\ddot{\boldsymbol{q}}\ \dot{\boldsymbol{q}}\ \ \mathrm{sign}(\ddot{\boldsymbol{q}})] \tag{223d}$$

and where the operators $[\cdot]_\times$ and $[\cdot]_\bullet$ are defined in section 4.2.2.

A.3. Online Numerical Computation of the Regressor Matrix in Plücker coordinates

When executing an online identification algorithm, it is essential to be capable of efficiently computing the regression matrix of the whole mechanism. Throughout the experiments performed on the REEM-C humanoid robot, described in chapter 6, the robot dynamics is computed numerically using the RBDL C++ library [322] based on the recursive Newton-Euler algorithm. Since this library exploits the spatial vector algebra with Plücker coordinates described in Featherstone et al. [77], the online computation of the regression matrix has to be formulated according to the same set of conventions. Let us consider a given robot kinematic tree, as for instance shown in figure 62. Then for a given link i within this tree, the term $\nu(i)$ refers to the link ids in the sub-tree starting at link i (depicted in green in Fig. 62). Similarly, the term $\mu(i)$ denotes the children of link i (depicted in blue in Fig. 62). In this context, let us define the binary term σ such that for two links i and j of a robot kinematic tree, $\sigma(i,j) = 1$ if link j belongs to $\nu(i)$ and $\sigma(i,j) = 0$ otherwise. Following the work of

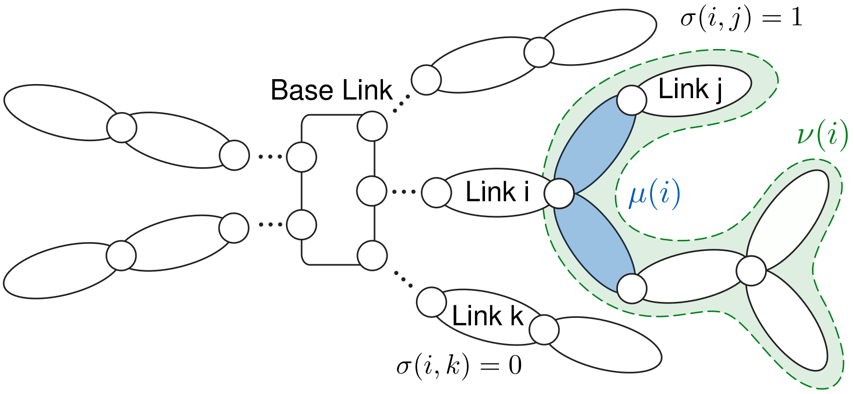


Figure 62 Notations referring to open kinematic chains.

[362, 355, 109, 356], the regressor of a kinematic tree can be computed row-wise, each row representing the contribution of a specific joint, yielding

$$\boldsymbol{Y} = \left[\boldsymbol{Y}_0^\top, \cdots, \boldsymbol{Y}_{N_B}^\top \right]^\top \tag{224}$$

where $\forall j = \{0, \cdots, N_B - 1\}$

$$\boldsymbol{Y}_j = {}^{\mu(j)}\boldsymbol{S}^\top_{\lambda(j),\mu(j)} \left[\sigma(j, \mathcal{L}(0))^{\mu(j)}\boldsymbol{X}^*_{\mathcal{L}(0)}\boldsymbol{A}_0, \cdots \sigma(j, N_B - 1)^{\mu(j)}\boldsymbol{X}^*_{N_B-1}\boldsymbol{A}_{N_B-1} \right] \tag{225}$$

In this equation, the matrix $\boldsymbol{A}_j \in \mathbb{R}^{6\times 10}$ is defined as

$$\boldsymbol{A}_j = \begin{bmatrix} \boldsymbol{0}_{3\times 1} & -\left[\boldsymbol{a}_j + \boldsymbol{\omega}_j \times \boldsymbol{v}_j\right]_\times & \left[\dot{\boldsymbol{\omega}}_j\right]_\bullet + \boldsymbol{\omega}_j \times \left[\boldsymbol{\omega}_j\right]_\bullet \\ \boldsymbol{a}_j + \boldsymbol{\omega}_j \times \boldsymbol{v}_j & \left[\dot{\boldsymbol{\omega}}_j\right]_\times + \left[\boldsymbol{\omega}_j\right]_\times \left[\boldsymbol{\omega}_j\right]_\times & \boldsymbol{0}_{3\times 6} \end{bmatrix} \in \mathbb{R}^{6\times 10} \tag{226}$$

where $\boldsymbol{v}_j, \boldsymbol{a}_j, \boldsymbol{\omega}_j \in \mathbb{R}^3$ and $\dot{\boldsymbol{\omega}}_j \in \mathbb{R}^3$ refer to the linear (resp. angular) velocity (resp. acceleration) of body j, expressed in the body frame and where

$$\forall \boldsymbol{u} \in \mathbb{R}^3, [\boldsymbol{u}]_\bullet = \begin{bmatrix} u_x & u_y & u_z & 0 & 0 & 0 \\ 0 & u_x & 0 & u_y & u_z & 0 \\ 0 & 0 & u_x & 0 & u_y & u_z \end{bmatrix} \text{ and } [\boldsymbol{u}]_\times = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix} \tag{227}$$

The floating base regression matrix is usually formulated in the floating base coordinate frame, yielding

$$\boldsymbol{Y}_b = \left[{}^b\boldsymbol{X}^*_{\mathcal{L}(0)}\boldsymbol{A}_0, \cdots, {}^b\boldsymbol{X}^*_{N_B-1}\boldsymbol{A}_{N_B-1} \right] \tag{228}$$

Finally, the regression matrix of a floating base robot can be obtained by concatenating the floating-base regressor with the mechanism regressor

$$\boldsymbol{Y} = \left[\boldsymbol{Y}_b^\top \boldsymbol{Y}_0^\top, \cdots, \boldsymbol{Y}_{N_B}^\top \right]^\top \tag{229}$$

Appendix B

Mathematical Toolbox

B.1. Alternative nullspace base computations

Given a matrix $\boldsymbol{A} \in \mathbb{R}^{n \times m}$ of rank r, then the QR decomposition of $\boldsymbol{A}^\top$, can be written as

$$\boldsymbol{A}^\top = \begin{bmatrix} \hat{\boldsymbol{Q}} & \boldsymbol{Q}_N \overset{!}{=} \boldsymbol{Z}_i \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{R}} \\ \boldsymbol{0} \end{bmatrix} \tag{230}$$

where the matrix $\hat{\boldsymbol{R}}$ is invertible. Then the last $n - r$ columns of $\boldsymbol{Q}$ form a base for the nullspace of $\boldsymbol{A}^\top$ An alternative formulation consists in using the LU decomposition of the matrix $\boldsymbol{A}$, written as

$$\boldsymbol{PAQ} = \boldsymbol{LU} \tag{231}$$

where $\boldsymbol{P} \in \mathbb{R}^{n \times n}$ is a permutation matrix, $\boldsymbol{Q} \in \mathbb{R}^{m \times m}$, $\boldsymbol{L} \in \mathbb{R}^{n \times n}$ and $\boldsymbol{U} \in \mathbb{R}^{n \times m}$ are upper and lower triangular matrices respectively. The Eigen C++ library exploits the fact that

$$\mathcal{N}(\boldsymbol{A}) = \boldsymbol{Q}\mathcal{N}(\boldsymbol{U}) \tag{232}$$

A base of the nullspace of the matrix $\boldsymbol{A}$ can hence be obtained efficiently by computing the nullspace of the upper-triangular matrix $\boldsymbol{U}$ and pre-multiply it by the orthogonal matrix $\boldsymbol{Q}$.

B.2. Schur complement

For $\boldsymbol{x} \in \mathbb{R}^{n \times n}$, consider the linear system

$$\underbrace{\begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{bmatrix}}_{\boldsymbol{A}} \underbrace{\begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{bmatrix}}_{\boldsymbol{x}} = \underbrace{\begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \end{bmatrix}}_{\boldsymbol{b}} \tag{233}$$

Resolving the linear system $\boldsymbol{Ax} = \boldsymbol{b}$ can be simplified by first eliminating a subset of the variables before resolving a reduced-size linear system for the remaining variables. Assuming that the sub-matrix $\boldsymbol{A}_{22}$ is invertible, one can write

$$\boldsymbol{x}_2 = \boldsymbol{A}_{22}^{-1} \left(\boldsymbol{b}_2 - \boldsymbol{A}_{21}\boldsymbol{x}_1 \right) \tag{234}$$

eventually yielding

$$\underbrace{\left(\boldsymbol{A}_{11} - \boldsymbol{A}_{12}\boldsymbol{A}_{22}^{-1}\boldsymbol{A}_{21} \right)}_{\text{Schur Complement}} \boldsymbol{x}_1 = \boldsymbol{b}_1 - \boldsymbol{A}_{12}\boldsymbol{A}_{22}^{-1}\boldsymbol{b}_2 \tag{235}$$

The term $\boldsymbol{A}/\boldsymbol{A}_{22}$ related to the last block $\boldsymbol{A}_{22}$ is referred to as the *Schur complement*:

$$\boldsymbol{A}/\boldsymbol{A}_{22} = \boldsymbol{A}_{11} - \boldsymbol{A}_{12}\boldsymbol{A}_{22}^{-1}\boldsymbol{A}_{21} \tag{236}$$

By virtue of proposition 16.1 of [363], if $\boldsymbol{A}$ is a symmetric matrix – implying $\boldsymbol{A}_{12} = \boldsymbol{A}_{21}^{\top}$ – then the following properties of the Schur complement are verified:

- $\boldsymbol{A} \succ 0$ iif $\boldsymbol{A}_{22} \succ 0$ and $\boldsymbol{A}_{11} - \boldsymbol{A}_{12}\boldsymbol{A}_{22}^{-1}\boldsymbol{A}_{12}^{\top} \succ 0$
- if $\boldsymbol{A}_{22} \succ 0$, then $\boldsymbol{A} \succeq 0$ iif $\boldsymbol{A}_{11} - \boldsymbol{A}_{12}\boldsymbol{A}_{22}^{-1}\boldsymbol{A}_{12}^{\top} \succeq 0$

Appendix C

Nonlinear Kalman Filters

This appendix chapter presents some of the nonlinear Kalman filter implementations used in this thesis, namely the Extended, Unscented and Central Difference Kalman Filters, as formulated in [76]. The notations used in these different algorithms follow the Matlab conventions, especially regarding the indexing. For a given time epoch k, the vector $\boldsymbol{x}_k$ denotes the state vector, the vector $\boldsymbol{y}_k$ is the observation vector, the vector $\boldsymbol{u}_k$ is the control input vector, the matrix $\boldsymbol{\Sigma}_{\boldsymbol{x}_k\boldsymbol{x}_k}$ is the covariance matrix associated with $\boldsymbol{x}_k$, the matrix $\boldsymbol{\Sigma}_{\boldsymbol{vv}}$ is the process noise covariance matrix and the matrix $\boldsymbol{\Sigma}_{\boldsymbol{ww}}$ is the measurement noise covariance matrix.

C.1. Extended Kalman Filter (EKF)

Algorithm C.1: Extended Kalman Filter (EKF)

```
Data: $\boldsymbol{y}_k \in \mathbb{R}^{n_y}$, $\boldsymbol{u}_k \in \mathbb{R}^{n_x}$, $\boldsymbol{x}_0 \in \mathbb{R}^{n_x}$,
      $\boldsymbol{\Sigma}_{\boldsymbol{x}_0\boldsymbol{x}_0} \in \mathbb{R}^{n_x \times n_x}, \boldsymbol{\Sigma}_{\boldsymbol{vv}} \in \mathbb{R}^{n_v \times n_v}, \boldsymbol{\Sigma}_{\boldsymbol{ww}} \in \mathbb{R}^{n_y \times n_y}$

for $t = 1, \cdots, \infty$ do
    Time Update Step:

    //Predicted state:
    $\boldsymbol{x}_k^+ \longleftarrow \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k)$
    //Jacobian matrices computation:
    $[\boldsymbol{F}, \boldsymbol{G}, \boldsymbol{H}, \boldsymbol{D}]$ = computeJacobianMatrices $(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k, \boldsymbol{x}_k^+)$
    //Covariance of the prediction:
    $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}_k\boldsymbol{x}_k} \longleftarrow \boldsymbol{F}\boldsymbol{\Sigma}_{\boldsymbol{x}_{k-1}\boldsymbol{x}_{k-1}}\boldsymbol{F}^\top + \boldsymbol{G}\boldsymbol{\Sigma}_{\boldsymbol{vv}}\boldsymbol{G}^\top$

    Measurement Update Step:

    //Predicted observation:
    $\boldsymbol{y}_k^+ \longleftarrow \boldsymbol{g}(\boldsymbol{x}_k^+, \boldsymbol{u}_k)$
    //Covariance of predicted observation:
    $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{y}_k\boldsymbol{y}_k} \longleftarrow \boldsymbol{H}\hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}_k\boldsymbol{x}_k}\boldsymbol{H}^\top + \boldsymbol{D}\boldsymbol{\Sigma}_{\boldsymbol{ww}}\boldsymbol{D}^\top;$
    //Covariance of predicted observation and state:
    $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}_k\boldsymbol{y}_k} \longleftarrow \hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}_k\boldsymbol{x}_k}\boldsymbol{H}^\top;$

    Kalman Update Step:

    //Kalman gain:
    $\boldsymbol{K}_k \longleftarrow \hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}_k\boldsymbol{y}_k^+}\left(\hat{\boldsymbol{\Sigma}}_{\boldsymbol{y}_k\boldsymbol{y}_k}\right)^{-1}$
    //State correction:
    $\boldsymbol{x}_k \longleftarrow \boldsymbol{x}_k^+ + \boldsymbol{K}_k\left(\boldsymbol{y}_k - \boldsymbol{y}_k^+\right)$
    //Covariance correction:
    $\boldsymbol{\Sigma}_{\boldsymbol{x}_k\boldsymbol{x}_k} \longleftarrow \hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}_k\boldsymbol{x}_k} - \boldsymbol{K}_k\hat{\boldsymbol{\Sigma}}_{\boldsymbol{y}_k\boldsymbol{y}_k}\boldsymbol{K}_k^\top$
```

C.2. Sigma Point Kalman Filter (SPKF)

Sigma point Kalman filters have in common the so called sigma point transform, for deterministic sampling of the prior estimate, assumed to be normally distributed.

