Alma Mater Studiorum - Università di Bologna

DOTTORATO DI RICERCA IN

AUTOMOTIVE PER UNA MOBILITÁ INTELLIGENTE

Ciclo 35

Settore Concorsuale: 09/A2 - MECCANICA APPLICATA ALLE MACCHINE

Settore Scientifico Disciplinare: ING-IND/13 - MECCANICA APPLICATA ALLE MACCHINE

TRAJECTORY PLANNING OF SINGLE AND DUAL-ARM ROBOTS FOR TIME-OPTIMAL HANDLING OF LIQUIDS AND OBJECTS

Presentata da: Roberto Di Leva

Coordinatore Dottorato

Nicoló Cavina

Supervisore

Marco Carricato

Co-supervisore

Alessandro Rivola

Esame finale anno 2023

Hana wa sakuragi,
Hito wa bushi

Abstract

This Thesis studies the optimal control problem of single-arm and dual-arm serial robots to achieve the time-optimal handling of liquids and objects.
The first topic deals with the planning of time-optimal anti-sloshing trajectories of an industrial robot carrying a cylindrical container filled with a liquid, considering 1-dimensional and 2-dimensional planar motions. A technique for the estimation of the *sloshing* height is presented, together with its extension to 3-dimensional motions. An experimental validation campaign is provided and discussed to assess the thoroughness of such a technique. As far as anti-sloshing trajectories are concerned, 2-dimensional paths are considered and, for each one of them, three constrained optimizations with different values of the sloshing-height thresholds are solved. Experimental results are presented to compare optimized and non-optimized motions.
The second part focuses on the time-optimal trajectory planning for dual-arm object handling, employing two collaborative robots (*cobots*) and adopting an admittance-control strategy. The chosen manipulation approach, known as *cooperative grasping*, is based on unilateral contact between the cobots and the object, and it may lead to slipping during motion if an internal prestress along the contact-normal direction is not prescribed. Thus, a *virtual penetration* is considered, aimed at generating the necessary internal prestress. The stability of cooperative grasping is ensured as long as the exerted forces on the object remain inside the static-friction cone. Constrained-optimization problems are solved for 3-dimensional paths: the virtual penetration is chosen among the control inputs of the problem and friction-cone conditions are treated as inequality constraints. Also in this case experiments are presented in order to prove evidence of the firm handling of the object, even for fast motions.

Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Meaning
w.r.t.	with respect to
i.e.	id est
e.g.	exempli gratia
NLP	Nonlinear Programming
IPOPT	Interior Point Optimizer
KKT	Karush-Kuhn-Tucker
QP	Quadratic Programming
SQP	Sequential Quadratic Programming
SOC	Second-Order Correction
ODE	Ordinary Differential Equation
OCP	Optimal Control Problem
FEM	Finite Element Method
SPH	Smooth Particle Hydrodynamics
DOF	Degree of Freedom
EOM	Equation of Motion
L model	Linear mass-spring-damper model
NL model	Nonlinear mass-spring-damper model
SH	Sloshing Height
MSH	Maximum Sloshing Height
OPT motion	Optimized motion
N-OPT motion	Non-optimized motion
AMR	Autonomous Mobile Robot
AGV	Automated Guided Vehicle
HRC	Human-Robot Cooperation
COM	Centre of Mass

Chapter 1

Introduction

Robotics entered the industrial world in the last decades, especially with the employment of serial manipulators[1]. To control the motion of a serial robot, offline planning of the trajectory that its end-effector has to follow is needed. In most cases, the trajectory planning of serial robots is made by writing the path in terms of a path parameter. The path parameter, together with its first and second-time derivatives, composes the motion law, hence giving the speed and acceleration with which the path has to be traveled. In other words, the path-parameter motion law allows the definition of the path time evolution, i.e., the end-effector trajectory. Nowadays, serial robots are widely used to fulfill tasks that could represent a danger for human operators or that may require an unsustainable human effort in the long run (i.e., welding of car chassis parts, grinding or milling, lifting of heavy objects, etc.). In addition, serial robots are inserted within operations that need a certain degree of precision and repeatability, such as pick&place tasks or accurate transport of goods between the machines of an industrial line. However, the benefits deriving from the use of serial robots may not justify the investment costs if the accomplishment of the described tasks does not envisage the observance of a certain *optimality* criterion.
The latter can be conveniently selected depending on the final purpose to be pursued by defining an appropriate objective function or cost functional that has to be minimized within the robot trajectory planning. Hence, the objective function can be chosen to:

- minimize the execution time of the trajectory in order to maximize the robot productivity without neglecting the limits on the available hardware in terms of maximum joint velocities and/or torques;

- minimize the power consumption for trajectory execution in order to reduce the mechanical stress in the motors and the variable costs for operating the robot.

Typically, multi-criterion optimizations are taken into account to minimize more than one quantity. The most common objective functions are the ones that consider a trade-off between minimum time and minimum energy consumption, or those balancing minimum time and minimum overall jerk of the path parameter, in order to obtain a smooth behavior in the robot accelerations.

[1] A serial manipulator represents an open kinematic chain in which a sequence of bodies is connected by means of successive revolute (in the majority of cases) or prismatic joints.

So far, the discussion has only focused on optimality concerning the robot characteristics and behavior without considering the state of the object to be processed or manipulated. In some cases, the condition of the components subject to the mechanical processing or the pick&place or transport operations is not influenced by the robot behavior during the execution of the trajectory. For example, in a welding operation, the parts to be linked are fixed on a specific frame; the robot only needs to follow the welding path executing the operation at a given velocity. Similarly, the rigid goods moved in a pick&place operation usually do not change their status along the trajectory, if they are suitably grabbed.
On the other hand, some industrial applications exist in which the trajectory planning needs the inclusion of *ad-hoc* time-dependent constraints, depending on the interaction between the robot and the manipulated item. An example is the transport of liquid-filled containers, in which the container is attached to the robot end-effector, and the liquid, free to move inside the container, is subject to the excitation represented by the end-effector acceleration. The behavior of the liquid inside the container is known as *sloshing* and has to be controlled during the whole motion to avoid overflowing. In this scenario, the trajectory planning needs to minimize an objective function (typically addressing minimal time), simultaneously respecting constraints on the given hardware (e.g., limits on the robot joint velocities and torques) and maintaining the sloshing height under a specified value.
A more complex case regards transporting heterogeneous objects with different sizes, shapes and weights. To achieve the highest degree of flexibility with respect to (w.r.t.) the object characteristics, a dual-arm setup can be considered. In order to avoid the use of specific grippers, the two robots may hold the manipulated item through contact points (or contact areas), without grabbing it, but rather relying on friction, thus realizing what is called *cooperative grasping*. In this case, the sole time-optimal trajectory would represent a failure in the execution of the task, as, given the unilateral constraints between the robot end-effectors and the object, the robots may lose contact with it. To avoid slipping of the object during motion, the forces exerted by the robots have to remain inside the static-friction cone. This constraint has to be added within the optimization, alongside the limits on the hardware.
This Thesis addresses the aforementioned two topics, namely the time-optimal anti-sloshing trajectory planning of a single-arm robot and the time-optimal dual-arm trajectory planning. In both cases, modeling the dynamics of the liquid and the object is needed. In the former case, the liquid dynamics have to be evaluated to estimate the peaks reached by the liquid during motion, hence allowing to relate the motion law of the path parameter with the liquid sloshing height. In the latter case, the net wrench necessary to grant the correct motion of the object has to be computed in order to express the friction-cone constraints as a function of the path parameter and its time derivatives.
The fact that the quantities under examination evolve within a specified time interval implies the arising of an optimal control problem. An optimal control problem can be seen as an infinite-dimensional extension of a *nonlinear programming* (NLP) problem, and its resolution requires particular techniques.
The structure of the Thesis is as follows. Chapter 2 gives a brief recap regarding the basics of nonlinear optimization; the general optimal control problem is presented, and the technique adopted to convert the infinite-dimensional problem into a problem with a finite set of variables and constraints is outlined, together with a practical

description of the algorithm employed for the NLP resolution.
Chapter 3 addresses the anti-sloshing problem; in particular, a novel technique for the sloshing-height estimation considering 3-dimensional motions of a cylindrical container carried by a serial robot is presented; the equations of motion governing the liquid behavior are provided, and the formulation of the sloshing height is given in terms of the corresponding generalized coordinates. The constrained-optimization problem is described with a particular focus on the expression of the sloshing constraints. The experimental setup is illustrated, and the video post-processing, adopted to compare the experimental results with the model predictions, is explained. Experiments, both for validation purposes of the proposed technique and for assessment of the optimization, are described and discussed.
Chapter 4 deals with dual-arm cooperative grasping. First, brief survey is given about the direct and indirect force control of serial robots. Then, the model for determining the interaction forces between the robots and the object is provided; the constrained-optimization problem is formulated, highlighting how the friction-cone inequalities are expressed in terms of the input variables. The setup for the experiments is depicted, and the relative results are provided and commented.
Finally, Chapter 5 draws conclusions and gives suggestions for further developments.

Chapter 2

Optimal Control Problem

2.1 Introduction

The constrained time-optimal trajectory planning of serial robots involves an *optimal control problem* (OCP) [1], in which the variables interested in the optimization evolve in time, and their trends are mostly represented by nonlinear functions, in the majority of cases. From a theoretical point of view, this implies an infinite-dimensional *nonlinear programming* (NLP) problem. The discretization of this kind of problem is crucial for a numerical resolution, and it can be achieved through direct methods. In particular, *multiple-shooting algorithms* enable the division of the time interval into smaller discretized time steps. This way, the problem results in a finite-dimensional NLP problem.
In NLP problems, the aim is to minimize a specific objective function or cost functional, simultaneously satisfying equality and inequality constraints. The main challenge in solving NLP problems is represented by the presence of inequality constraints. In particular, one of the main problems is detecting which constraints are active[1]. A technique, on which *active-set methods* are based, starts by guessing on the optimal active set (called *working set*), and then it solves a problem in which the constraints in the working set are treated as equalities and the others are ignored. If a solution is not found, a different working set is chosen and the procedure is repeated. It can be noted that the number of choices for the working set can be very large. For this reason, techniques based on active-set methods choose the working set in a more convenient way, by exploiting the knowledge of the functions that define the problem.
A different approach is represented by *interior-point methods*, in which the solutions generated at each iteration stay away from the barrier represented by the boundary of the feasible region defined by the inequality constraints. As the solution of the problem is approached, the barrier constraints are weakened to allow an increasingly accurate estimation of the solution.
The algorithms based on active-set or interior-point methods are usually combined with a series of techniques (called *globalization strategies*) that give information about the progress of the process toward the solution or prevent iterations from falling in infeasible points. More theoretical details regarding the available optimization techniques are outside the scope of this Thesis and can be found in the literature [2].
In our case, the OCP is tackled by adopting a multiple-shooting method to discretize the problem, and the problem-solving code is written by using CasADi [3], a framework

[1] An inequality constraint is termed active if it is satisfied as an equality.

implemented in Matlab, that adopts IPOPT[2], an open-source library based on a filter line-search interior-point method [4], for the NLP resolution.
This Chapter is structured as follows. Section 2.2 gives a short summary of the theory behind NLP resolution, starting from the Newton method up to the definition of the *Karush-Kuhn-Tucker* (KKT) system for an optimization problem with equality constraints. In Section 2.3 globalization strategies are illustrated. Section 2.4 provides an overview of the basic procedures of interior-point methods, before outlining the main steps of the IPOPT algorithm. Finally, in Section 2.5 the technique of multiple-shooting is introduced to translate the continuous OCP into a discrete NLP problem.

2.2 Nonlinear Programming Basics

The resolution of NLP problems requires finding a finite number of variables such that an objective function is minimized without violating a set of constraints. The involved functions are nonlinear in the interested variables, hence requiring particular techniques for the problem resolution [1].

2.2.1 Unconstrained Optimization in One Variable

A simple optimization problem involving one variable may be solved by the Newton method. The aim is to find x^* such that the value of the nonlinear objective function $F(x^*)$ is a minimum.
Hereafter, the solution of the problem will be indicated with x^*, the current guess for the solution will be denoted as $\overline{x}$ and x will represent the new point for the solution. To extend the Newton method for root finding to the optimization problems, the objective function is approximated about the current position x by the first three terms of the Taylor series. This yields the approximation of the function in correspondence of the new point x, namely:

$$F(x) = F(\overline{x}) + F'(\overline{x})(x-\overline{x}) + \frac{1}{2}F''(\overline{x})(x-\overline{x})^2 \tag{2.1}$$

where F' and F'' indicate the first and the second derivatives of F computed in $\overline{x}$, respectively. The condition for x to be a minimum point for F is to ask that the derivative of F computed in x equals zero, i.e.

$$\frac{\partial F}{\partial x} = 0 \implies F'(\overline{x}) + F''(\overline{x})(x-\overline{x}) = 0 \tag{2.2}$$

Rearranging Equation (2.2), a new estimation of the solution can be obtained as

$$x = \overline{x} - \frac{F'(\overline{x})}{F''(\overline{x})} \tag{2.3}$$

The condition $F'(x) = 0$ only defines a stationary point, which can be a minimum, a maximum, or a point of inflection. The missing information to discern between the type of stationary point is represented by the knowledge of the function curvature. Point x^* is a minimum point only if therein the function curves upwards, namely the second derivative F'' is positive. Hence, the conditions for x^* to represent an optimizer of the objective function F are:

[2] https://github.com/coin-or/Ipopt

- *necessary conditions*:

$$F'(x^*) = 0 \tag{2.4}$$
$$F''(x^*) \geq 0 \tag{2.5}$$

- *sufficient conditions*:

$$F'(x^*) = 0 \tag{2.6}$$
$$F''(x^*) > 0 \tag{2.7}$$

Sufficient conditions define a strong local minimizer, whereas necessary conditions define a weak local minimizer.

2.2.2 Unconstrained Optimization in n Variables

Considering the multidimensional extension of the case presented in Section 2.2.1, the aim is to find the solution vector $\mathbf{x}^* = [x_1^* \ \dots \ x_n^*]^T \in \mathbb{R}^n$ such that the scalar objective function $F : \mathbb{R}^n \to \mathbb{R}$ is at a minimum in $\mathbf{x}^*$. Like in the 1-dimensional case, F is approximated about point $\bar{\mathbf{x}}$ by means of the Taylor series up to the third term, namely:

$$F(\mathbf{x}) = F(\bar{\mathbf{x}}) + \mathbf{g}(\bar{\mathbf{x}})^T(\mathbf{x}-\bar{\mathbf{x}}) + \frac{1}{2}(\mathbf{x}-\bar{\mathbf{x}})^T\mathbf{H}(\mathbf{x})(\mathbf{x}-\bar{\mathbf{x}}) \tag{2.8}$$

where $\mathbf{g} \in \mathbb{R}^n$ is the gradient vector of F

$$\mathbf{g}(\mathbf{x}) = \nabla_{\mathbf{x}} F = \begin{bmatrix} \dfrac{\partial F}{\partial x_1} \\ \vdots \\ \dfrac{\partial F}{\partial x_n} \end{bmatrix} \tag{2.9}$$

and $\mathbf{H} \in \mathbb{R}^{n \times n}$ is the (symmetric) Hessian matrix

$$\mathbf{H}(\mathbf{x}) = \nabla^2_{\mathbf{x}} F = \begin{bmatrix} \dfrac{\partial^2 F}{\partial x_1^2} & \dfrac{\partial^2 F}{\partial x_1 \partial x_2} & \cdots & \dfrac{\partial^2 F}{\partial x_1 \partial x_n} \\ \dfrac{\partial^2 F}{\partial x_2 \partial x_1} & \dfrac{\partial^2 F}{\partial x_2^2} & \cdots & \dfrac{\partial^2 F}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 F}{\partial x_n \partial x_1} & \dfrac{\partial^2 F}{\partial x_n \partial x_2} & \cdots & \dfrac{\partial^2 F}{\partial x_n^2} \end{bmatrix} \tag{2.10}$$

Defining the *search direction* $\mathbf{p} = (\mathbf{x} - \bar{\mathbf{x}})$, Equation (2.8) can be re-written as

$$F(\mathbf{x}) = F(\bar{\mathbf{x}}) + \mathbf{g}^T(\bar{\mathbf{x}})\mathbf{p} + \frac{1}{2}\mathbf{p}^T\mathbf{H}(\bar{\mathbf{x}})\mathbf{p} \tag{2.11}$$

The scalar term $\mathbf{g}^T\mathbf{p}$ is the *directional derivative* along $\mathbf{p}$ and the scalar term $\mathbf{p}^T\mathbf{H}\mathbf{p}$ is the *curvature* or *second directional derivative* in the direction $\mathbf{p}$. Note that the gradient and the Hessian matrix are computed in correspondence of $\bar{\mathbf{x}}$.

By imposing that the derivative w.r.t. $\mathbf{x}$ of the function approximation (2.11) is equal to zero, the search direction $\mathbf{p}$ can be obtained, namely:

$$\frac{\partial F}{\partial \mathbf{x}} = 0 \implies \mathbf{g} + \mathbf{H}\mathbf{p} = \mathbf{0} \tag{2.12}$$

from which then new iteration can be computed as

$$\mathbf{x} = \bar{\mathbf{x}} - \mathbf{H}^{-1}(\bar{\mathbf{x}})\mathbf{g}(\bar{\mathbf{x}}) \tag{2.13}$$

If $\mathbf{x}^*$ is a local minimum, the value of the objective function in correspondence of its neighboring points must be larger, i.e. $F(\mathbf{x}) > F(\mathbf{x}^*)$. This implies that the slope of the function in all directions, computed in $\mathbf{x}^*$, has to be equal to zero, namely:

$$\mathbf{g}(\mathbf{x}^*) = \begin{bmatrix} g_1(\mathbf{x}^*) \\ \vdots \\ g_n(\mathbf{x}^*) \end{bmatrix} = \mathbf{0} \tag{2.14}$$

The local-minimum condition is satisfied if the function, computed in $\mathbf{x}^*$, curves upwards in all directions, hence requiring that the third term in Equation (2.11) is positive, i.e.

$$\mathbf{p}^T \mathbf{H}(\mathbf{x}^*)\mathbf{p} > 0 \tag{2.15}$$

which is granted if and only if the Hessian matrix is positive definite. If there are some directions with zero curvature, i.e. $\mathbf{p}^T \mathbf{H}(\mathbf{x}^*)\mathbf{p} \geq 0$, $\mathbf{H}(\mathbf{x}^*)$ is positive semidefinite.
To sum up, the following conditions for $\mathbf{x}^*$ to be a minimizer of F hold:

- *necessary conditions*:

$$\mathbf{g}(\mathbf{x}^*) = \mathbf{0} \tag{2.16}$$
$$\mathbf{p}^T \mathbf{H}(\mathbf{x}^*)\mathbf{p} \geq 0 \tag{2.17}$$

- *sufficient conditions*:

$$\mathbf{g}(\mathbf{x}^*) = \mathbf{0} \tag{2.18}$$
$$\mathbf{p}^T \mathbf{H}(\mathbf{x}^*)\mathbf{p} > 0 \tag{2.19}$$

2.2.3 Equality-Constrained Problem

Suppose that we want to find the vector $\mathbf{x}^* = [x_1^* \ \dots \ x_n^*]^T \in \mathbb{R}^n$ that satisfies the constraints

$$\mathbf{c}(\mathbf{x}^*) = \begin{bmatrix} c_1(\mathbf{x}^*) \\ \vdots \\ c_m(\mathbf{x}^*) \end{bmatrix} = \mathbf{0} \tag{2.20}$$

The linear approximation of the vector function $\mathbf{c} : \mathbb{R}^n \to \mathbb{R}^m$ around $\bar{\mathbf{x}}$ is

$$\mathbf{c}(\mathbf{x}) = \mathbf{c}(\bar{\mathbf{x}}) + \mathbf{G}(\bar{\mathbf{x}})(\mathbf{x} - \bar{\mathbf{x}}) \tag{2.21}$$

where $\mathbf{G} \in \mathbb{R}^{m \times n}$ is the Jacobian matrix

$$\mathbf{G}(\mathbf{x}) = \frac{\partial \mathbf{c}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial c_1}{\partial x_1} & \frac{\partial c_1}{\partial x_2} & \cdots & \frac{\partial c_1}{\partial x_n} \\ \frac{\partial c_2}{\partial x_1} & \frac{\partial c_2}{\partial x_2} & \cdots & \frac{\partial c_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial c_m}{\partial x_1} & \frac{\partial c_m}{\partial x_2} & \cdots & \frac{\partial c_m}{\partial x_n} \end{bmatrix} \tag{2.22}$$

The solution of the problem can be found by imposing that $\mathbf{c}(\mathbf{x}) = \mathbf{0}$ in Equation (2.21), thus obtaining the linear system

$$\mathbf{G}(\bar{\mathbf{x}})\mathbf{p} = -\mathbf{c}(\bar{\mathbf{x}}) \tag{2.23}$$

The resolution of the linear system in (2.23) provides the search direction $\mathbf{p}$, from which the new iteration for the solution can be computed as

$$\mathbf{x} = \bar{\mathbf{x}} + \mathbf{p} \tag{2.24}$$

Hence, each Newton iteration requires the linearization of the function $\mathbf{c}$ expressing the equality constraints. The n-dimensional Newton method for root finding shares the same properties of the simple 1-dimensional case. In particular, the method is quadratically convergent, but it may diverge if globalization strategies are not properly applied. In addition, the method requires the computation of matrix $\mathbf{G}$, which may be time consuming from a computational point of view.