Algorithm C.2: UKF and CDKF sigma-points computation

```
Function computeSigmaPoints($\gamma$,$\hat{\boldsymbol{z}}$, $\boldsymbol{\Sigma}_{\boldsymbol{zz}}$):
    Data: $\gamma \in \mathbb{R}$,$\hat{\boldsymbol{z}} \in \mathbb{R}^n$, $\boldsymbol{\Sigma}_{\boldsymbol{zz}} \in \mathbb{R}^{n\times n}$
    Result: $\boldsymbol{Z} \in \mathbb{R}^{n\times(2n+1)}$

    //The square root of the covariance matrix can be computed in two
     different ways, according to the selection variable
     "sqrtAlgorithm":

    switch sqrtAlgorithm do
        case cholesky decomposition do
            $\sqrt{\boldsymbol{\Sigma}_{\boldsymbol{zz}}} \longleftarrow \mathtt{chol}(\boldsymbol{\Sigma}_{\boldsymbol{zz}})^\top \in \mathbb{R}^{n\times n}$ //In case the covariance matrix is
             known to be positive-definite.  Very fast.

        case singular value decomposition do
            $[\boldsymbol{U}, \boldsymbol{\sigma}, \boldsymbol{V}] \longleftarrow \mathtt{svd}(\boldsymbol{\Sigma}_{\boldsymbol{zz}})^\top$      //General case where the covariance
             matrix may not be positive-definite.  Rather slow but
             robust.

            $\sqrt{\boldsymbol{\Sigma}_{\boldsymbol{zz}}} \longleftarrow \boldsymbol{U}\sqrt{\boldsymbol{\sigma}}\boldsymbol{V}^\top \in \mathbb{R}^{n\times n}$

    $\boldsymbol{Z} \longleftarrow \left[\hat{\boldsymbol{z}}, \hat{\boldsymbol{z}} + \gamma\sqrt{\boldsymbol{\Sigma}_{\boldsymbol{zz}}}, \hat{\boldsymbol{z}} - \gamma\sqrt{\boldsymbol{\Sigma}_{\boldsymbol{zz}}}\right]$      //Build the sigma-points matrix.
```

C.2.1. Unscented Kalman Filter (UKF)

The Unscented Kalman Filter (UKF) has a set of three tuning scalar coefficients, labeled as α, β and κ. $\boldsymbol{X}_{k_i}$ (resp. $\boldsymbol{Y}_{k_i}$) denote the i^{th} column of the sigma point matrix $\boldsymbol{X}_k$ (resp. $\boldsymbol{Y}_k$) at epoch k.

Algorithm C.3: Unscented Kalman Filter (UKF) Initialization Routine

```
Data: $\alpha \in \mathbb{R}, \beta \in \mathbb{R}, \kappa \in \mathbb{R}$
Result: $n$, $\lambda$, $\gamma$, $w_m$, $w_i$, $w_c$
Function init($\alpha, \beta, \kappa$):
    $n \longleftarrow n_x + n_v + n_y$                    //Dimension of the augmented system
    $\lambda \longleftarrow \alpha^2 (n + \kappa) - n$                          //UKF Weight
    $\gamma \longleftarrow \sqrt{n + \lambda}$                                  //UKF Weight
    $w_m \longleftarrow \frac{\lambda}{n+\lambda}$                                   //UKF Weight
    $w_i \longleftarrow \frac{1}{2(n+\lambda)}$                                  //UKF Weight
    $w_c \longleftarrow (1 - \alpha^2 + \beta) + w_m$                          //UKF Weight
```

Algorithm C.4: Unscented Kalman Filter (UKF) Update Cycle

```
Data: $\boldsymbol{y}_k \in \mathbb{R}^{n_y}$, $\boldsymbol{u}_k \in \mathbb{R}^{n_x}$, $\boldsymbol{x}_0 \in \mathbb{R}^{n_x}$,
      $\boldsymbol{\Sigma}_{\boldsymbol{x}_0\boldsymbol{x}_0} \in \mathbb{R}^{n_x \times n_x}, \boldsymbol{\Sigma}_{\boldsymbol{vv}} \in \mathbb{R}^{n_v \times n_v}, \boldsymbol{\Sigma}_{\boldsymbol{ww}} \in \mathbb{R}^{n_y \times n_y}, \alpha \in \mathbb{R}, \beta \in \mathbb{R}, \kappa \in \mathbb{R}$

$[\lambda, \gamma, w_m, w_i, w_c]$ = init $(\alpha, \beta, \kappa)$

for $t = 1, \cdots, \infty$ do
    Sigma-Points Computation:
    //Augmented state
    $\boldsymbol{z}_{k-1} \longleftarrow \left[\boldsymbol{x}_{k-1}^\top, \boldsymbol{0}_{n_v}^\top, \boldsymbol{0}_{n_y}^\top\right]^\top$
    //Augmented covariance
    $\boldsymbol{\Sigma}_{\boldsymbol{z}_{k-1}\boldsymbol{z}_{k-1}} \longleftarrow \mathrm{blockDiag}(\boldsymbol{\Sigma}_{\boldsymbol{x}_{k-1}\boldsymbol{x}_{k-1}}, \boldsymbol{\Sigma}_{\boldsymbol{vv}}, \boldsymbol{\Sigma}_{\boldsymbol{ww}})$
    //Sigma-points
    $\boldsymbol{Z}_{k-1} \longleftarrow \texttt{computeSigmaPoints}(\gamma, \boldsymbol{z}_{k-1}, \boldsymbol{\Sigma}_{\boldsymbol{z_{k-1}z_{k-1}}})$
    Time Update Step:
    //Sigma-points propagation:
    $\boldsymbol{X}_k \longleftarrow \boldsymbol{f}(\boldsymbol{Z}_{k-1}, \boldsymbol{u}_k)$
    //Predicted state:
    $\boldsymbol{x}_k^+ \longleftarrow w_m \boldsymbol{X}_{k_0} + \sum_{i=1}^{2n} w_i \boldsymbol{X}_{k_i}$
    //Covariance of the predicted state:
    $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}_k\boldsymbol{x}_k} \longleftarrow$
        $w_c \left(\boldsymbol{X}_{k_0} - \boldsymbol{x}_k^+\right) \left(\boldsymbol{X}_{k_0} - \boldsymbol{x}_k^+\right)^\top + \sum_{i=1}^{2n} w_i \left(\boldsymbol{X}_{k_i} - \boldsymbol{x}_k^+\right) \left(\boldsymbol{X}_{k_i} - \boldsymbol{x}_k^+\right)^\top + \boldsymbol{\Sigma}_{\boldsymbol{vv}}$
    Measurement Update Step:
    //Sigma-points propagation:
    $\boldsymbol{Y}_k \longleftarrow \boldsymbol{g}(\boldsymbol{X}_k, \boldsymbol{Z}_{k-1})$
    //Predicted observation:
    $\boldsymbol{y}_k^+ \longleftarrow w_m \boldsymbol{Y}_{k_0} + \sum_{i=1}^{2n} w_i \boldsymbol{Y}_{k_i}$
    //Covariance of the predicted observation:
    $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{y}_k\boldsymbol{y}_k} \longleftarrow$
        $w_c \left(\boldsymbol{Y}_{k_0} - \boldsymbol{y}_k^+\right) \left(\boldsymbol{Y}_{k_0} - \boldsymbol{y}_k^+\right)^\top + \sum_{i=1}^{2n} w_i \left(\boldsymbol{Y}_{k_i} - \boldsymbol{y}_k^+\right) \left(\boldsymbol{Y}_{k_i} - \boldsymbol{y}_k^+\right)^\top + \boldsymbol{\Sigma}_{\boldsymbol{ww}}$
    //Covariance of the predicted observation and state:
    $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}_k\boldsymbol{y}_k} \longleftarrow w_c \left(\boldsymbol{X}_{k_0} - \boldsymbol{x}_k^+\right) \left(\boldsymbol{Y}_{k_0} - \boldsymbol{y}_k^+\right)^\top + \sum_{i=1}^{2n} w_i \left(\boldsymbol{X}_{k_i} - \boldsymbol{x}_k^+\right) \left(\boldsymbol{Y}_{k_i} - \boldsymbol{y}_k^+\right)^\top$
    Kalman Update Step:
    //Kalman gain:
    $\boldsymbol{K}_k \longleftarrow \hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}_k\boldsymbol{y}_k^+} \left(\hat{\boldsymbol{\Sigma}}_{\boldsymbol{y}_k\boldsymbol{y}_k}\right)^{-1}$
    //State correction:
    $\boldsymbol{x}_k \longleftarrow \boldsymbol{x}_k^+ + \boldsymbol{K}_k \left(\boldsymbol{y}_k - \boldsymbol{y}_k^+\right)$
    //Covariance correction:
    $\boldsymbol{\Sigma}_{\boldsymbol{x}_k\boldsymbol{x}_k} \longleftarrow \hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}_k\boldsymbol{x}_k} - \boldsymbol{K}_k \hat{\boldsymbol{\Sigma}}_{\boldsymbol{y}_k\boldsymbol{y}_k} \boldsymbol{K}_k^\top$
```

C.2.2. Central Difference Kalman Filter (CDKF)

The Central Difference Kalman Filter (CDKF) has a single scalar tuning coefficient, referred to as the central difference step size, here labeled as h. As previously, $\boldsymbol{X}_{k_i}$ (resp. $\boldsymbol{Y}_{k_i}$) denote the i^{th} column of the sigma point matrix $\boldsymbol{X}_k$ (resp. $\boldsymbol{Y}_k$) at epoch k. Unlike the UKF, where an augmented system of the state, process and measurement noise variables is considered, the CDKF makes use of two distinct augmented state representations for time update and measurement update, namely state with process noise for time update and predicted state with measurement noise for the measurement update.

Algorithm C.5: Central Difference Kalman Filter (CDKF) Initialization Routine

```
Data: $h \in \mathbb{R}$
Result: $n$, $w_v$, $w_n$, $w_i$, $w_{c1}$, $w_{c2}$

Function init($h$):
    $n \longleftarrow n_x + n_v + n_y$             //Dimension of the augmented system
    $w_v \longleftarrow \frac{h^2 - n_x - n_v}{h^2}$                          //CDKF Weight
    $w_n \longleftarrow \frac{h^2 - n_x - n_y}{h^2}$                          //CDKF Weight
    $w_i \longleftarrow \frac{1}{2h^2}$                                       //CDKF Weight
    $w_{c1} \longleftarrow \frac{1}{4h^2}$                                    //CDKF Weight
    $w_{c2} \longleftarrow \frac{h^2 - 1}{4h^4}$                              //CDKF Weight
```

Algorithm C.6: Central Difference Kalman Filter (CDKF) Update Cycle

```
Data: $\boldsymbol{y}_k \in \mathbb{R}^{n_y}$, $\boldsymbol{u}_k \in \mathbb{R}^{n_x}$, $\boldsymbol{x}_0 \in \mathbb{R}^{n_x}$, $\boldsymbol{\Sigma}_{\boldsymbol{x}_0\boldsymbol{x}_0} \in \mathbb{R}^{n_x \times n_x}$, $\boldsymbol{\Sigma}_{\boldsymbol{vv}} \in \mathbb{R}^{n_v \times n_v}$,
      $\boldsymbol{\Sigma}_{\boldsymbol{ww}} \in \mathbb{R}^{n_y \times n_y}$, $h \in \mathbb{R}$
$[w_v, w_n, w_i, w_{c1}, w_{c2}]$ = init $(h)$
for $\underline{t = 1, \cdots, \infty}$ do
    Sigma-Points Computation:
    //Augmented state and covariance:
    $\boldsymbol{z}_{k-1} \longleftarrow \left[\boldsymbol{x}_{k-1}^\top, \boldsymbol{0}_{n_v}^\top\right]^\top$, $\boldsymbol{\Sigma}_{\boldsymbol{z}_{k-1}\boldsymbol{z}_{k-1}} \longleftarrow \text{blockDiag}(\boldsymbol{\Sigma}_{\boldsymbol{x}_{k-1}\boldsymbol{x}_{k-1}}, \boldsymbol{\Sigma}_{\boldsymbol{vv}})$
    //Sigma-points:
    $\boldsymbol{Z}_{k-1} \longleftarrow \texttt{computeSigmaPoints}(\gamma, \boldsymbol{z}_{k-1}, \boldsymbol{\Sigma}_{\boldsymbol{z_{k-1}z_{k-1}}})$
    Time Update Step:
    //Sigma-points propagation:
    $\boldsymbol{X}_k \longleftarrow \boldsymbol{f}(\boldsymbol{Z}_{k-1}, \boldsymbol{u}_k)$ //Predicted state:
    $\boldsymbol{x}_k^+ \longleftarrow w_v \boldsymbol{X}_{k_0} + \sum_{i=1}^{n_x+n_v} w_i \boldsymbol{X}_{k_i}$
    //Covariance of the predicted state:
    $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}_k\boldsymbol{x}_k} \longleftarrow \sum_{i=1}^{n_x+n_v} \Big[w_{c1}\left(\boldsymbol{X}_{k_i} - \boldsymbol{X}_{k_{n_x+n_v+i}}\right)\left(\boldsymbol{X}_{k_i} - \boldsymbol{X}_{k_{n_x+n_v+i}}\right)^\top + \cdots$
        $\cdots + w_{c2}\left(\boldsymbol{X}_{k_i} + \boldsymbol{X}_{k_{n_x+n_v+i}} - 2\boldsymbol{X}_{k_0}\right)\left(\boldsymbol{X}_{k_i} + \boldsymbol{X}_{k_{n_x+n_v+i}} - 2\boldsymbol{X}_{k_0}\right)^\top\Big]$

    Sigma-Points Computation:
    //Augmented state and covariance:
    $\boldsymbol{s}_{k-1} \longleftarrow \left[\boldsymbol{x}_k^{+\top}, \boldsymbol{0}_{n_y}^\top\right]^\top$, $\boldsymbol{\Sigma}_{\boldsymbol{s}_{k-1}\boldsymbol{s}_{k-1}} \longleftarrow \text{blockDiag}(\hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}_k\boldsymbol{x}_k}, \boldsymbol{\Sigma}_{\boldsymbol{ww}})$
    //Sigma-points:
    $\boldsymbol{S}_{k-1} \longleftarrow \texttt{computeSigmaPoints}(\gamma, \boldsymbol{s}_{k-1}, \boldsymbol{\Sigma}_{\boldsymbol{s_{k-1}s_{k-1}}})$
    Measurement Update Step:
    //Sigma-points propagation:
    $\boldsymbol{Y}_k \longleftarrow \boldsymbol{g}(\boldsymbol{S}_{k-1}, \boldsymbol{u}_k)$ //Predicted observation:
    $\boldsymbol{y}_k^+ \longleftarrow w_n \boldsymbol{Y}_{k_0} + \sum_{i=1}^{n_x+n_y} w_i \boldsymbol{Y}_{k_i}$
    //Covariance of the predicted observation:
    $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{y}_k\boldsymbol{y}_k} \longleftarrow \sum_{i=1}^{2n} \Big[w_{c1}\Big(\boldsymbol{Y}_{k_i} - \boldsymbol{Y}_{k_{n_x+n_y+i}}\Big)\Big(\boldsymbol{Y}_{k_i} - \boldsymbol{Y}_{k_{n_x+n_y+i}}\Big)^\top + \cdots$
        $\cdots + w_{c2}\Big(\boldsymbol{Y}_{k_i} + \boldsymbol{Y}_{k_{n_x+n_y+i}} - 2\boldsymbol{Y}_{k_0}\Big)\Big(\boldsymbol{Y}_{k_i} + \boldsymbol{Y}_{k_{n_x+n_y+i}} - 2\boldsymbol{Y}_{k_0}\Big)^\top\Big]$

    //Covariance of predicted observation and state:
    $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}_k\boldsymbol{y}_k} \longleftarrow \sqrt{w_{c1}\hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}_k\boldsymbol{x}_k}}\,(\boldsymbol{Y}_{k_i} - \boldsymbol{Y}_{k_i})^\top$
    Kalman Update Step:
    //Kalman gain:
    $\boldsymbol{K}_k \longleftarrow \hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}_k\boldsymbol{y}_k^+}\left(\hat{\boldsymbol{\Sigma}}_{\boldsymbol{y}_k\boldsymbol{y}_k}\right)^{-1}$
    //State correction:
    $\boldsymbol{x}_k \longleftarrow \boldsymbol{x}_k^+ + \boldsymbol{K}_k\left(\boldsymbol{y}_k - \boldsymbol{y}_k^+\right)$
    //Covariance correction:
    $\boldsymbol{\Sigma}_{\boldsymbol{x}_k\boldsymbol{x}_k} \longleftarrow \hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}_k\boldsymbol{x}_k} - \boldsymbol{K}_k\hat{\boldsymbol{\Sigma}}_{\boldsymbol{y}_k\boldsymbol{y}_k}\boldsymbol{K}_k^\top$
```