2.2.4 Equality-Constrained Optimization

The Newton method can be applied to both optimize an objective function $F(\mathbf{x})$ (see Section 2.2.2) or satisfy a set of constraints $\mathbf{c}(\mathbf{x}) = \mathbf{0}$ (see Section 2.2.3). In equality-constrained optimization, the aim is to solve both problems simultaneously, that is to find $\mathbf{x}^* \in \mathbb{R}^n$ to minimize $F(\mathbf{x})$ subject to $m \leq n$ constraints $\mathbf{c}(\mathbf{x}) = \mathbf{0}$, namely:

$$\min_{\mathbf{x}} F(\mathbf{x}) \tag{2.25a}$$

subject to

$$\mathbf{c}(\mathbf{x}) = \mathbf{0} \tag{2.25b}$$

The classical approach is to define the Lagrangian of the problem as

$$L(\mathbf{x}, \boldsymbol{\lambda}) = F(\mathbf{x}) - \boldsymbol{\lambda}^T \mathbf{c}(\mathbf{x}) = F(\mathbf{x}) - \sum_{i=1}^{m} \lambda_i c_i(\mathbf{x}) \tag{2.26}$$

where $\boldsymbol{\lambda} \in \mathbb{R}^m$ represents the array of Lagrange multipliers. Optimality requires the vanishing of the derivatives of L w.r.t. $\mathbf{x}$ and $\boldsymbol{\lambda}$. As a consequence, the necessary conditions for $(\mathbf{x}^*, \boldsymbol{\lambda}^*)$ to represent an optimum are

$$\nabla_{\mathbf{x}} L(\mathbf{x}^*, \boldsymbol{\lambda}^*) = \mathbf{0} \tag{2.27a}$$

$$\nabla_{\boldsymbol{\lambda}} L(\mathbf{x}^*, \boldsymbol{\lambda}^*) = \mathbf{0} \tag{2.27b}$$

where the gradients of L w.r.t. $\mathbf{x}$ and $\boldsymbol{\lambda}$ are, respectively:

$$\nabla_{\mathbf{x}} L = \nabla_{\mathbf{x}} F - \sum_{i=1}^{m} \lambda_i \nabla_{\mathbf{x}} c_i = \mathbf{g} - \mathbf{G}^T \boldsymbol{\lambda} \tag{2.28a}$$

$$\nabla_{\boldsymbol{\lambda}} L = -\mathbf{c} \tag{2.28b}$$

Similarly to Section 2.2.2, the conditions (2.27a), (2.27b) do not allow distinction between a minimum and a maximum point. Additional conditions on the curvature of L are needed. In particular, we have to require that

$$\mathbf{v}^T \mathbf{H}_L(\mathbf{x}^*, \boldsymbol{\lambda}^*)\mathbf{v} > 0 \tag{2.29}$$

with $\mathbf{H}_L$ being the Hessian matrix of the Lagrangian, namely

$$\mathbf{H}_L = \nabla^2_{\mathbf{x}} L = \nabla^2_{\mathbf{x}} F - \sum_{i=1}^{m} \lambda_i \nabla^2_{\mathbf{x}} c_i \tag{2.30}$$

and $\mathbf{v}$ is an arbitrary vector belonging to the constraint tangent space[3]. The main difference between the curvature condition in the unconstrained case (2.15) and the analogous condition in the constrained problem (2.29) is that, in the former case, the curvature has to be positive along all directions $\mathbf{p}$, whereas in the latter case (2.29) applies only to directions $\mathbf{v}$ in the tangent constraint space.
The Newton method can be exploited to find the values of $(\mathbf{x}, \boldsymbol{\lambda})$ that satisfy the conditions on the Lagrangian gradients (see (2.27a, 2.27b)). For the sake of clarity, a vector function $\mathbf{S} : \mathbb{R}^{m+n} \rightarrow \mathbb{R}^{m+n}$ can be defined as composed of the two functions $\nabla_{\mathbf{x}} L$ and $\nabla_{\lambda} L$, namely:

$$\mathbf{S} = \begin{bmatrix} \mathbf{S}_1 \\ \mathbf{S}_2 \end{bmatrix} = \begin{bmatrix} \nabla_{\mathbf{x}} L \\ \nabla_{\boldsymbol{\lambda}} L \end{bmatrix} \tag{2.31}$$

To find the point $(\mathbf{x}, \boldsymbol{\lambda})$, the function $\mathbf{S}$ is approximated around $(\overline{\mathbf{x}}, \overline{\boldsymbol{\lambda}})$, i.e.:

$$\mathbf{S}(\mathbf{x}, \boldsymbol{\lambda}) = \mathbf{S}(\overline{\mathbf{x}}, \overline{\boldsymbol{\lambda}}) + \mathbf{S}'(\overline{\mathbf{x}}, \overline{\boldsymbol{\lambda}}) \begin{bmatrix} \mathbf{x} - \overline{\mathbf{x}} \\ \boldsymbol{\lambda} - \overline{\boldsymbol{\lambda}} \end{bmatrix} \tag{2.32}$$

with $\mathbf{S}'$ indicating the Jacobian matrix of $\mathbf{S}$ w.r.t. both $\mathbf{x}$ and $\boldsymbol{\lambda}$, namely

$$\frac{\partial \mathbf{S}}{\partial (\mathbf{x}, \boldsymbol{\lambda})} = \begin{bmatrix} \dfrac{\partial \mathbf{S}_1}{\partial \mathbf{x}} & \dfrac{\partial \mathbf{S}_1}{\partial \boldsymbol{\lambda}} \\ \dfrac{\partial \mathbf{S}_2}{\partial \mathbf{x}} & \dfrac{\partial \mathbf{S}_2}{\partial \boldsymbol{\lambda}} \end{bmatrix} \tag{2.33}$$

where the block matrices can be computed by employing the results of Equations (2.28) and (2.30), hence obtaining:

$$\frac{\partial \mathbf{S}_1}{\partial \mathbf{x}} = \frac{\partial}{\partial \mathbf{x}}(\nabla_{\mathbf{x}} L) = \mathbf{H}_L \qquad \frac{\partial \mathbf{S}_1}{\partial \boldsymbol{\lambda}} = \frac{\partial}{\partial \boldsymbol{\lambda}}(\nabla_{\mathbf{x}} L) = -\mathbf{G}^T \tag{2.34}$$

$$\frac{\partial \mathbf{S}_2}{\partial \mathbf{x}} = \frac{\partial}{\partial \mathbf{x}}(\nabla_{\lambda} L) = -\mathbf{G} \qquad \frac{\partial \mathbf{S}_2}{\partial \boldsymbol{\lambda}} = \frac{\partial}{\partial \boldsymbol{\lambda}}(\nabla_{\lambda} L) = \mathbf{0} \tag{2.35}$$

[3] A vector $\mathbf{v} \neq \mathbf{0}$ belongs to the tangent space of the contraints if $\mathbf{G}(\mathbf{x})\mathbf{v} = \mathbf{0}$.

Imposing that $\mathbf{S}(\mathbf{x}, \boldsymbol{\lambda}) = \mathbf{0}$ and substituting the expression of $\mathbf{S}'(\overline{\mathbf{x}}, \overline{\boldsymbol{\lambda}})$ in Equation (2.32), we obtain a linear system called *Karush-Kuhn-Tucker* (KKT) system

$$\begin{bmatrix} \mathbf{H}_L(\overline{\mathbf{x}}) & \mathbf{G}(\overline{\mathbf{x}})^T \\ \mathbf{G}(\overline{\mathbf{x}}) & \mathbf{0} \end{bmatrix} \begin{bmatrix} -\mathbf{p} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{g}(\overline{\mathbf{x}}) \\ \mathbf{c}(\overline{\mathbf{x}}) \end{bmatrix} \tag{2.36}$$

where $\mathbf{p}$ is the search direction for a step $\mathbf{x} = \overline{\mathbf{x}} + \mathbf{p}$ and $\boldsymbol{\lambda}$ is the vector of Lagrange multipliers at the new point.

2.2.5 Inequality-Constrained Optimization

Inequality-constrained optimization problems aim at minimizing an objective function F subject to inequality constraints. In formulas:

$$\min_{\mathbf{x}} F(\mathbf{x}) \tag{2.37a}$$

subject to

$$\mathbf{c}(\mathbf{x}) \geq \mathbf{0} \tag{2.37b}$$

While in equality-constrained optimization (see Section 2.2.4)), the number of constraints must be lower or equal to the number of variables, i.e. $m \leq n$, inequality-constrained optimization enables a number of constraints greater than the number of variables. A point that satisfies the inequality constraint is called *feasible* and the set of all feasible points is indicated as *feasible region*.
Regarding the fulfillment of the inequality constraints, two situations could occur in correspondence of the solution $\mathbf{x}^*$:

- some constraints may be satisfied as equalities, i.e. $c_i(\mathbf{x}^*) = 0 \; \forall i \in A$, where A is called *active set*;
- some constraints may be strictly satisfied as inequalities, i.e. $c_i(\mathbf{x}^*) > 0 \; \forall i \in A'$, where A' is called *inactive set*.

Active constraints, for which the inequalities are fulfilled as equalities, can be treated as described in Section 2.2.4. Algorithms employed to solve inequality-constrained optimizations need a strategy to identify the active constraints, which is referred to as *active set strategy*. In addition, when applying active set strategies, another necessary condition on the sign of the Lagrange multipliers must be respected, i.e.:

$$\lambda_i^* \geq 0 \quad \forall i \in A. \tag{2.38}$$

2.2.6 Quadratic Programming

A special case of constrained-optimization problems is represented by the *quadratic programming* (QP) case, in which the objective function is a quadratic form of $\mathbf{x}$ and all constraints are linear. The formulation of a QP problem is thus as follows:

$$\min_{\mathbf{x}} F(\mathbf{x}) = \mathbf{r}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{D}\mathbf{x} \tag{2.39a}$$

subject to

$$\mathbf{A}\mathbf{x} = \mathbf{a} \tag{2.39b}$$

$$\mathbf{B}\mathbf{x} \geq \mathbf{b} \tag{2.39c}$$

where $\mathbf{r}$ is a vector of constant weights and $\mathbf{D}$ is a constant symmetric matrix; $\mathbf{A}$ and $\mathbf{a}$ are the coefficient matrix and the known-variable vector of the linear equality constraints $\mathbf{A}\mathbf{x} - \mathbf{a} = \mathbf{0}$, respectively, whereas $\mathbf{B}$ and $\mathbf{b}$ play the same role of $\mathbf{A}$ and $\mathbf{a}$, considering the linear inequality constraints $\mathbf{B}\mathbf{x} - \mathbf{b} \geq \mathbf{0}$. A problem of this type can be tackled by adopting an active-set method. Assume that an estimate of the active set A_0 is given together with the initial feasible point $\bar{\mathbf{x}}_0$. An active-set QP algorithm proceeds according to the following steps:

A-1 Solve the KKT system (2.36), where the constraints in the active set A are treated as equalities.

A-2 Take the largest possible step in the direction $\mathbf{p}$ that does not violate any inactive inequalities, i.e. $\mathbf{x} = \bar{\mathbf{x}} + \alpha\mathbf{p}$, where $\alpha \in [0,1]$ is chosen to maintain feasibility w.r.t. inactive inequality constraints.

A-3 If the step is restricted, i.e. $\alpha < 1$, then add the limiting inequality to the active set A and return to step A-1.

A-4 Otherwise, take the full step ($\alpha = 1$) and check the sign of the Lagrangian multipliers.

- A-4.1 If all inequalities have positive multipliers terminate the algorithm.
- A-4.2 Otherwise, delete the inequality with the most negative λ from the active set and return to A-1.

Step A-1 requires the resolution of the KKT system. To this aim, the formulation described in Section 2.2.4 can be exploited, by considering the following Lagrangian:

$$L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\eta}) = F(\mathbf{x}) - \boldsymbol{\lambda}^T(\mathbf{A}\mathbf{x} - \mathbf{a}) - \boldsymbol{\eta}^T(\tilde{\mathbf{B}}\mathbf{x} - \tilde{\mathbf{b}}) \tag{2.40}$$

where $\mathbf{A}$ and $\tilde{\mathbf{B}}$ represent the Jacobian matrices of the equalities and the active-inequality constraints. The Lagrange multipliers of the equalities are indicated with $\boldsymbol{\lambda}$, whereas the ones relative to the active set are indicated with $\boldsymbol{\eta}$. The gradient and the Hessian matrix of L w.r.t. $\mathbf{x}$ are given by:

$$\nabla_{\mathbf{x}} L = \mathbf{g} - \mathbf{A}^T\boldsymbol{\lambda} - \tilde{\mathbf{B}}\boldsymbol{\eta} = \mathbf{r} + \mathbf{D}\mathbf{x} - \mathbf{A}^T\boldsymbol{\lambda} - \tilde{\mathbf{B}}\boldsymbol{\eta} \tag{2.41a}$$

$$\mathbf{H} = \nabla^2_{\mathbf{x}} L = \mathbf{D} \tag{2.41b}$$

thus proving that the Hessian matrix for a QP problem is constant. Hence, the KKT system can be written as:

$$\begin{bmatrix} \mathbf{H} & \mathbf{A}^T & \tilde{\mathbf{B}}^T \\ \mathbf{A} & \mathbf{0} & \mathbf{0} \\ \tilde{\mathbf{B}} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} -\mathbf{p} \\ \boldsymbol{\lambda} \\ \boldsymbol{\eta} \end{bmatrix} = \begin{bmatrix} \mathbf{g}(\bar{\mathbf{x}}) \\ \mathbf{a} \\ \tilde{\mathbf{b}} \end{bmatrix} \tag{2.42}$$

It is worth noting that, as long as the initial active set coincides with the final one, the algorithm computes the solution in one iteration. On the other hand, the QP algorithm may require many iterations if the initial active set differ from the final one and

this may increase the computational time. Active-set QP algorithms represent the basis for *sequential quadratic programming* (SQP), in which the NLP problem (the constraints are not linear as in (2.39) but are represented by nonlinear functions of the varibale $\mathbf{x}$) is divided in subsequential QP problems. A more detailed presentation of active-set SQP algorithms is not object of this Thesis; however, it must be said that SQP methods represent an equivalent alternative to interior-point methods for the resolution of NLP problems.

2.3 Globalization Strategies

As far as the resolution of constrained optimization problems is concerned, the Newton method can be developed for the purpose of approximating the Lagrangian and finding the iterative solutions that satisfy the necessary and sufficient conditions. However, the Newton method may have some deficiencies, even for simple problems. To improve the behavior of the Newton method, the so-called *globalization strategies* are needed.

2.3.1 Merit Functions

If the method is working properly, the sequence of iterates $\mathbf{x}_k$ should converge to the solution $\mathbf{x}^*$. To measure the efficiency of the algorithm progress toward the solution, a merit function M can be defined and the following condition must hold, namely

$$M(\mathbf{x}_{k+1}) < M(\mathbf{x}_k) \tag{2.43}$$

In the case of unconstrained optimization (see Section 2.2.2), the merit function can be chosen as coincident with the objective function, i.e. $M(\mathbf{x}) = F(\mathbf{x})$. On the other hand, if the aim is to find the root of a vector function (see Section 2.2.3), the choice of the appropriate merit function may not be trivial. The most commonly used merit function for nonlinear equations is

$$M(\mathbf{x}) = \frac{1}{2}\mathbf{c}(\mathbf{x})^T\mathbf{c}(\mathbf{x}) \tag{2.44}$$

where $\mathbf{c}(\mathbf{x})$ represents the equality-constraint function computed at $\mathbf{x}$. Alternatively, different norms of the constraint function $\mathbf{c}$ can be employed:

$$M(\mathbf{x}) = ||\mathbf{c}||_1 = \sum_{i=1}^{m} |c_i| \tag{2.45}$$

$$M(\mathbf{x}) = ||\mathbf{c}||_2 = \sqrt{\sum_{i=1}^{m} c_i^2} \tag{2.46}$$

$$M(\mathbf{x}) = ||\mathbf{c}||_\infty = \max_{i=1}^{m} |c_i| \tag{2.47}$$

When constrained-optimization problems (see Section 2.2.4) have to be taken into account, the selection of the merit function becomes more complex, due to the presence of the conflicting goals represented by the minimization of the objective function and the fulfillment of the constraints. In this case, rather than employing a merit function giving information about the convergence progress, another type of function P can be

built. The function P has to be chosen such that its unconstrained minimum is the desired constrained solution $\mathbf{x}^*$ or it is related to $\mathbf{x}^*$ in a known way. This method will be indicated as *penalty function* method. One possible choice for P is

$$P(\mathbf{x}, \rho) = F(\mathbf{x}) + \frac{\rho}{2}\mathbf{c}(\mathbf{x})^T\mathbf{c}(\mathbf{x}) \tag{2.48}$$

with ρ being called *penalty weight* or *penalty parameter* and $\mathbf{c}(\mathbf{x})$ representing the equality-constraint function computed at $\mathbf{x}$. When this penalty function is minimized for successively larger values of ρ, it can be shown that the unconstrained minimizers approach the constrained solution. However, this implies computational inefficiency, due to the fact that, as $\rho \to \infty$, the successive unconstrained problems become difficult to solve. An alternative is the *augmented Lagrangian function* that is formulated as

$$P(\mathbf{x}, \boldsymbol{\lambda}, \rho) = L(\mathbf{x}, \boldsymbol{\lambda}) + \frac{\rho}{2}\mathbf{c}(\mathbf{x})^T\mathbf{c}(\mathbf{x}) \tag{2.49}$$

with $L(\mathbf{x}, \boldsymbol{\lambda})$ being the Lagrangian of the problem (see Equation 2.26). With this position, it can be proven that the unconstrained minimum is equal to $\mathbf{x}^*$ for a finite value of ρ. However, attention must be paid when the value of ρ is too large or too small, since, in these cases, the problem may be ill-conditioned.

2.3.2 Line-Search Methods

While a merit function (Section 2.3.1) is mainly employed as an indicator of the progress of the algorithm, an approach that can be used to alterate the search direction is represented by the *line-search* method. The basic idea is to modify the way the new iterate is computed by acting on the magnitude of the step $\mathbf{p}$, namely

$$\mathbf{x} = \bar{\mathbf{x}} + \alpha\mathbf{p} \tag{2.50}$$

where $0 \leq \alpha \leq 1$. This way, if at the k-th iteration the values of $\bar{\mathbf{x}}$ and $\mathbf{p}$ are fixed, an appropriate merit function can be written as a function of the single variable α

$$M(\mathbf{x}) = M(\bar{\mathbf{x}} + \alpha\mathbf{p}) = M(\alpha) \tag{2.51}$$

where α can be chosen so that $M(\alpha)$ is approximately minimized. In most algorithms implementing line search, the starting step is taken with $\alpha = 1$; then, estimations are computed until a steplength α_k is found satisfying a sufficient decrease condition based on the *Goldstein-Armijo* principle, that is

$$0 < -k_1\alpha_k M'(0) \leq M(0) - M(\alpha_k) \leq -k_2\alpha_k M'(0) \tag{2.52}$$

with $M'(0)$ indicating the direction derivative $(\nabla M(0))^T\mathbf{p}$ at $\alpha = 0$, and k_1, k_2 satisfying $0 < k_1 \leq k_2 < 1$.

2.3.3 Trust-Region Methods

Another technique that suitably modifies the search direction $\mathbf{p}$ is called *trust-region* method. Both the magnitude and the direction of $\mathbf{p}$ are conveniently adjusted. By fixing the current point $\mathbf{x}$, the aim is to find the value of $\mathbf{p}$ that minimizes

$$F(\mathbf{x} + \mathbf{p}) = F(\mathbf{x}) + \mathbf{g}^T\mathbf{p} + \frac{1}{2}\mathbf{p}^T\mathbf{H}\mathbf{p} \tag{2.53a}$$

subject to

$$\frac{1}{2}\mathbf{p}^T\mathbf{p} \leq \delta^2 \tag{2.53b}$$

We say that the prediction $F(\mathbf{x}+\mathbf{p})$ is trustable as long as the components of $\mathbf{p}$ lie within the region defined by the *trust radius* ρ. The Lagrangian of the problem expressed in (2.53) is

$$L_T(\mathbf{p},\tau) = F(\mathbf{x}) + \mathbf{g}^T\mathbf{p} + \frac{1}{2}\mathbf{p}^T\mathbf{H}\mathbf{p} - \tau\left(\delta^2 - \frac{1}{2}\mathbf{p}^T\mathbf{p}\right) \tag{2.54}$$

with τ representing the Lagrange multiplier associated with the trust-region inequality constraint. By writing the necessary condition, we obtain

$$\nabla_{\mathbf{p}} L_T(\mathbf{p},\tau) = \mathbf{g} + \mathbf{H}\mathbf{p} + \tau\mathbf{p} = \mathbf{g} + (\mathbf{H} + \tau\mathbf{I})\mathbf{p} = \mathbf{0} \tag{2.55}$$

If the trust-radius constraint is inactive (i.e. $\frac{1}{2}\mathbf{p}^T\mathbf{p} < \delta^2$), then $\tau = 0$ and the search direction is the unmodified Newton direction. Conversely, if the trust-radius constraint is active (i.e. $\frac{1}{2}\mathbf{p}^T\mathbf{p} = \delta^2$), the norm of $\mathbf{p}$ is $||\mathbf{p}|| = \sqrt{2}\delta$ and $\tau > 0$. The limit behavior of δ and τ can be summarized as follows

- $\tau \to \infty, \delta \to 0$: the search direction approaches the gradient direction $-\mathbf{g}/||\mathbf{g}||$ and its norm tends to zero $||\mathbf{p}|| \to 0$;
- $\tau \to 0, \delta \to \infty$: the search direction tends to $-\mathbf{H}^{-1}\mathbf{g}$.

Trust-region methods are often combined with other globalization strategies. For instance, it is a common practice to modify the trust radius from one iteration to the next one by comparing the values of a specific merit function in correspondence of the predicted and actual directions.