Appendix D

Data availability

D.1. BIRDy Code, Experiment Data and Results

The complete benchmark code as well as the raw experiment data are made available to the reader at the following addresses (accessed on the 06^{th} of October 2021):

- BIRDy source code: `https://github.com/TUM-ICS/BIRDy`
- Experimental data: `http://doi.org/10.5281/zenodo.4728085`
- Experimental results: `http://doi.org/10.5281/zenodo.4679467`

Appendix E

Copyright Permissions

E.1. MDPI Copyright and Licensing

As stated in `https://www.mdpi.com/authors/rights` (accessed on the 06^{th} of October 2021):

"For all articles published in MDPI journals, copyright is retained by the authors. Articles are licensed under an open access Creative Commons CC BY 4.0 license, meaning that anyone may download and read the paper for free. In addition, the article may be reused and quoted provided that the original published version is cited. These conditions allow for maximum use and exposure of the work, while ensuring that the authors receive proper credit.

In exceptional circumstances articles may be licensed differently. If you have specific condition (such as one linked to funding) that does not allow this license, please mention this to the editorial office of the journal at submission. Exceptions will be granted at the discretion of the publisher."

E.2. IEEE Copyright and Licensing

See the attached documents.

Tactile-based compliance with hierarchical force propagation for omnidirectional mobile manipulators

Conference Proceedings:
2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)

Author: Quentin Leboutet

Publisher: IEEE

Date: Nov. 2016

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE WINDOW

Second-order Kinematics for Floating-base Robots using the Redundant Acceleration Feedback of an Artificial Sensory Skin

IEEE Requesting permission to reuse content from an IEEE publication

Conference Proceedings:
2020 IEEE International Conference on Robotics and Automation (ICRA)

Author: Quentin Leboutet

Publisher: IEEE

Date: May 2020

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK | CLOSE WINDOW

Online Configuration Selection for Redundant Arrays of Inertial Sensors: Application to Robotic Systems Covered with a Multimodal Artificial Skin

Conference Proceedings:
2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)

Author: Quentin Leboutet

Publisher: IEEE

Date: 24 Oct. 2020

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE WINDOW

Tactile-Based Whole-Body Compliance With Force Propagation for Mobile Manipulators

Author: Quentin Leboutet

Publication: IEEE Transactions on Robotics

Publisher: IEEE

Date: April 2019

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE WINDOW

Bibliography

[1] Stefan Schaal. The new robotics towards human-centered machines. HFSP journal, 1(2):115–126, 2007.

[2] Patrick Lin, Keith Abney, and George Bekey. Robot ethics: Mapping the issues for a mechanized world. Artificial Intelligence, 175(5):942–949, 2011.

[3] Maartje MA de Graaf and Somaya Ben Allouch. Anticipating our future robot society: The evaluation of future robot applications from a user's perspective. In Robot and Human Interactive Communication (RO-MAN), 2016 25th IEEE International Symposium on, pages 755–762. IEEE, 2016.

[4] Rachid Alami, Alin Albu-Schäffer, Antonio Bicchi, Rainer Bischoff, Raja Chatila, Alessandro De Luca, Agostino De Santis, Georges Giralt, Jérémie Guiochet, Gerd Hirzinger, et al. Safe and dependable physical human-robot interaction in anthropic domains: State of the art and challenges. In Intelligent Robots and Systems (IROS), 2006 IEEE/RSJ International Conference on, pages 1–16. IEEE, 2006.

[5] Quentin Leboutet, Emmanuel Dean-Leon, Florian Bergner, and Gordon Cheng. Tactile-based whole-body compliance with force propagation for mobile manipulators. IEEE Transactions on Robotics, 35(2):330–342, 2019.

[6] G Cheng, E Dean-Leon, F Bergner, JR Guadarrama-Olvera, Q Leboutet, and P Mittendorfer. A comprehensive realisation of robot skin: Sensors, sensing, control and applications. Proceedings of the IEEE, 2019.

[7] Michael Perlmutter and Laurent Robin. High-performance, low cost inertial mems: A market in motion! In Proceedings of the 2012 IEEE/ION Position, Location and Navigation Symposium, pages 225–229. IEEE, 2012.

[8] Philipp Mittendorfer and Gordon Cheng. Integrating discrete force cells into multi-modal artificial skin. In Humanoid Robots (Humanoids), 2012 12th IEEE-RAS International Conference on, pages 847–852. IEEE, 2012.

[9] Mark W Spong and Mathukumalli Vidyasagar. Robot dynamics and control. John Wiley & Sons, 2008.

[10] Wisama Khalil and Etienne Dombre. Modeling, identification and control of robots. Butterworth-Heinemann, 2004.

[11] Maxime Gautier. Dynamic identification of robots with power model. In Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on, volume 3, pages 1922–1927. IEEE, 1997.

[12] Ror Bellman and Karl Johan Åström. On structural identifiability. Mathematical biosciences, 7(3-4):329–339, 1970.

[13] Maxime Gautier and Wisama Khalil. Exciting trajectories for the identification of base inertial parameters of robots. The International journal of robotics research, 11(4):362–375, 1992.

[14] G Calafiore, Marina Indri, and Basilio Bona. Robot dynamic calibration: Optimal excitation trajectories and experimental parameter estimation. Journal of robotic systems, 18(2):55–68, 2001.

[15] Jingfu Jin and Nicholas Gans. Parameter identification for industrial robots with a fast and robust trajectory design approach. Robotics and Computer-Integrated Manufacturing, 31:21–29, 2015.

[16] Jan Swevers, Chris Ganseman, D Bilgin Tukel, Joris De Schutter, and Hendrik Van Brussel. Optimal robot excitation and identification. IEEE transactions on robotics and automation, 13(5):730–740, 1997.

[17] Jan Swevers, Walter Verdonck, and Joris De Schutter. Dynamic model identification for industrial robots. IEEE Control Systems, 27(5):58–71, 2007.

[18] Fares J Abu-Dakka and Miguel Díaz-Rodríguez. Comparison of trajectory parametrization methods with statistical analysis for dynamic parameter identification of serial robot. In Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on, pages 5874–5879. IEEE, 2017.

[19] Ko Ayusawa, Antoine Rioux, Eiichi Yoshida, Gentiane Venture, and Maxime Gautier. Generating persistently exciting trajectory based on condition number optimization. In 2017 IEEE International Conference on Robotics and Automation (ICRA), pages 6518–6524. IEEE, 2017.

[20] Jovana Jovic, Franck Philipp, Adrien Escande, Ko Ayusawa, Eiichi Yoshida, Abderrahmane Kheddar, and Gentiane Venture. Identification of dynamics of humanoids: Systematic exciting motion generation. In 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 2173–2179. IEEE, 2015.

[21] Yong Han, Jianhua Wu, Chao Liu, and Zhenhua Xiong. An iterative approach for accurate dynamic model identification of industrial robots. IEEE Transactions on Robotics, 2020.

[22] Maxime Gautier and Wisama Khalil. Direct calculation of minimum set of inertial parameters of serial robots. IEEE Transactions on robotics and Automation, 6(3):368–373, 1990.

[23] Hirokazu Mayeda, Koji Yoshida, and Koichi Osuka. Base parameters of manipulator dynamic models. IEEE Transactions on Robotics and Automation, 6(3):312–321, 1990.

[24] Manuel Beschi, Enrico Villagrossi, Nicola Pedrocchi, and Lorenzo Molinari Tosatti. A general analytical procedure for robot dynamic model reduction. In 2015 IEEE/RSJ

International Conference on Intelligent Robots and Systems (IROS), pages 4127–4132. IEEE, 2015.

[25] Cristóvão D Sousa and Rui Cortesão. Physical feasibility of robot base inertial parameter identification: A linear matrix inequality approach. The International Journal of Robotics Research, 33(6):931–944, 2014.

[26] Maxime Gautier, Sébastien Briot, and Gentiane Venture. Identification of consistent standard dynamic parameters of industrial robots. In 2013 IEEE/ASME International Conference on Advanced Intelligent Mechatronics, pages 1429–1435. IEEE, 2013.

[27] Maxime Gautier, Alexandre Janot, and Pierre-Olivier Vandanjon. A new closed-loop output error method for parameter identification of robot dynamics. IEEE Transactions on Control Systems Technology, 21(2):428–444, 2013.

[28] Mathieu Brunot, Alexandre Janot, Francisco Carrillo, Joono Cheong, and Noel Jean-Philippe. Output error methods for robot identification. IEEE/ASME Transactions on Mechatronics, X(X):XX–XX, 2019.

[29] Alexandre Janot, Pierre-Olivier Vandanjon, and Maxime Gautier. A generic instrumental variable approach for industrial robot identification. IEEE Transactions on Control Systems Technology, 22(1):132–145, 2014.

[30] Anthony Jubien, Maxime Gautier, and Alexandre Janot. Dynamic identification of the kuka lwr robot using motor torques and joint torque sensors data. IFAC Proceedings Volumes, 47(3):8391–8396, 2014.

[31] G. Venture, P. . Ripert, W. Khalil, M. Gautier, and P. Bodson. Modeling and identification of passenger car dynamics using robotics formalism. IEEE Transactions on Intelligent Transportation Systems, 7(3):349–359, Sep. 2006.

[32] F. Khatounian, S. Moreau, E. Monmasson, A. Janot, and F. Louveau. Parameters estimation of the actuator used in haptic interfaces: Comparison of two identification methods. In 2006 IEEE International Symposium on Industrial Electronics, volume 1, pages 211–216, July 2006.

[33] Charles-Eric Lemaire, Pierre-Olivier Vandanjon, Maxime Gautier, and Charles Lemaire. Dynamic identification of a vibratory asphalt compactor for contact efforts estimation. IFAC Proceedings Volumes, 39(1):973–978, 2006.

[34] Peter J Huber et al. The 1972 wald lecture robust statistics: A review. The Annals of Mathematical Statistics, 43(4):1041–1067, 1972.

[35] Peter J Huber. Robust regression: asymptotics, conjectures and monte carlo. The Annals of Mathematical Statistics, 1(5):799–821, 1973.

[36] Paul W Holland and Roy E Welsch. Robust regression using iteratively reweighted least-squares. Communications in Statistics-theory and Methods, 6(9):813–827, 1977.

[37] Alexandre Janot, Pierre-Olivier Vandanjon, and Maxime Gautier. Using robust regressions and residual analysis to verify the reliability of ls estimation: Application in robotics. In 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 1962–1967. IEEE, 2009.

[38] Maxime Gautier and Ph Poignet. Extended kalman filtering and weighted least squares dynamic identification of robot. Control Engineering Practice, 9(12):1361–1372, 2001.

[39] Mathieu Brunot, Alexandre Janot, Peter C Young, and Francisco Carrillo. An improved instrumental variable method for industrial robot model identification. Control Engineering Practice, 74:107–117, 2018.

[40] Martin M. Olsen, Jan Swevers, and Walter Verdonck. Maximum Likelihood Identification of a Dynamic Robot Model: Implementation Issues. The International Journal of Robotics Research, 21(2):89–96, 2002.