2.3.4 Filters

In order to evaluate if the algorithm is going towards the goals of constraint fulfillment and objective-function reduction, an ad-hoc filter can be established. The filter accepts the Newton step if the objective function or the constraint violation is decreasing [5]. This filtering approach recognizes the two conflicting aims in NLP, i.e., choose $\mathbf{x}$ to minimize $F(\mathbf{x})$ and choose $\mathbf{x}$ to minimize the constraint violation $v[\mathbf{c}(\mathbf{x})]$. The latter can be defined by considering any suitable norm of $\mathbf{c}$.
The basic idea is to compare information from the current iteration with information from previous iterates and then discard the bad iterates. By denoting the values of the objective function and the constraint violation in correspondence of point $\mathbf{x}_k$ as

$$\{F^{(k)}, v^{(k)}\} = \{F(\mathbf{x}_k), v[\mathbf{c}(\mathbf{x}_k)]\} \tag{2.56}$$

we will say that, considering two points $\mathbf{x}_k$ and $\mathbf{x}_j$, a couple $\{F^{(k)}, v^{(k)}\}$ dominates another pair $\{F^{(j)}, v^{(j)}\}$ if and only if both the following conditions hold

$$F^{(k)} \leq F^{(j)} \tag{2.57}$$

$$v^{(k)} \leq v^{(j)} \tag{2.58}$$

This way, a pair $\{F^{(l)}, v^{(l)}\}$ is included in the filter if it is not dominated by any other pair in the current filter.
The filter method only provides a way to accept or reject an iterate, hence requiring an additional strategy to correct the step if a point is rejected. Typically, filter methods are combined with a trust-region approach or a line-search technique.

2.4 Interior-Point Methods

2.4.1 Overview of the Method

The general constrained problem that has to be taken into account has the form:

$$\min_{\mathbf{x}} F(\mathbf{x}) \tag{2.59a}$$

subject to

$$\mathbf{c}(\mathbf{x}) = \mathbf{0} \tag{2.59b}$$
$$\mathbf{b}(\mathbf{x}) \geq \mathbf{0} \tag{2.59c}$$

where $\mathbf{x}^* = [x_1^* \ \dots \ x_n^*]^T \in \mathbb{R}^n$ is the vector of variables to be determined to minimize $F : \mathbb{R}^n \to \mathbb{R}$, simultaneously satisfying the equality constraints expressed by the vector function $\mathbf{c} : \mathbb{R}^n \to \mathbb{R}^{m_\mathscr{E}}$ and the inequalities formulated in the vector function $\mathbf{b} : \mathbb{R}^n \to \mathbb{R}^{m_\mathscr{I}}$, with $m_\mathscr{E}$ and $m_\mathscr{I}$ denoting the number of equalities $\mathscr{E}$ and inequalities $\mathscr{I}$, respectively.
Interior-point methods were first established in the 1960s, but only 30 years later, the promising results obtained in the linear-programming resolution renewed interest in them also for nonlinear optimization. The numerical results show that interior-point methods are often faster than active-set SQP methods on large problems, particularly when the number of free variables is large. For convenience reasons, the problem at hand is re-formulated in the following fashion:

$$\min_{\mathbf{x},\mathbf{s}} F(\mathbf{x}) \tag{2.60a}$$

subject to

$$\mathbf{c}(\mathbf{x}) = \mathbf{0} \tag{2.60b}$$
$$\mathbf{b}(\mathbf{x}) - \mathbf{s} = \mathbf{0} \tag{2.60c}$$
$$\mathbf{s} \geq \mathbf{0} \tag{2.60d}$$

where the vector $\mathbf{x}$ and the functions $F, \mathbf{c}, \mathbf{b}$ have the same meaning as described in (2.59), with the difference that the inequalities $\mathbf{b}(\mathbf{x}) \geq \mathbf{0}$ have been translated to equalities with the introduction of a vector $\mathbf{s}$ of slack variables.
The interior-point method can be interpreted in two ways. Regarding the first approach, one can write the KKT conditions for the nonlinear problem, namely:

$$\nabla F - \mathbf{G}_\mathscr{E}^T \boldsymbol{\lambda} - \mathbf{G}_\mathscr{I}^T \boldsymbol{\nu} = \mathbf{0} \tag{2.61a}$$
$$\mathbf{S}\boldsymbol{\nu} - \mu\mathbf{e} = \mathbf{0} \tag{2.61b}$$
$$\mathbf{c}(\mathbf{x}) = \mathbf{0} \tag{2.61c}$$
$$\mathbf{b}(\mathbf{x}) - \mathbf{s} = \mathbf{0} \tag{2.61d}$$

having denoted the Jacobian matrices of $\mathbf{c}$ and $\mathbf{b}$ w.r.t. $\mathbf{x}$ with $\mathbf{G}_\mathscr{E}$ and $\mathbf{G}_\mathscr{I}$, respectively; matrix $\mathbf{S}$ is the diagonal matrix composed of the slack-vector components, i.e. $\mathbf{S} = \text{diag}(\mathbf{s})$ and $\mathbf{e} = [1 \ \dots \ 1]^T \in \mathbb{R}^{m_\mathscr{I}}$. The described approach consists of solving the perturbed KKT conditions for a sequence of positive parameters μ_k that converges to zero, while maintaining $\mathbf{s}, \boldsymbol{\nu} \geq \mathbf{0}$. By doing so, a point that satisfies KKT conditions can be reached. Finally, by requiring the iterates to decrease according to a merit function

or to be accepted by an appropriate filter, the iteration tends to a problem minimizer. Hence, for all sufficiently small positive values of μ, the KKT conditions have a locally unique solution $\big(\mathbf{x}(\mu), \mathbf{s}(\mu), \boldsymbol{\lambda}(\mu), \boldsymbol{\nu}(\mu)\big)$, that converges to the optimizer $(\mathbf{x}^*, \mathbf{s}^*, \boldsymbol{\lambda}^*, \boldsymbol{\nu}^*)$ as $\mu \to 0$. The trajectory described by the points $\big(\mathbf{x}(\mu), \mathbf{s}(\mu), \boldsymbol{\lambda}(\mu), \boldsymbol{\nu}(\mu)\big)$ towards the solution is called *primal-dual central path*.
The second interpretation of interior-point methods foresees the combination of the problem with the barrier approach, namely:

$$\underset{\mathbf{x},\mathbf{s}}{min}\, F(\mathbf{x}) - \mu \sum_{i=1}^{m_{\mathscr{I}}} \log s_i \tag{2.62a}$$

subject to

$$\mathbf{c}(\mathbf{x}) = \mathbf{0} \tag{2.62b}$$

$$\mathbf{b}(\mathbf{x}) - \mathbf{s} = \mathbf{0} \tag{2.62c}$$

where μ is a positive parameter and $\log(\cdot)$ indicates the natural logarithm function. The condition $\mathbf{s} \geq \mathbf{0}$ is not included, due to the fact that minimizing the logarithmic term $-\mu \sum_{i=1}^{m_{\mathscr{I}}} \log s_i$ prevents the components of $\mathbf{s}$ from becoming too close to zero[4]. The solution to the barrier problem does not coincide with the one sought for the general problem (2.59). The barrier approach aims at finding approximate solutions to the barrier problem for a sequence of positive barrier parameters μ_k that converges to zero and then verify if they represent an optimizer for the overall main problem (2.59) with a certain tolerance. If the KKT conditions for the problem (2.62) are written, we obtain:

$$\nabla F - \mathbf{G}_{\mathscr{E}}^T \boldsymbol{\lambda} - \mathbf{G}_{\mathscr{I}}^T \boldsymbol{\nu} = \mathbf{0} \tag{2.63a}$$

$$-\mu \mathbf{S}^{-1}\mathbf{e} + \boldsymbol{\nu} = \mathbf{0} \tag{2.63b}$$

$$\mathbf{c}(\mathbf{x}) = \mathbf{0} \tag{2.63c}$$

$$\mathbf{b}(\mathbf{x}) - \mathbf{s} = \mathbf{0} \tag{2.63d}$$

The second condition (2.63b) differs from the second condition (2.61b) of the corresponding primal-dual version. However, we can re-arrange Equation (2.63) by multiplying all the terms by the diagonal matrix $\mathbf{S}$, hence obtaining that the KKT conditions for the barrier problem 2.63 coincide with the perturbed KKT system 2.61.
The barrier approach is the reason behind the name *interior point*. Indeed, early barrier methods [6] do not use slack variable and start from an initial point $\mathbf{x}_0$ feasible w.r.t. the inequalities $\mathbf{b}(\mathbf{x}) \geq \mathbf{0}$; by employing a barrier function, the methods prevent the iterates from leaving the feasible region, hence always lying internally.
Applying the Newton method to the nonlinear system (2.61), we obtain the so-called *primal-dual system*:

$$\begin{bmatrix} \mathbf{H}_L(\mathbf{x}_k) & \mathbf{0} & \mathbf{G}_{\mathscr{E}}(\mathbf{x}_k)^T & \mathbf{G}_{\mathscr{I}}(\mathbf{x}_k)^T \\ \mathbf{0} & \text{diag}(\boldsymbol{\nu}_k) & \mathbf{0} & \mathbf{S}_k \\ \mathbf{G}_{\mathscr{E}}(\mathbf{x}_k) & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{G}_{\mathscr{I}}(\mathbf{x}_k) & -\mathbf{I} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{p}_k^x \\ \mathbf{p}_k^s \\ \mathbf{p}_k^\lambda \\ \mathbf{p}_k^\nu \end{bmatrix} = \begin{bmatrix} \nabla F(\mathbf{x}_k) - \mathbf{G}_{\mathscr{E}}(\mathbf{x}_k)^T \boldsymbol{\lambda}_k - \mathbf{G}_{\mathscr{I}}(\mathbf{x}_k)^T \boldsymbol{\nu}_k \\ \mathbf{S}_k \boldsymbol{\nu}_k - \mu_k \mathbf{e} \\ \mathbf{c}(\mathbf{x}_k) \\ \mathbf{b}(\mathbf{x}_k) - \mathbf{s}_k \end{bmatrix} \tag{2.64}$$

[4] Remember that $(-\log t) \to \infty$ as $t \to 0$.

where $\mathbf{H}_L$ is the Hessian matrix corresponding to the Lagrangian $L(\mathbf{x},\mathbf{s},\boldsymbol{\lambda},\boldsymbol{\nu}) = F(\mathbf{x}) - \boldsymbol{\lambda}^T\mathbf{c}(\mathbf{x}) - \boldsymbol{\nu}^T(\mathbf{b}(\mathbf{x}) - \mathbf{s})$. The resolution of the primal-dual system (2.64) allows the determination of the search directions $(\mathbf{p}_k^x, \mathbf{p}_k^s, \mathbf{p}_k^\lambda, \mathbf{p}_k^\nu)$, through which the new iterates can be computed, namely:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_s^{max}\mathbf{p}_k^x \tag{2.65a}$$
$$\mathbf{s}_{k+1} = \mathbf{s}_k + \alpha_s^{max}\mathbf{p}_k^s \tag{2.65b}$$
$$\boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k + \alpha_\nu^{max}\mathbf{p}_k^\lambda \tag{2.65c}$$
$$\boldsymbol{\nu}_{k+1} = \boldsymbol{\nu}_k + \alpha_\nu^{max}\mathbf{p}_k^\nu \tag{2.65d}$$

where the step lengths α_s^{max} and α_ν^{max} are employed to prevent variables $\mathbf{s}$ and ν from reaching their lower bounds of $\mathbf{0}$ too quickly, and are computed thanks to the following expressions:

$$\alpha_s^{max} = \max\{\alpha \in]0,1] : \mathbf{s}_k + \alpha\mathbf{p}_s \geq (1-\tau)\mathbf{s}_k\} \tag{2.66a}$$
$$\alpha_\nu^{max} = \max\{\alpha \in]0,1] : \boldsymbol{\nu}_k + \alpha\mathbf{p}_\nu \geq (1-\tau)\boldsymbol{\nu}_k\} \tag{2.66b}$$

The described iterations are the basis of an interior-point method, together with the choice of the sequence of the barrier parameters μ_k. Some techniques [6] keep the barrier parameter constant for a series of iterations until the KKT conditions are satisfied according to a certain tolerance. Alternatively, the barrier parameter is updated at each iteration. A way to verify if the KKT conditions are respected is by defining the error function, based on the perturbed KKT system:

$$E_\mu(\mathbf{x},\mathbf{s},\boldsymbol{\lambda},\boldsymbol{\nu}) = \max\{||\nabla F - \mathbf{G}_{\mathscr{E}}^T\boldsymbol{\lambda} - \mathbf{G}_{\mathscr{I}}^T\boldsymbol{\nu}||, ||\mathbf{S}\boldsymbol{\nu} - \mu\mathbf{e}||, ||\mathbf{c}(\mathbf{x})||, ||\mathbf{b}(\mathbf{x}) - \mathbf{s}||\} \tag{2.67}$$

An example of a basic interior-point algorithm is depicted in the procedure described below:

A-1 *Initialize.* Initialize the counter $k = 0$ and $\sigma \in]0,1]$.

A-2 *Check convergence of the overall problem.* Repeat the outer loop until an optimality-error condition is satisfied to stop the algorithm.

- A-2.1 *Check convergence of the barrier problem.* Repeat the inner loop until $E_{\mu_k}(\mathbf{x}_k,\mathbf{s}_k,\boldsymbol{\lambda}_k,\boldsymbol{\nu}_k) \leq \mu_k$.
  - A-2.1.1 *Compute the search direction.* Solve (2.64) to find $(\mathbf{p}_k^x, \mathbf{p}_k^s, \mathbf{p}_k^\lambda, \mathbf{p}_k^\nu)$.
  - A-2.1.2 *Compute the step lengths.* Compute α_s^{max} and α_ν^{max} using (2.66).
  - A-2.1.3 *Compute the next iterate.* Compute $(\mathbf{x}_{k+1}, \mathbf{x}_{k+1}, \boldsymbol{\lambda}_{k+1}, \mathbf{s}_{k+1})$ using (2.65).
  - A-2.1.4 *Increase the counter.* Set $k = k+1$ and $\mu_k = \mu_{k+1}$.
- A-2.2 *End inner loop.*
- A-2.3 *Choose the barrier parameter.* Choose $\mu_k \in]0, \sigma\mu_k[$.

A-3 *End the outer loop.*

2.4.2 IPOPT Algorithm

As briefly seen in Section 2.4.1, interior-point methods need to be combined with globalization strategies, in order to define a decreasing logic of the barrier parameter, together with a way to assess the progress of the iterations toward the solution and decide whenever the algorithm has to stop. In [4], the implementation of a filter line-search strategy integrated within an interior-point method is proposed. The algorithm is then written in the IPOPT code. Here, we provide a summary of the main features of IPOPT. To simplify the following formulations, the tackled NLP problem is set in the form:

$$\min_{\mathbf{x}} F(\mathbf{x}) \tag{2.68a}$$

subject to

$$\mathbf{c}(\mathbf{x}) = \mathbf{0} \tag{2.68b}$$

$$\mathbf{x} \geq \mathbf{0} \tag{2.68c}$$

The extension of the technique for the general case can be found in [4]. The equivalent barrier problem of (2.68) can be considered, i.e.:

$$\min_{\mathbf{x}} \quad \phi_\mu(\mathbf{x}) = F(\mathbf{x}) - \mu \sum_{i=1}^{n} \log x_i \tag{2.69a}$$

subject to

$$\mathbf{c}(\mathbf{x}) = \mathbf{0} \tag{2.69b}$$

The Lagrangian of the original problem (2.68) is

$$L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) = F(\mathbf{x}) - \boldsymbol{\lambda}^T \mathbf{c} - \boldsymbol{\nu} \tag{2.70}$$

where $\boldsymbol{\lambda}$ and $\boldsymbol{\nu}$ represent the Lagrangian multipliers for the equality constraints (2.68b) and the inequalities (2.68c), respectively. Hence, the KKT conditions can be derived, namely

$$\nabla F - \mathbf{G}^T \boldsymbol{\lambda} - \boldsymbol{\nu} = \mathbf{0} \tag{2.71a}$$

$$\mathbf{c}(\mathbf{x}) = \mathbf{0} \tag{2.71b}$$

$$\mathbf{X}\,\mathrm{diag}(\boldsymbol{\nu})\mathbf{e} - \mu\mathbf{e} = \mathbf{0} \tag{2.71c}$$

The primal-dual Equations (2.71), for $\mu = 0$ together with $\mathbf{x}, \boldsymbol{\nu} \geq \mathbf{0}$, coincide with the KKT conditions of the original problem (2.68). The proposed method computes an approximate solution of the barrier problem (2.69) for a fixed value of μ, then decreases the barrier parameter and continues the solution of the next barrier problem (characterized by a lower value of μ), starting from the approximate solution of the previous barrier problem. The procedure ends when some conditions are respected regarding the achievement of the problem optimizer.
Exploiting the primal-dual Equations (2.71) the optimality error for the barrier problem can be defined as

$$E_\mu(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) = \max\left\{ \frac{||\nabla F - \mathbf{G}^T \boldsymbol{\lambda} - \boldsymbol{\nu}||_\infty}{s_d}, \frac{||\mathbf{S}\boldsymbol{\nu} - \mu\mathbf{e}||_\infty}{s_c}, ||\mathbf{c}(\mathbf{x})||_\infty \right\} \tag{2.72}$$

with $s_d, s_c \geq 1$ being scaling parameters. If the multipliers $\boldsymbol{\lambda}$ and $\boldsymbol{\nu}$ become very large and the algorithm may encounter numerical issues satisfying the unscaled primal-dual Equations (2.71). To handle these circumstances, the termination criteria can be

adapted by including the aforementioned scaling parameters, whose expressions can be suitably chosen [4]. By definition of E_μ, the optimality error for the original problem is given by Equation (2.72) with $\mu = 0$ and is indicated as $E_0(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu})$. The overall algorithm terminates if the point $(\mathbf{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\nu}^*)$ satisfies the condition

$$E_0(\mathbf{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\nu}^*) \leq \epsilon_{tol} \tag{2.73}$$

with $\epsilon_{tol} > 0$ being chosen by the user. Considering the iterations employed to find the approximate solution of the barrier problem, the barrier parameter μ_j is decreased step by step, where j indicates the iteration counter. The next approximate solution for the barrier problem has to respect the condition

$$E_{\mu_j}(\hat{\mathbf{x}}^*, \hat{\boldsymbol{\lambda}}^*, \hat{\boldsymbol{\nu}}^*) \leq k_\epsilon \mu_j \tag{2.74}$$

where the symbol $\wedge$ on the variables is employed to indicate a solution of the barrier problem, whereas $k_\epsilon > 0$. The achievement of fast local convergence to a local solution of the main problem (2.68) satisfying the second-order sufficient optimality conditions is granted by following the strategy 2 described in [7], where the new value of μ_{j+1} is computed as:

$$\mu_{j+1} = \max\left\{\frac{\epsilon_{tol}}{10}, \min\left\{k_\mu \mu_j, \mu_j^{\theta_\mu}\right\}\right\} \tag{2.75}$$

where the values of k_μ and θ_μ belong to the intervals $k_\mu \in]0,1[$ and $\theta_\mu \in]1,2[$. This way, the computation of μ_{j+1} does not allow the barrier parameter to become smaller than necessary given the desired value of the tolerance ϵ_{tol}, in order to avoid numerical issues at the end of the optimization.
To solve the barrier problem for a fixed value of μ_j the linearization of the primal-dual equations (2.71) can be performed by extending what was explained in Section (2.2.4). In particular, by approximating the functions on the left-hand side of Equations (2.71) in Taylor series around the new point $(\mathbf{x}_{k+1}, \boldsymbol{\lambda}_{k+1}, \boldsymbol{\nu}_{k+1})$ and imposing their zeroing, we can write the linear system

$$\begin{bmatrix} \mathbf{H}_L(\mathbf{x}_k) & -\mathbf{G}^T(\mathbf{x}_k) & -\mathbf{I} \\ -\mathbf{G}(\mathbf{x}_k) & \mathbf{0} & \mathbf{0} \\ \text{diag}(\boldsymbol{\nu}_k) & \mathbf{0} & \text{diag}(\mathbf{x}_k) \end{bmatrix} \begin{bmatrix} \mathbf{p}_k^x \\ \mathbf{p}_k^\lambda \\ \mathbf{p}_k^\nu \end{bmatrix} = \begin{bmatrix} \nabla F(\mathbf{x}_k) - \mathbf{G}^T(\mathbf{x}_k)\boldsymbol{\lambda}_k - \boldsymbol{\nu}_k \\ \mathbf{c}(\mathbf{x}_k) \\ \text{diag}(\mathbf{x}_k)\text{diag}(\boldsymbol{\nu}_k) - \mu_j \mathbf{e} \end{bmatrix} \tag{2.76}$$

where the vector of unknowns represents the search directions $\mathbf{p}_k^x = \mathbf{x}_{k+1} - \mathbf{x}_k$, $\mathbf{p}_k^\lambda = \boldsymbol{\lambda}_{k+1} - \boldsymbol{\lambda}_k$, $\mathbf{p}_k^\nu = \boldsymbol{\nu}_{k+1} - \boldsymbol{\nu}_k$, from which the next iterates can be compute as

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k^x \tag{2.77}$$

$$\boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k + \alpha_k \mathbf{p}_k^\lambda \tag{2.78}$$

$$\boldsymbol{\nu}_{k+1} = \boldsymbol{\nu}_k + \alpha_k^\nu \mathbf{p}_k^\nu \tag{2.79}$$

where α_k^ν is given by

$$\alpha_k^\nu = \max\{\alpha \in]0,1] : \boldsymbol{\nu}_k + \alpha \mathbf{p}_k^\nu \geq (1-\tau_j)\boldsymbol{\nu}_k\}, \tag{2.80}$$

whereas the value of α_k is chosen among the interval $]0, \alpha_k^{max}]$ by a backtracking line-search procedure that employs a decreasing sequence of trial step sizes $\alpha_{k,l} = 2^{-l}\alpha_k^{max}$ $(l = 0, 1, 2, \ldots)$, to ensure global convergence [5], [8]. The value of α_k^{max} is given by

$$\alpha_k^{max} = \max\{\alpha \in]0,1] : \mathbf{x}_k + \alpha \mathbf{p}_k^x \geq (1-\tau_j)\mathbf{x}_k\} \tag{2.81}$$

The value of τ_j present in Equations (2.80), (2.81) is given by

$$\tau_j = \max\{\tau_{min}, 1-\mu_j\} \quad \text{with } \tau_{min} \in]0,1[\tag{2.82}$$