[41] F. Xi. Effect of non-geometric errors on manipulator inertial calibration. In Proceedings of 1995 IEEE International Conference on Robotics and Automation, volume 2, pages 1808–1813 vol.2, May 1995.

[42] Ake Bjorck. Numerical methods for least squares problems, volume 51. Siam, 1996.

[43] Sabine Van Huffel and Joos Vandewalle. The total least squares problem: computational aspects and analysis, volume 9. Siam, 1991.

[44] Stephan Rhode, Konstantin Usevich, Ivan Markovsky, and Frank Gauterin. A recursive restricted total least-squares algorithm. IEEE Transactions on Signal Processing, 62(21):5652–5662, 2014.

[45] S. Van Huffel and J. . Vandewalle. Comparison of total least squares and instrumental variable methods for parameter estimation of transfer function models. International Journal of Control, 50(4):1039–1056, April 1989.

[46] Torsten Söderström and K. Mahata. On instrumental variable and total least squares approaches for identification of noisy systems. International Journal of Control, 75(6):381–389, 2002.

[47] Alexandre Janot. On the identification of continuous-time inverse dynamic model of electromechanical systems operating in closed loop with an instrumental variable approach: application to industrial robots. PhD thesis, CENTRE MIDI PYRENEES TOULOUSE, 2017.

[48] Alexandre Janot, Pierre-Olivier Vandanjon, and Maxime Gautier. A revised durbin-wu-hausman test for industrial robot identification. Control Engineering Practice, 48:52–62, 2016.

[49] S. Van Huffel and J. . Vandewalle. Analysis and properties of the generalized total least squares problem ax=b when some or all columns in a are subject to error. SIAM Journal on Matrix Analysis and Applications, 10(3):294 – 315, July 1989.

[50] Maxime Gautier and Sebastien Briot. Global identification of joint drive gains and dynamic parameters of robots. J. Dyn. Sys., Meas., Control, 136(5):1–9, 2014.

[51] Ivan Markovsky and Sabine Van Huffel. Overview of total least-squares methods. Signal Processing, 87(10):2283 – 2302, 2007.

[52] Olav Reiersøl. Confluence analysis by means of lag moments and other methods of confluence analysis. Econometrica: Journal of the Econometric Society, pages 1–24, 1941.

[53] Torsten Söderström and Petre Stoica. Instrumental variable methods for system identification. Circuits, Systems and Signal Processing, 21(1):1–9, 2002.

[54] Peter.C. Young. Recursive Estimation and Time-Series Analysis: An Introduction for the Student and Practitioner. Springer Verlag Berlin, 2011.

[55] Sarat C Puthenpura and Naresh K Sinha. Identification of continuous-time systems using instrumental variables with application to an industrial robot. IEEE Transactions on Industrial Electronics, IE-33(3):224–229, 1986.

[56] Koji Yoshida, Nobuya Ikeda, and Hirokazu Mayeda. Experimental study of the identification methods for an industrial robot manipulator. In Intelligent Robots and Systems, 1992., Proceedings of the 1992 IEEE/RSJ International Conference on, volume 1, pages 263–270. IEEE, 1992.

[57] Mathieu Brunot. Identification of rigid industrial robots-A system identification perspective. PhD thesis, Toulouse, INPT, 2017.

[58] M Brunot, A Janot, and F Carrillo. An automated instrumental variable method for rigid industrial robot identification. IFAC-PapersOnLine, 51(15):431–436, 2018.

[59] A Janot. A separable instrumental variable method for robot identification. In 2021 American Control Conference (ACC), pages 4339–4344. IEEE, 2021.

[60] Martin M Olsen and Henrik Gordon Petersen. A new method for estimating parameters of a dynamic robot model. IEEE transactions on robotics and automation, 17(1):95–100, 2001.

[61] K. Dolinský and S. Čelikovský. Application of the method of maximum likelihood to identification of bipedal walking robots. IEEE Transactions on Control Systems Technology, 26(4):1500–1507, 2018.

[62] Peter. C. Young. Refined instrumental variable estimation: maximum likelihood optimization of a unified box–jenkins model. Automatica, 52:35–46, 2014.

[63] Ioan Doré Landau and Alireza Karimi. An output error recursive algorithm for unbiased identification in closed loop. Automatica, 33(5):933–938, 1997.

[64] ID Landau, BDO Anderson, and F De Bruyne. Closed-loop output error identification algorithms for nonlinear plants. In Decision and Control, 1999. Proceedings of the 38th IEEE Conference on, volume 1, pages 606–611. IEEE, 1999.

[65] P Aivaliotis, E Papalitsa, G Michalos, and S Makris. Identification of dynamic robot's parameters using physics-based simulation models for improving accuracy. Procedia CIRP, 96:254–259, 2021.

[66] Alexandre Janot, Maxime Gautier, Anthony Jubien, and Pierre Olivier Vandanjon. Comparison between the cloe method and the didim method for robots identification. IEEE Transactions on Control Systems Technology, 22(5):1935–1941, 2014.

[67] Maxime Gautier, Pierre-Olivier Vandanjon, and Alexandre Janot. Dynamic identification of a 6 dof robot without joint position data. In 2011 IEEE International Conference on Robotics and Automation, pages 234–239. IEEE, 2011.

[68] M. Brunot, A. Janot, F. Carrillo, and H. Garnier. Comparison between the idim-iv method and the didim method for industrial robots identification. In 2017 IEEE International Conference on Advanced Intelligent Mechatronics (AIM), pages 571–576, July 2017.

[69] Eric A Wan, Rudolph Van Der Merwe, and Alex T Nelson. Dual estimation and the unscented transformation. In Advances in neural information processing systems, pages 666–672, 2000.

[70] Anindya S Paul and Eric A Wan. Dual kalman filters for autonomous terrain aided navigation in unknown environments. In Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on, volume 5, pages 2784–2789. IEEE, 2005.

[71] Vladimir Joukov, Vincent Bonnet, Gentiane Venture, and Dana Kulić. Constrained dynamic parameter estimation using the extended kalman filter. In Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on, pages 3654–3659. IEEE, 2015.

[72] Claudio Urrea and José Pascal. Parameter identification methods for real redundant manipulators. Journal of Applied Research and Technology, 15(4):320–331, 2017.

[73] Brian Dellon and Yoky Matsuoka. Modeling and system identification of a life-size brake-actuated manipulator. IEEE Transactions on Robotics, 25(3):481–491, 2009.

[74] Vincent Bonnet, Gauthier Daune, Vladimir Joukov, Raphaël Dumas, Philippe Fraisse, D Kulić, Antoine Seilles, Sebastien Andary, and Gentiane Venture. A constrained extended kalman filter for dynamically consistent inverse kinematics and inertial parameters identification. In 2016 6th IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob), pages 944–949. IEEE, 2016.

[75] Mauro Hernán Riva, Mark Wielitzka, and Tobias Ortmaier. Sensitivity-based adaptive srukf for state, parameter, and covariance estimation on mechatronic systems. Kalman Filters: Theory for Advanced Applications, page 77, 2018.

[76] Rudolph Van Der Merwe. Sigma-point Kalman filters for probabilistic inference in dynamic state-space models. PhD thesis, Oregon Health & Science University, 2004.

[77] Roy Featherstone. Rigid body dynamics algorithms. Springer, 2014.

[78] Jane Liu and Mike West. Combined parameter and state estimation in simulation-based filtering. In Sequential Monte Carlo methods in practice, pages 197–223. Springer, 2001.

[79] M Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. IEEE Transactions on signal processing, 50(2):174–188, 2002.

[80] Léon Bottou. Stochastic gradient descent tricks. In Neural networks: Tricks of the trade, pages 421–436. Springer, 2012.

[81] Claudio Urrea and José Pascal. Design, simulation, comparison and evaluation of parameter identification methods for an industrial robot. Computers & Electrical Engineering, 2016.

[82] John J Hopfield and David W Tank. “neural” computation of decisions in optimization problems. Biological cybernetics, 52(3):141–152, 1985.

[83] Miguel Atencia, Gonzalo Joya, and Francisco Sandoval. Hopfield neural networks for parametric identification of dynamical systems. Neural Processing Letters, 21(2):143–152, 2005.

[84] Abe. Theories on the hopfield neural networks. In International 1989 Joint Conference on Neural Networks, pages 557–564 vol.1, 1989.

[85] Kate Smith, Marimuthu Palaniswami, and Mohan Krishnamoorthy. Neural techniques for combinatorial optimization with applications. IEEE Transactions on Neural Networks, 9(6):1301–1318, 1998.

[86] John J Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. Proceedings of the national academy of sciences, 81(10):3088–3092, 1984.

[87] Kumpati S Narendra and Kannan Parthasarathy. Identification and control of dynamical systems using neural networks. IEEE Transactions on neural networks, 1(1):4–27, 1990.

[88] S Reynold Chu, Rahmat Shoureshi, and Manoel Tenorio. Neural networks for system identification. IEEE Control systems magazine, 10(3):31–35, 1990.

[89] JR Raol. Neural network based parameter estimation of unstable aerospace dynamic systems. IEE Proceedings-Control Theory and Applications, 141(6):385–388, 1994.

[90] JR Raol. Parameter estimation of state space models by recurrent neural networks. IEE Proceedings-Control Theory and Applications, 142(2):114–118, 1995.

[91] A Delgado, C Kambhampati, and Kevin Warwick. Dynamic recurrent neural network for system identification and control. IEE Proceedings-Control Theory and Applications, 142(4):307–314, 1995.

[92] Miguel Atencia, Gonzalo Joya, and Francisco Sandoval. Parametric identification of robotic systems with stable time-varying hopfield networks. Neural Computing & Applications, 13(4):270–280, 2004.

[93] Koji Yoshida and Wisama Khalil. Verification of the positive definiteness of the inertial matrix of manipulators using base inertial parameters. The International Journal of Robotics Research, 19(5):498–510, 2000.

[94] Vicente Mata, Francesc Benimeli, Nidal Farhat, and Angel Valera. Dynamic parameter identification in industrial robots considering physical feasibility. Advanced Robotics, 19(1):101–119, 2005.

[95] Jo-Anne Ting, Michael N Mistry, Jan Peters, Stefan Schaal, and Jun Nakanishi. A bayesian approach to nonlinear parameter identification for rigid body dynamics. In Robotics: Science and systems, pages 32–39. Philadelphia, USA, 2006.

[96] Robert M Freund. Introduction to semidefinite programming (sdp). Massachusetts Institute of Technology, pages 8–11, 2004.

[97] Silvio Traversaro, Stanislas Brossette, Adrien Escande, and Francesco Nori. Identification of fully physical consistent inertial parameters using optimization on manifolds. In 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 5446–5451. IEEE, 2016.

[98] Cristóvão D Sousa and Rui Cortesão. Inertia tensor properties in robot dynamics identification: A linear matrix inequality approach. IEEE/ASME Transactions on Mechatronics, 24(1):406–411, Feb 2019.

[99] P. M. Wensing, S. Kim, and J. J. E. Slotine. Linear matrix inequalities for physically consistent inertial parameter identification: A statistical perspective on the mass distribution. IEEE Robotics and Automation Letters, 3(1):60–67, Jan 2018.

[100] Alexandre Janot and Patrick M. Wensing. Sequential semidefinite optimization for physically and statistically consistent robot identification. Control Engineering Practice, 107:104699, 2021.

[101] Ko Ayusawa and Yoshihiko Nakamura. Identification of standard inertial parameters for large-dof robots considering physical consistency. In 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 6194–6201. IEEE, 2010.

[102] Ko Ayusawa, Gentiane Venture, and Yoshihiko Nakamura. Real-time implementation of physically consistent identification of human body segments. In 2011 IEEE International Conference on Robotics and Automation, pages 6282–6287, 2011.

[103] Taeyoon Lee, Patrick M Wensing, and Frank C Park. Geometric robot dynamic identification: A convex programming approach. IEEE Transactions on Robotics, 36(2):348–365, 2019.

[104] Taeyoon Lee and Frank C Park. A geometric algorithm for robust multibody inertial parameter identification. IEEE Robotics and Automation Letters, 3(3):2455–2462, 2018.

[105] M. Gautier and G. Venture. Identification of standard dynamic parameters of robots with positive definite inertia matrix. In 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 5815–5820, 2013.

[106] Jovana Jovic, Adrien Escande, Ko Ayusawa, Eiichi Yoshida, Abderrahmane Kheddar, and Gentiane Venture. Humanoid and human inertia parameter identification using hierarchical optimization. IEEE Transactions on Robotics, 32(3):726–735, 2016.

[107] Michael Mistry, Stefan Schaal, and Katsu Yamane. Inertial parameter estimation of floating base humanoid systems using partial force sensing. In 2009 9th IEEE-RAS International Conference on Humanoid Robots, pages 492–497. IEEE, 2009.

[108] Katsu Yamane. Practical kinematic and dynamic calibration methods for force-controlled humanoid robots. In 2011 11th IEEE-RAS International Conference on Humanoid Robots, pages 269–275. IEEE, 2011.

[109] Ko Ayusawa, Gentiane Venture, and Yoshihiko Nakamura. Identifiability and identification of inertial parameters using the underactuated base-link dynamics for legged multibody systems. The International Journal of Robotics Research, 33(3):446–468, 2014.

[110] Gentiane Venture, Ko Ayusawa, and Yoshihiko Nakamura. Motion capture based identification of the human body inertial parameters. In 2008 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pages 4575–4578. IEEE, 2008.

[111] Randa Mallat, Vincent Bonnet, Samer Mohammed, and Mohamad Khalil. Dynamic identification of a human-exoskeleton system. In 2017 Fourth International Conference on Advances in Biomedical Engineering (ICABME), pages 1–4. IEEE, 2017.

[112] Randa Mallat, Vincent Bonnet, Weiguang Huo, Patrick Karasinski, Yacine Amirat, Mohamad Khalil, and Samer Mohammed. Human-exoskeleton system dynamics identification using affordable sensors. In 2018 IEEE International Conference on Robotics and Automation (ICRA), pages 6759–6765. IEEE, 2018.

[113] Steven Dubowsky and Evangelos Papadopoulos. The kinematics, dynamics, and control of free-flying and free-floating space robotic systems. IEEE Transactions on robotics and automation, 9(5):531–543, 1993.

[114] Quentin Leboutet, J Rogelio Guadarrama-Olvera, Florian Bergner, and Gordon Cheng. Second-order kinematics for floating-base robots using the redundant acceleration feedback of an artificial sensory skin. In 2020 IEEE International Conference on Robotics and Automation (ICRA), pages 4687–4694. IEEE, 2020.

[115] Ko Ayusawa, Shin'ichiro Nakaoka, Eiichi Yoshida, Yumeko Imamura, and Takayuki Tanaka. Evaluation of assistive devices using humanoid robot with mechanical parameters identification. In 2014 IEEE-RAS International Conference on Humanoid Robots, pages 205–211. IEEE, 2014.

[116] Yusuke Ogawa, Gentiane Venture, and Christian Ott. Dynamic parameters identification of a humanoid robot using joint torque sensors and/or contact forces. In 2014 IEEE-RAS International Conference on Humanoid Robots, pages 457–462. IEEE, 2014.

[117] K Ayusawa, G Venture, and Y Nakamura. Inertial parameters identifiability of humanoid robot based on the baselink equation of motion. In Proc. of the Conf. on Robotics and Mechatronics, pages 2P1–F09, 2008.

[118] Vincent Bonnet, André Crosnier, Gentiane Venture, Maxime Gautier, and Philippe Fraisse. Inertial parameters identification of a humanoid robot hanged to a fix force sensor. In 2018 IEEE International Conference on Robotics and Automation (ICRA), pages 4927–4932. IEEE, 2018.

[119] Dana Kulić, Gentiane Venture, Katsu Yamane, Emel Demircan, Ikuo Mizuuchi, and Katja Mombaur. Anthropomorphic movement analysis and synthesis: A survey of methods and applications. IEEE Transactions on Robotics, 32(4):776–795, 2016.

[120] C Presse and Maxime Gautier. New criteria of exciting trajectories for robot identification. In [1993] Proceedings IEEE International Conference on Robotics and Automation, pages 907–912. IEEE, 1993.

[121] Kyung-Jo Park. Fourier-based optimal excitation trajectories for the dynamic identification of robots. Robotica, 24(5):625–633, 2006.

[122] Jun Wu, Jinsong Wang, and Zheng You. An overview of dynamic parameter identification of robots. Robotics and computer-integrated manufacturing, 26(5):414–419, 2010.

[123] Gentiane Venture, Ko Ayusawa, and Yoshihiko Nakamura. A numerical method for choosing motions with optimal excitation properties for identification of biped dynamics-an application to human. In 2009 IEEE International Conference on Robotics and Automation, pages 1226–1231. IEEE, 2009.

[124] Ko Ayusawa, Yoshihiko Nakamura, and Gentiane Venture. Optimal estimation of human body segments dynamics using realtime visual feedback. In 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 1627–1632. IEEE, 2009.

[125] Vincent Bonnet and Gentiane Venture. Fast determination of the planar body segment inertial parameters using affordable sensors. IEEE Transactions on neural systems and rehabilitation engineering, 23(4):628–635, 2015.

[126] Vincent Bonnet, Philippe Fraisse, André Crosnier, Maxime Gautier, Alejandro González, and Gentiane Venture. Optimal exciting dance for identifying inertial parameters of an anthropomorphic structure. IEEE Transactions on Robotics, 32(4):823–836, 2016.

[127] Vincent Bonnet, Kai Pfeiffer, Philippe Fraisse, André Crosnier, and Gentiane Venture. Self-generation of optimal exciting motions for identification of a humanoid robot. International Journal of Humanoid Robotics, 15(06):1850024, 2018.

[128] Mark W Spong, Seth Hutchinson, and Mathukumalli Vidyasagar. Robot modeling and control, volume 3. Wiley New York, 2006.

[129] Joel W Burdick. On the inverse kinematics of redundant manipulators: Characterization of the self-motion manifolds. In Advanced Robotics: 1989, pages 25–34. Springer, 1989.

[130] Adi Ben-Israel and Thomas NE Greville. Generalized inverses: theory and applications, volume 15. Springer Science & Business Media, 2003.

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

[132] Yoshihiko Nakamura and Hideo Hanafusa. Inverse kinematic solutions with singularity robustness for robot manipulator control. Journal of dynamic systems, measurement, and control, 108(3):163–171, 1986.

[133] Alexander Dietrich, Alin Albu-Schäffer, and Gerd Hirzinger. On continuous null space projections for torque-based, hierarchical, multi-objective manipulation. In Robotics and Automation (ICRA), 2012 IEEE International Conference on, pages 2978–2985. IEEE, 2012.

[134] Laura Kelmar and Pradeep K Khosla. Automatic generation of kinematics for a reconfigurable modular manipulator system. In Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on, pages 663–668. IEEE, 1988.

[135] Alain Liegeois. Automatic supervisory control of the configuration and behavior of multibody mechanisms. IEEE transactions on systems, man, and cybernetics, 7(12):868–871, 1977.

[136] Luis Sentis and Oussama Khatib. Control of free-floating humanoid robots through task prioritization. In Proceedings of the 2005 IEEE International Conference on Robotics and Automation, pages 1718–1723. IEEE, 2005.

[137] Alexander Dietrich, Christian Ott, and Alin Albu-Schäffer. An overview of null space projections for redundant, torque-controlled robots. The International Journal of Robotics Research, page 0278364914566516, 2015.

[138] Luis Sentis, Jaeheung Park, and Oussama Khatib. Compliant control of multicontact and center-of-mass behaviors in humanoid robots. IEEE Transactions on robotics, 26(3):483–501, 2010.

[139] Nicolas Mansard and Franois Chaumette. Task sequencing for high-level sensor-based control. IEEE Transactions on Robotics, 23(1):60–72, 2007.

[140] Oussama Kanoun, Florent Lamiraux, and Pierre-Brice Wieber. Kinematic control of redundant manipulators: Generalizing the task-priority framework to inequality task. Robotics, IEEE Transactions on, 27(4):785–792, 2011.

[141] Nicolas Mansard. A dedicated solver for fast operational-space inverse dynamics. In 2012 IEEE International Conference on Robotics and Automation, pages 4943–4949. IEEE, 2012.

[142] Layale Saab, Oscar E Ramos, François Keith, Nicolas Mansard, Philippe Soueres, and J Fourquet. Dynamic whole-body motion generation under rigid contacts and other unilateral constraints. Robotics, IEEE Transactions on, 29(2):346–362, 2013.

[143] Alexander Sherikov. Balance preservation and task prioritization in whole body motion control of humanoid robots. PhD thesis, Université Grenoble Alpes, 2016.

[144] Alexander Herzog, Nicholas Rotella, Sean Mason, Felix Grimminger, Stefan Schaal, and Ludovic Righetti. Momentum control with hierarchical inverse dynamics on a torque-controlled humanoid. Autonomous Robots, 40(3):473–491, 2016.

[145] Karim Bouyarmane and Abderrahmane Kheddar. Using a multi-objective controller to synthesize simulated humanoid robot motion with changing contact configurations. In 2011 IEEE/RSJ international conference on intelligent robots and systems, pages 4414–4419. IEEE, 2011.

[146] Martin De Lasa, Igor Mordatch, and Aaron Hertzmann. Feature-based locomotion controllers. ACM Transactions on Graphics (TOG), 29(4):1–10, 2010.

[147] Karim Bouyarmane, Kevin Chappellet, Joris Vaillant, and Abderrahmane Kheddar. Quadratic programming for multirobot and task-space force control. IEEE Transactions on Robotics, 35(1):64–77, 2018.

[148] Stéphane Caron, Abderrahmane Kheddar, and Olivier Tempier. Stair climbing stabilization of the hrp-4 humanoid robot using whole-body admittance control. In 2019 International Conference on Robotics and Automation (ICRA), pages 277–283. IEEE, 2019.

[149] Pierre Gergondet, Arnaud Tanguy, Hervé Audren, Masaki Murooka, Stéphane Caron, Adrien Escande, Mehdi Benallegue, Fumio Kanehiro, Benjamin Chrétien, Raphael Grimm, Kevin Chappellet, Yuquan Wang, Anastasia Bolotnikova, and Iori Kumagai. mc-rtc control software. `https://jrl-umi3218.github.io/mc_rtc`, 2021. [Online; accessed 6-October-2021].

[150] Joris Vaillant, Abderrahmane Kheddar, Hervé Audren, François Keith, Stanislas Brossette, Adrien Escande, Karim Bouyarmane, Kenji Kaneko, Mitsuharu Morisawa, Pierre Gergondet, et al. Multi-contact vertical ladder climbing with an hrp-2 humanoid. Autonomous Robots, 40(3):561–580, 2016.

[151] Enrico Mingo Hoffman, Alessio Rocchi, Nikos G Tsagarakis, and Darwin G Caldwell. Robot dynamics constraint for inverse kinematics. In Advances in Robot Kinematics 2016, pages 275–283. Springer, 2018.

[152] Ludovic Righetti and Stefan Schaal. Quadratic programming for inverse dynamics with optimal distribution of contact forces. In Humanoid Robots (Humanoids), 2012 12th IEEE-RAS International Conference on, pages 538–543. IEEE, 2012.

[153] Rafael Cisneros, Mehdi Benallegue, Abdelaziz Benallegue, Mitsuharu Morisawa, Hervé Audren, Pierre Gergondet, Adrien Escande, Abderrahmane Kheddar, and Fumio Kanehiro. Robust humanoid control using a qp solver with integral gains. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 7472–7479. IEEE, 2018.

[154] Stéphane Caron, Quang-Cuong Pham, and Yoshihiko Nakamura. Stability of surface contacts for humanoid robots: Closed-form formulae of the contact wrench cone for rectangular support areas. In 2015 IEEE International Conference on Robotics and Automation (ICRA), pages 5107–5112. IEEE, 2015.

[155] Hisashi Sugiura, Michael Gienger, Herbert Janssen, and Christian Goerick. Real-time self collision avoidance for humanoids by means of nullspace criteria and task intervals. In 2006 6th IEEE-RAS International Conference on Humanoid Robots, pages 575–580. IEEE, 2006.

[156] Olivier Stasse, Adrien Escande, Nicolas Mansard, Sylvain Miossec, Paul Evrard, and Abderrahmane Kheddar. Real-time (self)-collision avoidance task on a hrp-2 humanoid robot. In 2008 ieee international conference on robotics and automation, pages 3200–3205. IEEE, 2008.

[157] Adrien Escande, Sylvain Miossec, Mehdi Benallegue, and Abderrahmane Kheddar. A strictly convex hull for computing proximity distances with continuous gradients. IEEE Transactions on Robotics, 30(3):666–678, 2014.

[158] C Bradford Barber, David P Dobkin, and Hannu Huhdanpaa. The quickhull algorithm for convex hulls. ACM Transactions on Mathematical Software (TOMS), 22(4):469–483, 1996.

[159] M. Benallegue, A. Escande, S. Miossec, and A. Kheddar. Fast c1 proximity queries using support mapping of sphere-torus-patches bounding volumes. In Robotics and Automation, 2009. ICRA ’09. IEEE International Conference on, pages 483–488, May 2009.

[160] Adrien Escande, Sylvain Miossec, Mehdi Benallegue, and Abderrahmane Kheddar. A strictly convex hull for computing proximity distances with continuous gradients. Robotics, IEEE Transactions on, 30(3):666–678, June 2014.

[161] Adrien Escande, Nicolas Mansard, and Pierre-Brice Wieber. Hierarchical quadratic programming: Fast online humanoid-robot motion generation. The International Journal of Robotics Research, page 0278364914521306, 2014.

[162] Neville Hogan. Impedance control: An approach to manipulation: Part i-theory, part ii-implementation, part iii-applications. Journal of Dynamic Systems, Measurement, and Control, 107:1–24, 1985.

[163] Hun-Ok Lim and Kazuo Tanie. Human safety mechanisms of human-friendly robots: passive viscoelastic trunk and passively movable base. The International Journal of Robotics Research, 19(4):307–335, 2000.

[164] Antonio Bicchi, Stefano Lodi Rizzini, and Giovanni Tonietti. Compliant design for intrinsic safety: General issues and preliminary design. In Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on, volume 4, pages 1864–1869. IEEE, 2001.

[165] Michael Zinn, Bernard Roth, Oussama Khatib, and J Kenneth Salisbury. A new actuation approach for human friendly robot design. The international journal of robotics research, 23(4-5):379–398, 2004.

[166] Sang-Ho Hyon, Joshua G Hale, and Gordon Cheng. Full-body compliant human–humanoid interaction: balancing in the presence of unknown external forces. IEEE Transactions on Robotics, 23(5):884–898, 2007.

[167] Taisuke Sugaiwa, Hiroyasu Iwata, and Shigeki Sugano. Shock absorbing skin design for human-symbiotic robot at the worst case collision. In Humanoid Robots, 2008. Humanoids 2008. 8th IEEE-RAS International Conference on, pages 481–486. IEEE, 2008.

[168] Oliver Eiberger, Sami Haddadin, Michael Weis, Alin Albu-Schäffer, and Gerd Hirzinger. On joint design with intrinsic variable compliance: Derivation of the dlr qa-joint. In Robotics and Automation (ICRA), 2010 IEEE International Conference on, pages 1687–1694. IEEE, 2010.

[169] Alin Albu-Schäffer, Oliver Eiberger, Matthias Fuchs, Markus Grebenstein, Sami Haddadin, Christian Ott, Andreas Stemmer, Thomas Wimböck, Sebastian Wolf, Christoph Borst, et al. Anthropomorphic soft robotics–from torque control to variable intrinsic compliance. In Robotics research, pages 185–207. Springer, 2011.

[170] Joris De Schutter and Hendrik Van Brussel. Compliant robot motion i. a formalism for specifying compliant motion tasks. The International Journal of Robotics Research, 7(4):3–17, 1988.

[171] Joris De Schutter and Hendrik Van Brussel. Compliant robot motion ii. a control approach based on external control loops. The International Journal of Robotics Research, 7(4):18–33, 1988.