The line-search filter method used in the algorithm to solve the barrier problem for μ_j accepts the trial point $\mathbf{x}_k(\alpha_{k,l}) = \mathbf{x}_k + \alpha_{k,l}\mathbf{p}_k^x$ if it leads to sufficient progress toward the goal of minimizing the barrier function $\phi_{\mu_j}(\mathbf{x})$ and the constraint violation $\theta(\mathbf{x}) = ||\mathbf{c}(\mathbf{x})||$, i.e., if

$$\phi_{\mu_j}(\mathbf{x}_k(\alpha_{k,l})) \leq \phi_{\mu_j}(\mathbf{x}_k) - \gamma_\phi\theta(\mathbf{x}_k) \tag{2.83a}$$

$$\theta(\mathbf{x}_k(\alpha_{k,l})) \leq (1-\gamma_\theta)\theta(\mathbf{x}_k) \tag{2.83b}$$

with $\gamma_\theta, \gamma_\phi \in]0,1[$. If, in correspondence of the iterate $\mathbf{x}_k$, it results $\theta(\mathbf{x}_k) \leq \theta^{min}$, for an arbitrary $\theta^{min} \in]0,\infty[$ and the following conditions hold

$$\nabla\phi_{\mu_j}(\mathbf{x}_k)^T\mathbf{p}_k^x < 0 \quad \cap \quad \alpha_{k,l}[-\nabla\phi_{\mu_j}(\mathbf{x}_k)^T\mathbf{p}_k^x]^{s_\phi} > \delta[\theta(\mathbf{x}_k)]^{s_\theta} \tag{2.84}$$

with $\delta > 0$, $s_\theta > 1$ and $s_\phi \geq 1$, the trial point has to satisfy the Armijo condition

$$\phi_{\mu_j}(\mathbf{x}_k(\alpha_{k,l})) \leq \phi_{\mu_j}(\mathbf{x}_k) + \eta_\phi\alpha_{k,l}\nabla\phi_{\mu_j}(\mathbf{x}_k)^T\mathbf{p}_k^x \tag{2.85}$$

instead of 2.83, to be accepted; the value of η_ϕ is chosen among the interval $]0, \frac{1}{2}[$.
In addition, another filter is operated at iteration k. This filter, indicated with $\mathscr{F}_k$, contains the combination of constraint violation and objective function values that are prohibited. Hence, during the line search, a trial point $\mathbf{x}_k(\alpha_{k,l})$ is rejected if the couple $\big(\phi_{\mu_j}(\mathbf{x}_k(\alpha_{k,l})), \theta(\mathbf{x}_k(\alpha_{k,l}))\big) \in \mathscr{F}_k$. When the optimization starts, the filter is initialized to

$$\mathscr{F}_0 = \{(\phi,\theta) \in \mathbb{R}^2 : \theta \geq \theta^{max}\} \tag{2.86}$$

for some θ^{max}, to ensure that the algorithm will never allow points for which the constraint violation is larger than θ^{max}. As the algorithm evolves, the filter is augmented using the formula

$$\mathscr{F}_{k+1} = \mathscr{F}_k \cup \{(\phi,\theta) \in \mathbb{R}^2 : \phi \geq \phi_{\mu_j}(\mathbf{x}_k^\diamond) - \gamma_\phi\theta(\mathbf{x}_k^\diamond) \cap \theta \geq (1-\gamma_\theta)\theta(\mathbf{x}_k^\diamond)\} \tag{2.87}$$

where $\mathbf{x}_k^\diamond$ represents a point that does not satisfy specific conditions explained in detail in [4]. With the filter augmentation, the iterates are prevented from falling into the neighborhood of $\mathbf{x}_k^\diamond$. A *feasibility restoration phase* may be needed if it is not possible to find a trial step size $\alpha_{k,l}$ that satisfies the criteria of the filter. In such a scenario, linearization of the involved functions is performed to find a value of $\alpha_{k,l}$ that may be lower than a specified threshold [4]. If this case occurs, the algorithm attempts to find a new iterate $\mathbf{x}_{k+1}$ that can be accepted by the current filter and for which the conditions in (2.83) hold, but attention must be paid, since this restoration procedure might lead to the infeasibility of the problem.
Furthermore, it may happen that an iterate is rejected by the filter, even if it actually grants better progress toward the solution of the problem. This undesirable phenomenon is often called *Maratos effect* [2]. A way to avoid the Maratos effect is the *second-order correction* (SOC), in which the search direction is conveniently adjusted [9].
Once the basics of the algorithm implemented in IPOPT have been briefly described, its pseudocode can be depicted:

A-1 *Initialize.* Initialize the counters $j = 0$ and $k = 0$, as well as the filter $\mathscr{F}_0$ and τ_0 from 2.82.

A-2 *Check convergence of the overall problem.* If $E_0(\mathbf{x}_k, \boldsymbol{\lambda}_k, \boldsymbol{\nu}_k) \leq \epsilon_{tol}$, then stop the algorithm.

A-3 *Check convergence for the barrier problem.* If $E_{\mu_j}(\mathbf{x}_k, \boldsymbol{\lambda}_k, \boldsymbol{\nu}_k) \leq k_\epsilon \mu_j$, then:

A-3.1 Compute μ_{j+1} and τ_{j+1} and set $j = j + 1$.

A-3.2 Re-initialize the filter $\mathscr{F}_k = \{(\phi, \theta) \in \mathbb{R}^2 : \theta \geq \theta^{max}\}$.

A-3.3 If $k = 0$ repeat A-3; otherwise go to A-4.

A-4 *Compute the search direction.* Compute $(\mathbf{p}_k^x, \mathbf{p}_k^\lambda, \mathbf{p}_k^\nu)$ from resolution of (2.76).

A-5 *Perform the backtracking line search.*

A-5.1 *Initialize the line search.* Set $\alpha_{k,0} = \alpha_k^{max}$ and set $l = 0$.

A-5.2 *Compute the new trial point.* Set $\mathbf{x}_k(\alpha_{k,l}) = \mathbf{x}_k + \alpha_{k,l}\mathbf{p}_k^x$.

A-5.3 *Check acceptance to the filter.* If $\big(\phi_{\mu_j}(\mathbf{x}_k(\alpha_{k,l})), \theta(\mathbf{x}_k(\alpha_{k,l}))\big) \in \mathscr{F}_k$, reject the trial step and go to steo A-5.5.

A-5.4 *Check sufficient decrease with respect to the current iterate.*

- *Case I*: $\theta(\mathbf{x}_k) \leq \theta^{min}$ and (2.84) holds; if (2.85) holds, accept the trial step $\mathbf{x}_{k+1} := \mathbf{x}_k(\alpha_{k,l})$ and go to A-6. Otherwise, continue to A-5.5.
- *Case II*: $\theta(\mathbf{x}_k) > \theta^{min}$ or (2.84) is not satisfied; if (2.85) holds, accept the trial step $\mathbf{x}_{k+1} := \mathbf{x}_k(\alpha_{k,l})$ and go to A-6. Otherwise, continue to A-5.5.

A-5.5 *Perform the second-order correction.* Initialize and compute the second-order correction; apply the verifications A-5.3 and A-5.4 for the new SOC iterate.

A-5.6 *Choose the new trial step size.* Set $\alpha_{k,l+1} = \frac{1}{2}\alpha_{k,l}$ and $l = l + 1$. If the trial step becomes too small, i.e., $\alpha_{k,l} \leq \alpha_k^{min}$, go to the feasibility restoration phase A-9. Otherwise go back to A-5.2.

A-6 *Accept the trial point.* Set $\alpha_k = \alpha_{k,l}$ (or $\alpha_k = \alpha_k^{SOC}$ if the SOC point was selected in A-5.5) and update the multipliers $\boldsymbol{\lambda}_{k+1}$, $\boldsymbol{\nu}_{k+1}$ with α_k^ν.

A-7 *Augment the filter if necessary.* If (2.84) and (2.85) do not hold for α_k, augment the filter using (2.87). Otherwise, leave the filter unchanged, i.e., set $\mathscr{F}_{k+1} = \mathscr{F}_k$.

A-8 *Continue with the next iteration.* Increase the iteration counter $k = k + 1$ and go back to A-2.

A-9 *Perform the feasibility restoration phase.* Augment the filter using (2.87) and compute a new iterate $\mathbf{x}_{k+1} > \mathbf{0}$ by relaxing the infeasibility measure $\theta(\mathbf{x})$, so that $\mathbf{x}_{k+1}$ is accepted by the augmented filter, i.e., $\big(\phi_{\mu_j}(\mathbf{x}_{k+1}), \theta(\mathbf{x}_{k+1})\big) \notin \mathscr{F}_{k+1}$. Then continue re-starting from step A-8.

2.5 Investigation on Optimal Control Problems

In the time-optimal trajectory planning of serial robots, point-to-point or prescribed-path motions are typically considered [10]. In the former case, only the initial and final poses of the robot end-effector are fixed, whereas in the latter case, the path that the robot end-effector needs to follow is imposed. The path can be parameterized in terms of a path parameter s, whose motion law $s(t)$ and its time derivatives $\dot{s}(t)$, $\ddot{s}(t)$ describe the evolving in time of the path, i.e., the trajectory.
The aim of time-optimal trajectory planning is to search for the optimal control input $\mathbf{u}(t)$ that grants the trajectory execution in the minimum time (hence minimizing a cost functional), at the same time satisfying the constraints imposed not only on the control $\mathbf{u}(t)$ but eventually also on a state $\mathbf{x}(t)$ that explicitly describes the status of the system (e.g., the state $\mathbf{x}(t)$ can include the joint angles of the robot under exam). In most cases, the time evolution of the state $\mathbf{x}(t)$ is obtained by integrating the so-called dynamical system, that connects the state $\mathbf{x}(t)$ with the input $\mathbf{u}(t)$ through a system of *ordinary differential equations* (ODEs).
A problem of this type, composed of the minimization of a cost functional, the resolution of the dynamical system, and the fulfillment of some physically meaningful constraints goes under the name of *optimal control problem* (OCP), whose general formulation can be written as:

$$\min_{t_e,\mathbf{x}(t),\mathbf{u}(t)} \left[\int_0^{t_e} [1 + L(\mathbf{x}(t),\mathbf{u}(t))]\, dt \right] \tag{2.88a}$$

subject to

$$\dot{\mathbf{x}}(t) - \mathbf{f}(\mathbf{x}(t),\mathbf{u}(t)) = \mathbf{0} \tag{2.88b}$$

$$\mathbf{x}(0) - \bar{\mathbf{x}}_0 = \mathbf{0} \tag{2.88c}$$

$$\mathbf{g}(\mathbf{x}(t_e)) \leq \mathbf{0} \tag{2.88d}$$

$$\mathbf{h}(\mathbf{x}(t),\mathbf{u}(t)) \leq \mathbf{0}, \quad \forall\, t \in [0, t_e] \tag{2.88e}$$

The cost functional (2.88a) is a trade-off between minimal time ($\int_0^{t_e} dt$) and minimal integral cost ($\int_0^{t_e} L(\mathbf{x}(t),\mathbf{u}(t))dt$). The function $L(\mathbf{x}(t),\mathbf{u}(t))$ can be conveniently chosen and written by the user to minimize a specific quantity throughout the motion, depending on the application. The OCP is subject to initial and final conditions (2.88c) and (2.88d), respectively. Initial conditions are usually written in the form of equalities, representing the starting values attributed to the system state $\mathbf{x}$. In contrast, final conditions can be written as inequalities, in order to limit the system state $\mathbf{x}$ below a specified limit at the end of the motion, rather than constraining it to a precise value. While the aforementioned initial and final conditions represent discrete constraints, the dynamical system (2.88b) and the constraints in (2.88e) are continuous constraints that have to be fulfilled during the whole motion, i.e. $\forall\, t \in [0, t_e]$, thus resulting in an infinite-dimensional problem. To deal with this issue, the original OCP is discretized in order to obtain a finite-dimensional problem. Therefore, the time domain is divided into N itervals with $(N+1)$ knots, going from $t_0 = 0$ to $t_N = t_e$.