[172] Marc G Carmichael, Dikai Liu, and Kenneth J Waldron. A framework for singularity-robust manipulator control during physical human-robot interaction. The International Journal of Robotics Research, 36(5-7):861–876, 2017.

[173] Xiang Li, Yongping Pan, Gong Chen, and Haoyong Yu. Multi-modal control scheme for rehabilitation robotic exoskeletons. The International Journal of Robotics Research, 36(5-7):759–777, 2017.

[174] Milad Geravand, Wolfgang Rampeltshammer, and Angelika Peer. Control of mobility assistive robot for human fall prevention. In Rehabilitation Robotics (ICORR), 2015 IEEE International Conference on, pages 882–887. IEEE, 2015.

[175] Giovanni Tonietti, Riccardo Schiavi, and Antonio Bicchi. Design and control of a variable stiffness actuator for safe and fast physical human/robot interaction. In Robotics and Automation (ICRA), 2005 IEEE International Conference on, pages 526–531. IEEE, 2005.

[176] Alin Albu-Schaffer, Oliver Eiberger, Markus Grebenstein, Sami Haddadin, Christian Ott, Thomas Wimbock, Sebastian Wolf, and Gerd Hirzinger. Soft robotics. IEEE Robotics & Automation Magazine, 15(3), 2008.

[177] Emmanuel Dean-Leon, Florian Bergner, Karinne Ramirez-Amaro, and Gordon Cheng. From multi-modal tactile signals to a compliant control. In Humanoid Robots (Humanoids), 2016 IEEE-RAS 16th International Conference on, pages 892–898. IEEE, 2016.

[178] Alexander Dietrich, Kristin Bussmann, Florian Petit, Paul Kotyczka, Christian Ott, Boris Lohmann, and Alin Albu-Schäffer. Whole-body impedance control of wheeled mobile manipulators. Autonomous Robots, pages 1–13, 2015.

[179] Roberto Calandra, Serena Ivaldi, Marc Peter Deisenroth, et al. Learning torque control in presence of contacts using tactile sensing from robot skin. In 2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids), pages 690–695. IEEE, 2015.

[180] Ewald Lutscher, Martin Lawitzky, Gordon Cheng, and Sandra Hirche. A control strategy for operating unknown constrained mechanisms. In Robotics and Automation (ICRA), 2010 IEEE International Conference on, pages 819–824. IEEE, 2010.

[181] Rhama Dwiputra and Gerhard K Kraetzschmar. Haptic interface for domestic service robot. Journal of Automation and Control Engineering Vol, 2(3), 2014.

[182] François Ferland, Dominic Létourneau, Arnaud Aumont, Julien Frémy, Marc-Antoine Legault, Michel Lauria, and François Michaud. Natural interaction design of a humanoid robot. Journal of Human-Robot Interaction, 1(2):118–134, 2012.

[183] François Ferland, Arnaud Aumont, Dominic Létourneau, and François Michaud. Taking your robot for a walk: Force-guiding a mobile robot using compliant arms. In Proceedings of the 8th ACM/IEEE international conference on Human-robot interaction, pages 309–316. IEEE Press, 2013.

[184] Luis Sentis, Josh Petersen, and Roland Philippsen. Implementation and stability analysis of prioritized whole-body compliant controllers on a wheeled humanoid robot in uneven terrains. Autonomous Robots, 35(4):301–319, 2013.

[185] Alexander Dietrich. Whole-body impedance control of wheeled humanoid robots, volume 116. Springer, 2016.

[186] Yuquan Wang, Christian Smith, Yiannis Karayiannidis, and Petter Ögren. Whole body control of a dual-arm mobile robot using a virtual kinematic chain. International Journal of Humanoid Robotics, 13(1), 2016.

[187] Yasuhisa Hirata. Coordinated transportation of a single object by omni-directional mobile robots with body force sensor. Journal of Robotics and Mechatronics, pages 242–248, 2000.

[188] Kazuhiro Kosuge, Tomohiro Hayashi, Yasuhisa Hirata, and Ryosuke Tobiyama. Dance partner robot-ms dancer. In Intelligent Robots and Systems (IROS), 2003 IEEE/RSJ International Conference on, volume 4, pages 3459–3464. IEEE, 2003.

[189] Takahiro Takeda, Yasuhisa Hirata, and Kazuhiro Kosuge. Dance step estimation method based on hmm for dance partner robot. IEEE Transactions on Industrial Electronics, 54(2):699–706, 2007.

[190] Zhao Liu, Yoshinori Koike, Takahiro Takeda, Yasuhisa Hirata, Ken Chen, and Kazuhiro Kosuge. Development of a passive type dance partner robot. In Advanced Intelligent Mechatronics, 2008. AIM 2008. IEEE/ASME International Conference on, pages 1070–1075. IEEE, 2008.

[191] Kwan Suk Kim, Alan S Kwok, Gray C Thomas, and Luis Sentis. Fully omnidirectional compliance in mobile robots via drive-torque sensor feedback. In Intelligent Robots and Systems (IROS), 2014 IEEE/RSJ International Conference on, pages 4757–4763. IEEE, 2014.

[192] Kwan Suk Kim, Travis Llado, and Luis Sentis. Full-body collision detection and reaction with omnidirectional mobile platforms: a step towards safe human–robot interaction. Autonomous Robots, 40(2):325–341, 2016.

[193] Julien Frémy, François Ferland, Michel Lauria, and François Michaud. Force-guidance of a compliant omnidirectional non-holonomic platform. Robotics and Autonomous Systems, 62(4):579–590, 2014.

[194] Diego Felipe Paez Granados, Jun Kinugawa, Yasuhisa Hirata, and Kazuhiro Kosuge. Guiding human motions in physical human-robot interaction through com motion control of a dance teaching robot. In Humanoid Robots (Humanoids), 2016 IEEE-RAS 16th International Conference on, pages 279–285. IEEE, 2016.

[195] Tiffany L Chen and Charles C Kemp. Lead me by the hand: Evaluation of a direct physical interface for nursing assistant robots. In Proceedings of the 5th ACM/IEEE international conference on Human-robot interaction, pages 367–374. IEEE Press, 2010.

[196] Tiffany L Chen and Charles C Kemp. A direct physical interface for navigation and positioning of a robotic nursing assistant. Advanced Robotics, 25(5):605–627, 2011.

[197] Tiffany L. Chen, Tapomayukh Bhattacharjee, J. Lucas McKay, Jacquelyn E. Borinski, Madeleine E. Hackney, Lena H. Ting, and Charles C. Kemp. Evaluation by expert dancers of a robot that performs partnered stepping via haptic interaction. PLOS ONE, 10(5):1–24, 5 2015.

[198] Ken'ichiro Nagasaka, Yasunori Kawanami, Satoru Shimizu, Takashi Kito, Toshimitsu Tsuboi, Atsushi Miyamoto, Tetsuharu Fukushima, and Hideki Shimomura. Whole-body cooperative force control for a two-armed and two-wheeled mobile robot using generalized inverse dynamics and idealized joint units. In Robotics and Automation (ICRA), 2010 IEEE International Conference on, pages 3377–3383. IEEE, 2010.

[199] Giorgio Cannata, Marco Maggiali, Giorgio Metta, and Giulio Sandini. An embedded artificial skin for humanoid robots. In Multisensor Fusion and Integration for Intelligent Systems (MFI), 2008. IEEE International Conference on, pages 434–438. IEEE, 2008.

[200] Brenna D Argall and Aude G Billard. A survey of tactile human–robot interactions. Robotics and autonomous systems, 58(10):1159–1176, 2010.

[201] Abderrahmane Kheddar and Aude Billard. A tactile matrix for whole-body humanoid haptic sensing and safe interaction. In Robotics and Biomimetics (ROBIO), 2011 IEEE International Conference on, pages 1433–1438. IEEE, 2011.

[202] Philipp Mittendorfer and Gordon Cheng. Humanoid multimodal tactile-sensing modules. Robotics, IEEE Transactions on, 27(3):401–410, 2011.

[203] Philipp Mittendorfer. From a Multi-modal Intelligent Cell to a Self-organizing Robotic Skin-Realizing Self and Enriching Robot Tactile Interaction. PhD thesis, Technische Universität München, 2015.

[204] Quentin Leboutet, Emmanuel Dean-León, and Gordon Cheng. Tactile-based compliance with hierarchical force propagation for omnidirectional mobile manipulators. In

Humanoid Robots (Humanoids), 2016 IEEE-RAS 16th International Conference on, pages 926–931. IEEE, 2016.

[205] Jerry Pratt, John Carff, Sergey Drakunov, and Ambarish Goswami. Capture point: A step toward humanoid push recovery. In 2006 6th IEEE-RAS international conference on humanoid robots, pages 200–207. IEEE, 2006.

[206] Benjamin Stephens. Humanoid push recovery. In 2007 7th IEEE-RAS International Conference on Humanoid Robots, pages 589–595. IEEE, 2007.

[207] Taisuke Kobayashi, Emmanuel Dean-Leon, Julio Rogelio Guadarrama-Olvera, Florian Bergner, and Gordon Cheng. Whole-body multicontact haptic human–humanoid interaction based on leader–follower switching: A robot dance of the “box step”. Advanced Intelligent Systems, page 2100038, 2021.

[208] Nicolas Mansard, Olivier Stasse, Paul Evrard, and Abderrahmane Kheddar. A versatile generalized inverted kinematics implementation for collaborative working humanoid robots: The stack of tasks. In 2009 International conference on advanced robotics, pages 1–6. IEEE, 2009.

[209] Emanuele Magrini, Fabrizio Flacco, and Alessandro De Luca. Control of generalized contact motion and force in physical human-robot interaction. In Robotics and Automation (ICRA), 2015 IEEE International Conference on, pages 2298–2304. IEEE, 2015.

[210] Emanuele Magrini and Alessandro De Luca. Human-robot coexistence and contact handling with redundant robots. In Intelligent Robots and Systems, 2017 IEEE/RSJ International Conference on, pages 4611–4617. IEEE, 2017.

[211] Kwan Suk Kim and Luis Sentis. Human body part multicontact recognition and detection methodology. In Robotics and Automation (ICRA), 2017 IEEE International Conference on, pages 1908–1915. IEEE, 2017.

[212] Alessandro De Luca, Alin Albu-Schaffer, Sami Haddadin, and Gerd Hirzinger. Collision detection and safe reaction with the dlr-iii lightweight manipulator arm. In Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on, pages 1623–1630. IEEE, 2006.

[213] Fabrizio Flacco, Antonio Paolillo, and Abderrahmane Kheddar. Residual-based contacts estimation for humanoid robots. In Humanoid Robots (Humanoids), 2016 IEEE-RAS 16th International Conference on, pages 409–415. IEEE, 2016.

[214] Miomir Vukobratović and J Stepanenko. On the stability of anthropomorphic systems. Mathematical biosciences, 15(1-2):1–37, 1972.

[215] Shuuji Kajita, Hirohisa Hirukawa, Kensuke Harada, and Kazuhito Yokoi. Introduction to humanoid robotics, volume 101. Springer, 2014.

[216] J Ferreau, H and al. qpoases user's manual – version 3.0. http://www.coin-or.org/qpOASES/doc/3.0/manual.pdf, 2014.

[217] Ewald Lutscher and Gordon Cheng. Hierarchical inequality task specification for indirect force controlled robots using quadratic programming. In Intelligent Robots and Systems, 2014 IEEE/RSJ International Conference on, pages 4722–4727. IEEE, 2014.

[218] Sebastian OH Madgwick, Andrew JL Harrison, Paul M Sharkey, Ravi Vaidyanathan, and William S Harwin. Measuring motion with kinematically redundant accelerometer arrays: Theory, simulation and implementation. Mechatronics, 23(5):518–529, 2013.

[219] Emmanuel Dean-Leon, Brennand Pierce, Florian Bergner, Philipp Mittendorfer, Karinne Ramirez-Amaro, Wolfgang Burger, and Gordon Cheng. Tomm: Tactile omnidirectional mobile manipulator. In Robotics and Automation (ICRA), 2017 IEEE International Conference on, pages 2441–2447. IEEE, 2017.

[220] Sachin Chitta, Eitan Marder-Eppstein, Wim Meeussen, Vijay Pradeep, Adolfo Rodríguez Tsouroukdissian, Jonathan Bohren, David Coleman, Bence Magyar, Gennaro Raiola, Mathias Lüdtke, et al. ros_control: A generic and simple control framework for ros. The Journal of Open Source Software, 2(20):456–456, 2017.

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

[222] Naotaka Mamizuka, Masataka Sakane, Koji Kaneoka, Noriyuki Hori, and Naoyuki Ochiai. Kinematic quantitation of the patellar tendon reflex using a tri-axial accelerometer. Journal of Biomechanics, 40(9):2107–2111, 2007.

[223] John Studenny and Pierre R Belanger. Robot manipulator control by acceleration feedback. In The 23rd IEEE Conference on Decision and Control, pages 1070–1072. IEEE, 1984.

[224] Andrea Calanca and Paolo Fiorini. A rationale for acceleration feedback in force control of series elastic actuators. IEEE Transactions on Robotics, 34(1):48–61, 2018.

[225] Francesco Nori, Silvio Traversaro, and Maurice Fallon. Sensor fusion and state estimation of the robot. Humanoid Robotics: A Reference, pages 1–29, 2018.

[226] John-Olof Nilsson and Isaac Skog. Inertial sensor arrays—a literature review. In Navigation Conference (ENC), 2016 European, pages 1–10. IEEE, 2016.

[227] A JKKW Padgaonkar, KW Krieger, and AI King. Measurement of angular acceleration of a rigid body using linear accelerometers. Journal of Applied Mechanics, 42(3):552–556, 1975.

[228] Davide Antonio Cucci, Omar García Crespillo, and Mehran Khaghani. An analysis of a gyro-free inertial system for ins/gnss navigation. In Navigation Conference (ENC), 2016 European, pages 1–7. IEEE, 2016.

[229] AG Mikov and AS Galov. Data processing algorithms for mems based multi-component inertial measurement unit for indoor navigation. In Integrated Navigation Systems (ICINS), 2017 24th Saint Petersburg International Conference on, pages 1–7. IEEE, 2017.

[230] Isaac Skog, John-Olof Nilsson, and Peter Händel. Pedestrian tracking using an imu array. In 2014 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT), pages 1–4. IEEE, 2014.

[231] Isaac Skog, John-Olof Nilsson, Peter Händel, and Arye Nehorai. Inertial sensor arrays, maximum likelihood, and cramér–rao bound. IEEE Transactions on Signal Processing, 64(16):4218–4227, 2016.

[232] Johan Wahlström, Isaac Skog, and Peter Händel. Inertial sensor array processing with motion models. In 2018 21st International Conference on Information Fusion (FUSION), pages 788–793. IEEE, 2018.

[233] Claudia Latella, Silvio Traversaro, Diego Ferigo, Yeshasvi Tirupachuri, Lorenzo Rapetti, Francisco Javier Andrade Chavez, Francesco Nori, and Daniele Pucci. Simultaneous floating-base estimation of human kinematics and joint torques. Sensors, 19(12):2794, 2019.

[234] Morgan Quigley, Reuben Brewer, Sai P Soundararaj, Vijay Pradeep, Quoc Le, and Andrew Y Ng. Low-cost accelerometers for robotic manipulator perception. In 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 6168–6174. IEEE, 2010.

[235] Peng Cheng and Bengt Oelmann. Joint-angle measurement using accelerometers and gyroscopes–a survey. IEEE Transactions on instrumentation and measurement, 59(2):404–414, 2010.

[236] Nicholas Rotella, Sean Mason, Stefan Schaal, and Ludovic Righetti. Inertial sensor-based humanoid joint state estimation. In Robotics and Automation (ICRA), 2016 IEEE International Conference on, pages 1825–1831. IEEE, 2016.

[237] Yizhou Wang, Wenjie Chen, and Masayoshi Tomizuka. Extended kalman filtering for robot joint angle estimation using mems inertial sensors. IFAC Proceedings Volumes, 46(5):406–413, 2013.

[238] Juho Vihonen, Janne Honkakorpi, Janne Tuominen, Jouni Mattila, and Ari Visa. Linear accelerometers and rate gyros for rotary joint angle estimation of heavy-duty mobile manipulators using forward kinematic modeling. IEEE/ASME Transactions on Mechatronics, 21(3):1765–1774, 2016.

[239] Juho Vihonen, Jouni Mattila, and Ari Visa. Joint-space kinematic model for gravity-referenced joint angle estimation of heavy-duty manipulators. IEEE Transactions on Instrumentation and Measurement, 66(12):3280–3288, 2017.

[240] Xiaolong Zhang, Eelis Peltola, and Jouni Mattila. Joint angle estimation for floating base robots utilizing mems imus. In 2017 IEEE International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM), pages 282–287. IEEE, 2017.

[241] Xiaolong Zhang, Eelis Peltola, and Jouni Mattila. Angle estimation for robotic arms on floating base using low-cost imus. In 2018 IEEE International Conference on Robotics and Automation (ICRA), pages 1458–1465. IEEE, 2018.

[242] Daniel Kubus, Corrado Guarino Lo Bianco, and Friedrich M Wahl. A sensor fusion approach to improve joint angle and angular rate signals in articulated robots. In 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 2736–2741. IEEE, 2012.

[243] Philip Roan, Nikhil Deshpande, Yizhou Wang, and Benjamin Pitzer. Manipulator state estimation with low cost accelerometers and gyroscopes. In 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 4822–4827. IEEE, 2012.

[244] Janne Honkakorpi. Mems-based motion state estimation and control of hydraulic manipulators. Tampereen teknillinen yliopisto. Julkaisu-Tampere University of Technology. Publication; 1219, 2014.

[245] X Xinjilefu, Siyuan Feng, and Christopher G Atkeson. A distributed mems gyro network for joint velocity estimation. In Robotics and Automation (ICRA), 2016 IEEE International Conference on, pages 1879–1884. IEEE, 2016.

[246] Juho Vihonen, Janne Honkakorpi, Jouni Mattila, and Ari Visa. Geometry-aided angular acceleration sensing of rigid multi-body manipulator using mems rate gyros and linear accelerometers. In 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 2514–2520. IEEE, 2013.

[247] Juho Vihonen, Janne Honkakorpi, Jouni Mattila, and Ari Visa. Novel pairwise coupled kinematic solution for algebraic angular acceleration estimation of serial link manipulators. In Robotics and Automation (ICRA), 2015 IEEE International Conference on, pages 809–814. IEEE, 2015.

[248] Laurence McLean, Lei Chen, and William SP Robertson. A robotic joint sensor. In 2015 Australasian Conference on Robotics and Automation. ARAA, 2015.

[249] Laurence McLean. High-Order Robotic Joint Sensing with Multiple Accelerometer and Gyroscope Systems. PhD thesis, University of Adelaide, 2018.

[250] Patrick Schopp, Hagen Graf, Michael Maurer, Michailas Romanovas, Lasse Klingbeil, and Yiannos Manoli. Observing relative motion with three accelerometer triads. IEEE Transactions on Instrumentation and Measurement, 63(12):3137–3151, 2014.

[251] Sungsu Park and Sung Kyung Hong. Angular rate estimation using a distributed set of accelerometers. Sensors, 11(11):10444–10457, 2011.

[252] Patrick Schopp, Lasse Klingbeil, Christian Peters, and Yiannos Manoli. Design, geometry evaluation, and calibration of a gyroscope-free inertial measurement unit. Sensors and Actuators A: Physical, 162(2):379–387, 2010.

[253] Patrick Schopp, Axel Rottmann, Lasse Klingbeil, Wolfram Burgard, and Yiannos Manoli. Gaussian process based state estimation for a gyroscope-free imu. In SENSORS, 2010 IEEE, pages 873–878. IEEE, 2010.

[254] Ezzaldeen Edwan, Stefan Knedlik, and Otmar Loffeld. Constrained angular motion estimation in a gyro-free imu. IEEE Transactions on Aerospace and Electronic Systems, 47(1):596–610, 2011.

[255] Chaojun Liu, Shuai Yu, Shengzhi Zhang, Xuebing Yuan, and Sheng Liu. An effective unscented kalman filter for state estimation of a gyro-free inertial measurement unit. In Position, Location and Navigation Symposium-PLANS 2014, 2014 IEEE/ION, pages 230–235. IEEE, 2014.

[256] Jeng-Heng Chen, Sou-Chen Lee, and Daniel B DeBra. Gyroscope free strapdown inertial measurement unit by six linear accelerometers. Journal of Guidance, Control, and Dynamics, 17(2):286–290, 1994.

[257] William T Conlin. Inertial measurement. arXiv preprint arXiv:1708.04325, 2017.

[258] Pei-Chun Lin, Jau-Ching Lu, Chia-Hung Tsai, and Chi-Wei Ho. Design and implementation of a nine-axis inertial measurement unit. IEEE/ASME Transactions on Mechatronics, 17(4):657–668, 2011.

[259] Adrian Waegli, Jan Skaloud, Stéphane Guerrier, Maria Eulalia Parés, and Ismael Colomina. Noise reduction and estimation in multiple micro-electro-mechanical inertial systems. Measurement Science and Technology, 21(6):065201, 2010.

[260] Isaac Skog, John-Olof Nilsson, and Peter Händel. An open-source multi inertial measurement unit (mimu) platform. In 2014 International Symposium on Inertial Sensors and Systems (ISISS), pages 1–4. IEEE, 2014.

[261] T Brunner, S Changey, J Ph Lauffenburger, and M Basset. Multiple mems-imu localization: Architecture comparison and performance assessment. ICINS, 2015, 2015.

[262] Derek K Shaeffer. Mems inertial sensors: A tutorial overview. IEEE Communications Magazine, 51(4):100–109, 2013.

[263] Shih-Chii Liu and Tobi Delbruck. Neuromorphic sensory systems. Current opinion in neurobiology, 20(3):288–295, 2010.

[264] Stefano Caviglia, Maurizio Valle, and Chiara Bartolozzi. Asynchronous, event-driven readout of posfet devices for tactile sensing. In Circuits and Systems (ISCAS), 2014 IEEE International Symposium on, pages 2648–2651. IEEE, 2014.

[265] Stefano Caviglia, Luigi Pinna, Maurizio Valle, and Chiara Bartolozzi. An event-driven posfet taxel for sustained and transient sensing. In Circuits and Systems (ISCAS), 2016 IEEE International Symposium on, pages 349–352. IEEE, 2016.

[266] Wang Wei Lee, Sunil L Kukreja, and Nitish V Thakor. A kilohertz kilotaxel tactile sensor array for investigating spatiotemporal features in neuromorphic touch. In Biomedical Circuits and Systems Conference (BioCAS), 2015 IEEE, pages 1–4. IEEE, 2015.

[267] Florian Bergner, Philipp Mittendorfer, Emmanuel Dean-Leon, and Gordon Cheng. Event-based signaling for reducing required data rates and processing power in a large-scale artificial robotic skin. In Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on, pages 2124–2129. IEEE, 2015.

[268] Florian Bergner, Emmanuel Dean-Leon, and Gordon Cheng. Event-based signaling for large-scale artificial robotic skin-realization and performance evaluation. In Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on, pages 4918–4924. IEEE, 2016.

[269] Florian Bergner, Emmanuel Dean-Leon, and Gordon Cheng. Efficient event-driven reactive control for large scale robot skin. In Robotics and Automation (ICRA), 2017 IEEE International Conference on, pages 394–400. IEEE, 2017.

[270] Hosam Rowaihy, Sharanya Eswaran, Matthew Johnson, Dinesh Verma, Amotz Bar-Noy, Theodore Brown, and Thomas La Porta. A survey of sensor selection schemes in wireless sensor networks. In Unattended Ground, Sea, and Air Sensor Technologies and Applications IX, volume 6562, page 65621A. International Society for Optics and Photonics, 2007.

[271] Phani Chavali and Arye Nehorai. Scheduling and power allocation in a cognitive radar network for multiple-target tracking. IEEE Transactions on Signal Processing, 60(2):715–729, 2011.

[272] Sundeep Prabhakar Chepuri and Geert Leus. Sparsity-promoting sensor selection for non-linear measurement models. IEEE Transactions on Signal Processing, 63(3):684–698, 2014.

[273] Fang Bian, David Kempe, and Ramesh Govindan. Utility based sensor selection. In Proceedings of the 5th international conference on Information processing in sensor networks, pages 11–18, 2006.

[274] Alfred O Hero and Douglas Cochran. Sensor management: Past, present, and future. IEEE Sensors Journal, 11(12):3064–3075, 2011.

[275] William J Welch. Branch-and-bound search for experimental designs based on d optimality and other criteria. Technometrics, 24(1):41–48, 1982.

[276] Siddharth Joshi and Stephen Boyd. Sensor selection via convex optimization. IEEE Transactions on Signal Processing, 57(2):451–462, 2008.

[277] Manohar Shamaiah, Siddhartha Banerjee, and Haris Vikalo. Greedy sensor selection: Leveraging submodularity. In 49th IEEE conference on decision and control (CDC), pages 2572–2577. IEEE, 2010.

[278] Abolfazl Hashemi, Mahsa Ghasemi, Haris Vikalo, and Ufuk Topcu. A randomized greedy algorithm for near-optimal sensor scheduling in large-scale sensor networks. In 2018 Annual American Control Conference (ACC), pages 1027–1032. IEEE, 2018.

[279] TR Williams and KR Fyfe. Planar accelerometer configurations. Journal of applied mechanics, 71(1):10–14, 2004.

[280] Philippe Cardou and Jorge Angeles. Singularity analysis of accelerometer strapdowns for the estimation of the acceleration field of a planar rigidbody motion. In Proceedings of the 12th World Congress in Mechanism and Machine Science, 2007.

[281] Bruno Zappa, Giovanni Legnani, Anton J Van Den Bogert, and Riccardo Adamini. On the number and placement of accelerometers for angular velocity and acceleration determination. Transactions-American Society of Mechanical Engineers, Journal of Dynamic Systems Measurement and Control, 123(3):552–553, 2001.

[282] Giuseppe Baselli, G Legnani, P Franco, F Brognoli, A Marras, F Quaranta, and B Zappa. Assessment of inertial and gravitational inputs to the vestibular system. Journal of Biomechanics, 34(6):821–826, 2001.

[283] Nitesh Sahu, Prabhu Babu, Arun Kumar, and Rajendar Bahl. A novel algorithm for optimal placement of multiple inertial sensors to improve the sensing accuracy. IEEE Transactions on Signal Processing, 68:142–154, 2019.

[284] Reza N Jazar. Theory of applied robotics: kinematics, dynamics, and control. Springer Science & Business Media, 2010.

[285] Zhongkai Qin, Luc Baron, and Lionel Birglen. Robust design of inertial measurement units based on accelerometers. Journal of Dynamic Systems, Measurement, and Control, 131(3):031010, 2009.

[286] Mahdi Jafari and Jafar Roshanian. Optimal redundant sensor configuration for accuracy and reliability increasing in space inertial navigation systems. The Journal of Navigation, 66(2):199–208, 2013.

[287] Mehdi Jafari. Optimal redundant sensor configuration for accuracy increasing in space inertial navigation system. Aerospace Science and Technology, 47:467–472, 2015.

[288] Jin Woo Song and Chan Gook Park. Optimal configuration of redundant inertial sensors considering lever arm effect. IEEE Sensors Journal, 16(9):3171–3180, 2016.

[289] Bosch Sensortec. Bma250. digital triaxial acceleration sensor. datasheet, 2014.

[290] Patrick Schopp, Hagen Graf, Wolfram Burgard, and Yiannos Manoli. Self-calibration of accelerometer arrays. IEEE Transactions on Instrumentation and Measurement, 65(8):1913–1925, 2016.

[291] John-Olof Nilsson, Isaac Skog, and Peter Händel. Aligning the forces—eliminating the misalignments in imu arrays. IEEE Transactions on Instrumentation and Measurement, 63(10):2498–2500, 2014.

[292] Håkan Carlsson, Isaac Skog, and Joakim Jaldén. On-the-fly geometric calibration of inertial sensor arrays. In Indoor Positioning and Indoor Navigation (IPIN), 2017 International Conference on, pages 1–6. IEEE, 2017.

[293] Philipp Mittendorfer and Gordon Cheng. 3d surface reconstruction for robotic body parts with artificial skins. In Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on, pages 4505–4510. IEEE, 2012.

[294] Taeyoon Lee, Jaewoon Kwon, and Frank C Park. A natural adaptive control law for robot manipulators. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 1–9. IEEE, 2018.

[295] Bruno Siciliano and Oussama Khatib. Springer handbook of robotics. Springer, 2016.

[296] Nacim Ramdani and Philippe Poignet. Robust dynamic experimental identification of robots with set membership uncertainty. IEEE/ASME Transactions on Mechatronics, 10(2):253–256, 2005.

[297] Giovanni Sutanto, Austin Wang, Yixin Lin, Mustafa Mukadam, Gaurav Sukhatme, Akshara Rai, and Franziska Meier. Encoding physical constraints in differentiable newton-euler algorithm. arXiv preprint arXiv:2001.08861, 2020.

[298] Vinzenz Bargsten, Jose de Gea Fernandez, and Yohannes Kassahun. Experimental robot inverse dynamics identification using classical and machine learning techniques. In Proceedings of ISR 2016: 47st International Symposium on Robotics, pages 1–6. VDE, 2016.

[299] Fernando Díaz Ledezma and Sami Haddadin. First-order-principles-based constructive network topologies: An application to robot inverse dynamics. In 2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids), pages 438–445. IEEE, 2017.

[300] H. Hjalmarsson, C. R. Rojas, and D. E. Rivera. System identification: A Wiener-Hammerstein benchmark. Control Engineering Practice, 20(11):1095–1096, 2012.

[301] A. Kroll and H. Schulte. Benchmark problems for nonlinear system identification and control using soft computing methods: need and overview. Applied Soft Computing, 25(12):496–513, 2014.

[302] C. R. Rojas, P. E. Valenzuela, and R. A. Rojas. A critical view on benchmarks based on randomly generated systems. In 17th IFAC Symposium on System Identification, Beijing, Chine, October 2015. IFAC, 2015.

[303] Stefan Bethge, Jörn Malzahn, Nikolaos Tsagarakis, and Darwin Caldwell. Flobaroid–a software package for the identification of robot dynamics parameters. In International Conference on Robotics in Alpe-Adria Danube Region, pages 156–165. Springer, 2017.

[304] Yan Wang, Radian Gondokaryono, Adnan Munawar, and Gregory Scott Fischer. A convex optimization-based dynamic model identification package for the da vinci research kit. IEEE Robotics and Automation Letters, 4(4):3657–3664, 2019.

[305] Poommitol Chaicherdkiat, Tobias Osterloh, Chayakorn Netramai, and Jürgen Reßmann. Simulation-based parameter identification framework for the calibration of rigid body simulation models. In 2020 SICE International Symposium on Control Systems (SICE ISCS), pages 12–19. IEEE, 2020.

[306] Gianluca Antonelli, Fabrizio Caccavale, and Pasquale Chiacchio. A systematic procedure for the identification of dynamic parameters of robot manipulators. Robotica, 17(4):427–435, 1999.

[307] Peter I Corke et al. A robotics toolbox for matlab. IEEE Robotics & Automation Magazine, 3(1):24–32, 1996.

[308] Lorenzo Sciavicco, Bruno Siciliano, and Luigi Villani. Lagrange and newton-euler dynamic modeling of a gear-driven robot manipulator with inclusion of motor inertia effects. Advanced robotics, 10(3):317–334, 1995.

[309] Dawoon Jung, Joono Cheong, Dong Il Park, and Chanhun Park. Backward sequential approach for dynamic parameter identification of robot manipulators. International Journal of Advanced Robotic Systems, 15(1):1729881418758578, 2018.

[310] Takuma Katsumata, Benjamin Navarro, Vincent Bonnet, Philippe Fraisse, André Crosnier, and Gentiane Venture. Optimal exciting motion for fast robot identification. application to contact painting tasks with estimated external forces. Robotics and Autonomous Systems, 113:149–159, 2019.

[311] Taeyoon Lee. Geometric Methods for Dynamic Model-Based Identification and Control of Multibody Systems. PhD thesis, Seoul National University, 2019.

[312] C Canudas De Wit, Hans Olsson, Karl Johan Astrom, and Pablo Lischinsky. A new model for control of systems with friction. IEEE Transactions on automatic control, 40(3):419–425, 1995.

[313] Maxime Gautier and Sébastien Briot. New method for global identification of the joint drive gains of robots using a known payload mass. In 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 3728–3733. IEEE, 2011.

[314] Maxime Gautier and Sébastien Briot. Global identification of drive gains parameters of robots using a known payload. In 2012 IEEE International Conference on Robotics and Automation, pages 2812–2817. IEEE, 2012.

[315] S. M. Hashemi and H. Werner. Parameter identification of a robot arm using separable least squares technique. In 2009 European Control Conference (ECC), pages 2199–2204, Aug 2009.

[316] B. Bona and M. Indri. Friction compensation in robotics: An overview. In Proceedings of the 44th IEEE Conference on Decision and Control, pages 4360–4367, Dec 2005.

[317] Michael Grant and Stephen Boyd. Cvx: Matlab software for disciplined convex programming, version 2.1, 2014.

[318] MOSEK ApS. Mosek optimization suite, 2019.

[319] Bruce W Suter. Multirate and wavelet signal processing. Elsevier, 1997.

[320] Wisama Khalil and Denis Creusot. Symoro+: a system for the symbolic modelling of robots. Robotica, 15(2):153–161, 1997.

[321] Wisama Khalil, Aravindkumar Vijayalingam, Bogdan Khomutenko, Izzatbek Mukhanov, Philippe Lemoine, and Gaël Ecorchard. Opensymoro: An open-source software package for symbolic modelling of robots. In 2014 IEEE/ASME International Conference on Advanced Intelligent Mechatronics, pages 1206–1211. IEEE, 2014.

[322] Martin L Felis. Rbdl: an efficient rigid-body dynamics library using recursive algorithms. Autonomous Robots, 41(2):495–511, 2017.

[323] Ruben Smits, H Bruyninckx, and E Aertbeliën. Kdl: Kinematics and dynamics library, 2011.

[324] Toru Iwasaki, Gentiane Venture, and Eiichi Yoshida. Identification of the inertial parameters of a humanoid robot using grounded sole link. In 2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012), pages 449–454. IEEE, 2012.

[325] Patrick M Wensing, Sangbae Kim, and Jean-Jacques E Slotine. Linear matrix inequalities for physically consistent inertial parameter identification: A statistical perspective on the mass distribution. IEEE Robotics and Automation Letters, 3(1):60–67, 2017.

[326] Junji Oaki and Shuichi Adachi. Decoupling identification for serial two-link robot arm with elastic joints. IFAC Proceedings Volumes, 42(10):1417–1422, 2009.

[327] Jean-Jacques E Slotine and Weiping Li. On the adaptive control of robot manipulators. The international journal of robotics research, 6(3):49–59, 1987.

[328] Jean-Jacques E Slotine and Weiping Li. Composite adaptive control of robot manipulators. Automatica, 25(4):509–519, 1989.

[329] John J Craig, Ping Hsu, and S Shankar Sastry. Adaptive control of mechanical manipulators. The International Journal of Robotics Research, 6(2):16–28, 1987.

[330] Daniele Pucci, Francesco Romano, and Francesco Nori. Collocated adaptive control of underactuated mechanical systems. IEEE Transactions on Robotics, 31(6):1527–1536, 2015.

[331] Shuuji Kajita, Fumio Kanehiro, Kenji Kaneko, Kiyoshi Fujiwara, Kensuke Harada, Kazuhito Yokoi, and Hirohisa Hirukawa. Biped walking pattern generation by using preview control of zero-moment point. In 2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422), volume 2, pages 1620–1626. IEEE, 2003.

[332] Dimitar Dimitrov, Alexander Sherikov, and Pierre-Brice Wieber. A sparse model predictive control formulation for walking motion generation. In 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 2292–2299. IEEE, 2011.

[333] Shuuji Kajita, Fumio Kanehiro, Kenji Kaneko, Kiyoshi Fujiwara, Kensuke Harada, Kazuhito Yokoi, and Hirohisa Hirukawa. Resolved momentum control: Humanoid motion planning based on the linear and angular momentum. In Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453), volume 2, pages 1644–1650. IEEE, 2003.

[334] Hongkai Dai, Andrés Valenzuela, and Russ Tedrake. Whole-body motion planning with centroidal dynamics and full kinematics. In 2014 IEEE-RAS International Conference on Humanoid Robots, pages 295–302. IEEE, 2014.

[335] Camille Brasseur, Alexander Sherikov, Cyrille Collette, Dimitar Dimitrov, and Pierre-Brice Wieber. A robust linear mpc approach to online generation of 3d biped walking motion. In 2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids), pages 595–601. IEEE, 2015.

[336] Nicholas Rotella, Alexander Herzog, Stefan Schaal, and Ludovic Righetti. Humanoid momentum estimation using sensed contact wrenches. In Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on, pages 556–563. IEEE, 2015.

[337] Stylianos Piperakis and Panos Trahanias. Non-linear zmp based state estimation for humanoid robot locomotion. In 2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids), pages 202–209. IEEE, 2016.

[338] Ken Masuya and Tomomichi Sugihara. Com motion estimation of a humanoid robot based on a fusion of dynamics and kinematics information. In 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 3975–3980. IEEE, 2015.

[339] Justin Carpentier, Mehdi Benallegue, Nicolas Mansard, and Jean-Paul Laumond. Center-of-mass estimation for a polyarticulated system in contact—a spectral approach. IEEE Transactions on Robotics, 32(4):810–822, 2016.

[340] Kenya Mori, Ko Ayusawa, and Eiichi Yoshida. Online center of mass and momentum estimation for a humanoid robot based on identification of inertial parameters. In 2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids), pages 1–9. IEEE, 2018.

[341] Noémie Laure Gwendoline Jaquier. Robot skills learning with riemannian manifolds: Leveraging geometry-awareness in robot learning, optimization and control. Technical report, EPFL, 2020.

[342] Maher Moakher and Philipp G Batchelor. Symmetric positive-definite matrices: From geometry to applications and visualization. In Visualization and Processing of Tensor Fields, pages 285–298. Springer, 2006.

[343] Rajendra Bhatia. Positive definite matrices. Princeton university press, 2009.

[344] Xavier Pennec, Pierre Fillard, and Nicholas Ayache. A riemannian framework for tensor computing. International Journal of computer vision, 66(1):41–66, 2006.

[345] Michael Lutter, Johannes Silberbauer, Joe Watson, and Jan Peters. Differentiable physics models for real-world offline model-based reinforcement learning. arXiv preprint arXiv:2011.01734, 2020.

[346] Eric Heiden, David Millard, Erwin Coumans, Yizhou Sheng, and Gaurav S Sukhatme. Neuralsim: Augmenting differentiable simulators with neural networks. arXiv preprint arXiv:2011.04217, 2020.

[347] Balakumar Sundaralingam and Tucker Hermans. In-hand object-dynamics inference using tactile fingertips. IEEE Transactions on Robotics, 2021.

[348] Maher Moakher. A differential geometric approach to the geometric mean of symmetric positive-definite matrices. SIAM Journal on Matrix Analysis and Applications, 26(3):735–747, 2005.

[349] Fredrik Gustafsson. Determining the initial states in forward-backward filtering. IEEE Transactions on signal processing, 44(4):988–992, 1996.

[350] qpmad qp solver. `https://github.com/asherikov/qpmad`, 2021.

[351] Bartolomeo Stellato, Goran Banjac, Paul Goulart, Alberto Bemporad, and Stephen Boyd. Osqp: An operator splitting solver for quadratic programs. Mathematical Programming Computation, 12(4):637–672, 2020.

[352] Qld qp solver. `https://github.com/jrl-umi3218/eigen-qld`, 2019.

[353] Qp solver benchmark. `https://github.com/asherikov/qpmad_benchmark`, 2021.

[354] Donald Goldfarb and Ashok Idnani. A numerically stable dual method for solving strictly convex quadratic programs. Mathematical programming, 27(1):1–33, 1983.

[355] Silvio Traversaro, Andrea Del Prete, Riccardo Muradore, Lorenzo Natale, and Francesco Nori. Inertial parameter identification including friction and motor dynamics. In 2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids), pages 68–73. IEEE, 2013.

[356] Silvio Traversaro, Andrea Del Prete, Serena Ivaldi, and Francesco Nori. Inertial parameters identification and joint torques estimation with proximal force/torque sensing. In 2015 IEEE International Conference on Robotics and Automation (ICRA), pages 2105–2110. IEEE, 2015.

[357] Eagle S Jones and Stefano Soatto. Visual-inertial navigation, mapping and localization: A scalable real-time causal approach. The International Journal of Robotics Research, 30(4):407–430, 2011.

[358] J Rogelio Guadarrama Olvera, Emmanuel Dean Leon, Florian Bergner, and Gordon Cheng. Plantar tactile feedback for biped balance and locomotion on unknown terrain. International Journal of Humanoid Robotics, 17(01):1950036, 2020.

[359] Thomas Feix, Javier Romero, Heinz-Bodo Schmiedmayer, Aaron M Dollar, and Danica Kragic. The grasp taxonomy of human grasp types. IEEE Transactions on human-machine systems, 46(1):66–77, 2015.

[360] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo. Robotics: modelling, planning and control. Springer Science & Business Media, 2009.

[361] P-B Wieber. Holonomy and nonholonomy in the dynamics of articulated motion. In Fast motions in biomechanics and robotics, pages 411–425. Springer, 2006.

[362] Chae H An, Christopher G Atkeson, and John M Hollerbach. Estimation of inertial parameters of rigid body links of manipulators. In 1985 24th IEEE Conference on Decision and Control, pages 990–995. IEEE, 1985.

[363] Roger A Horn and Fuzhen Zhang. Basic properties of the schur complement. In The Schur Complement and Its Applications, pages 17–46. Springer, 2005.