2.5.1 Single Shooting Method

The OCP presented in Section 2.5 can be re-written as:

$$\min_{t_e,\mathbf{u}(t)} \left[\int_0^{t_e} [1 + L(\mathbf{x}(t),\mathbf{u}(t))]\, dt \right] \tag{2.89a}$$

subject to

$$\mathbf{x}(t) = \mathbf{x}(0) + \int_0^{t_e} \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))\, dt \tag{2.89b}$$

$$\mathbf{g}(\mathbf{x}(t_e)) \leq \mathbf{0} \tag{2.89c}$$

$$\mathbf{h}(\mathbf{x}(t), \mathbf{u}(t)) \leq \mathbf{0}, \quad \forall\, t \in [0, t_e] \tag{2.89d}$$

By virtue of the constraint (2.89b), that allows to express the state $\mathbf{x}(t)$ as a function of $\mathbf{u}(t)$, the considered problem is only influenced by the control $\mathbf{u}(t)$. This is the reason why, in the cost functional (2.89a), the quantity $\mathbf{x}(t)$ has been omitted (compared with the cost functional in (2.88a)). As a consequence, the OCP is subject to the dynamical system which is included in (2.89b) through forward time integration with initial condition $\mathbf{x}(0)$, to the final condition (2.89c) and to the continuous constraints (2.89d) that must be respected $\forall\, t \in [0, t_e]$.
The discretization of the problem is achieved by splitting the continuity of $\mathbf{u}(t)$ in $(N+1)$ variables stored inside the discrete matrix $\mathbf{w_u}$

$$\mathbf{w_u} = [\mathbf{u}_0 \quad \mathbf{u}_1 \quad \dots \quad \mathbf{u}_N] \tag{2.90}$$

This way, the discretization of the OCP can be formulated as:

$$\underset{t_e, \mathbf{w_u}}{min} \Big[\int_0^{t_e} [1 + L(\mathbf{x}(t), \mathbf{u}(t))]\, dt \Big] \tag{2.91a}$$

subject to

$$\mathbf{x}(t) = \mathbf{x}(0) + \int_0^{t_e} \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))\, dt \tag{2.91b}$$

$$\mathbf{g}(\mathbf{x}_N) \leq \mathbf{0} \tag{2.91c}$$

$$\mathbf{h}(\mathbf{x}_k, \mathbf{u}_k) \leq \mathbf{0}, \quad k = 0, \dots, N \tag{2.91d}$$

The time domain is discretized in N time intervals with a constant time step $\Delta t = t_e / N$. The discretized cost functional (2.91a) is composed of the final time t_e (to be minimized) and the integral cost which is computed through a suitable numerical integration (e.g. Runge-Kutta method), as well as the dynamical system. The final conditions (2.91c) are checked in correspondence of $\mathbf{x}_N$, whereas the constraints (2.91d) are verified at the checkpoints $(\mathbf{x}_k, \mathbf{u}_k)$, rather than being examined along the whole time domain $[0, T]$ (2.89d).
The main advantage of single shooting methods lies in the small number of variables that characterize the problem regarding the resolution of the dynamical system. It can be shown that the number of iteration variables is equal to the number of differential equations. However, when the differential equations are either nonlinear or stiff (or both), the single shooting method suffers from large propagation errors that may compromise the fulfillment of the final conditions. To reduce the sensitivity affecting single shooting methods, an alternative is to solve the problem considering shorter time intervals (see Section 2.5.2).

2.5.2 Multiple Shooting Method

In multiple shooting methods, the problem is divided into shorter steps, in order "not to shoot too far". The optimization variables are represented by both the control and

the state. In particular, the discretized trends of $\mathbf{u}$ and $\mathbf{x}$ can be stored inside the matrices:

$$\mathbf{w_u} = [\mathbf{u}_0 \quad \mathbf{u}_1 \quad \dots \quad \mathbf{u}_N] \tag{2.92a}$$

$$\mathbf{w_x} = [\mathbf{x}_0 \quad \mathbf{x}_1 \quad \dots \quad \mathbf{x}_N] \tag{2.92b}$$

This implies the storage of the problem inside the discrete matrix $\mathbf{w}$

$$\mathbf{w} = \begin{bmatrix} \mathbf{w_u} \\ \mathbf{w_x} \end{bmatrix} \tag{2.93}$$

The OCP can be re-written in finite-dimensional form, i.e.:

$$\min_{t_e, \mathbf{w}} \sum_{k=0}^{N-1} \left[\int_{t_k}^{t_{k+1}} [1 + L(\mathbf{x}(t), \mathbf{u}(t))] \, dt \right] \tag{2.94a}$$

subject to

$$\mathbf{x}_{k+1} = \int_{t_k}^{t_{k+1}} \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) dt \quad k = 0, \dots, N-1 \tag{2.94b}$$

$$\mathbf{x}_0 - \overline{\mathbf{x}}_0 = \mathbf{0} \tag{2.94c}$$

$$\mathbf{g}(\mathbf{x}_N) \leq \mathbf{0} \tag{2.94d}$$

$$\mathbf{h}(\mathbf{x}_k, \mathbf{u}_k) \leq \mathbf{0}, \quad k = 0, \dots, N \tag{2.94e}$$

The cost functional (2.94a) is reformulated to reflect the fact that the state, being an optimization variable belonging to $\mathbf{w}$ (see Equation (2.93)), is split into intervals. The dynamical system is solved by integration along the single time intervals $[t_k, t_{k+1}]$ with $k = 0, \dots, N-1$, rather than over the entire time domain.
The partition of the time domain into smaller time intervals aids the numerical integration of the dynamical system, also avoiding large propagation errors that may occur for longer time intervals. As a consequence of the multiple shooting method, the size of the problem increases w.r.t. the case in which a single shooting method is adopted. In particular, the number of NLP variables and constraints, characterizing the discretization of the dynamical system, for a multiple shooting application is equal to $n_{\mathbf{x}}N$, where $n_{\mathbf{x}}$ is the dimension of the state vector $\mathbf{x}$. Indeed, the constraints corresponding to the resolution of the dynamical system 2.94b are necessary to grant the continuity between the shooting procedure performed along the time steps $[t_k, t_{k+1}]$ and can be written in the form:

$$\mathbf{c}(\mathbf{w_x}) = \begin{bmatrix} \mathbf{x}_1 - \hat{\mathbf{x}}_1 \\ \mathbf{x}_2 - \hat{\mathbf{x}}_2 \\ \vdots \\ \mathbf{x}_N - \hat{\mathbf{x}}_N \end{bmatrix} = \begin{bmatrix} \mathbf{c}_1(\mathbf{w_x}) \\ \mathbf{c}_2(\mathbf{w_x}) \\ \vdots \\ \mathbf{c}_N(\mathbf{w_x}) \end{bmatrix} = \mathbf{0} \tag{2.95}$$

where $\mathbf{c} : \mathbb{R}^{n_{\mathbf{x}}N} \rightarrow \mathbb{R}^{n_{\mathbf{x}}N}$ is the function expressing the closing-gap equalities. Note that from the 2-nd to the N-th row of $\mathbf{c}$, $\hat{\mathbf{x}}_1, \dots \hat{\mathbf{x}}_N$ are the quantities obtained by numerical integration along the time-steps, i.e.

$$\hat{\mathbf{x}}_{k+1} = \int_{t_k}^{t_{k+1}} \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) dt, \quad k = 0, \dots, N-1 \tag{2.96}$$

The linearization of the equality constraints (see Section 2.2.3) requires the computation of the Jacobian matrix $\frac{\partial \mathbf{c}}{\partial \mathbf{w_x}} \in \mathbb{R}^{n_\mathbf{x} N \times n_\mathbf{x} N}$, namely

$$\frac{\partial \mathbf{c}}{\partial \mathbf{w_x}} = \begin{bmatrix} \frac{\partial \mathbf{c}_1}{\partial x_1} & \frac{\partial \mathbf{c}_1}{\partial x_2} & \cdots & \frac{\partial \mathbf{c}_1}{\partial x_N} \\ \frac{\partial \mathbf{c}_2}{\partial x_1} & \frac{\partial \mathbf{c}_2}{\partial x_2} & \cdots & \frac{\partial \mathbf{c}_2}{\partial x_N} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \mathbf{c}_N}{\partial x_1} & \frac{\partial \mathbf{c}_N}{\partial x_2} & \cdots & \frac{\partial \mathbf{c}_N}{\partial x_n} \end{bmatrix} \tag{2.97}$$

in which it can be noticed that the sub-matrices $\frac{\partial \mathbf{c}_j}{\partial x_l} = \mathbf{0}$ for $j \neq l$. The fact that the constraints influencing the k-th time interval do not depend on the variables of the $(k-1)$-th one grants the sparsity of the Jacobian matrix related to the nonlinear problem. In particular, among the $(n_\mathbf{x} N)^2$ elements of the Jacobian matrix, $n_\mathbf{x}^2 N$ are nonzero. Thus, the percentage of nonzeros is proportional to $1/N$, indicating that the problem becomes sparser as the number of time intervals N grows. Furthermore, the independency between intervals concerning the time integration allows parallel computation of all integrations, hence reducing the computational time.

2.6 Conclusions

In this Chapter, the main ingredients for the numerical resolution of OCPs were introduced, by giving an overview of the interior-point method and a multiple-shooting technique. Before explaining the main features of the optimization algorithm used in this Thesis, the fundamentals concerning NLP problems were provided. In particular, the first and second-order conditions for the optimality of NLP problems were illustrated. The discussion also focused on the use of particular globalization strategies aimed at giving information about the progress of the iterations toward the solution or about the logic used to change the search direction during the algorithm advance. A general procedure for the interior-point method was introduced, in which the main meaning of the method is to maintain the iterates inside the feasible region, never letting them approach the boundaries. The IPOPT algorithm, based on a filter line-search interior-point method, was presented, highlighting the main steps characterizing it. In the end, the multiple-shooting method adopted to convert the time continuity of the variables into a set of discretized variables was described. The obtained discretized variables constitute the variables to be optimized, in order to minimize an objective function, at the same time satisfying the initial and final constraints and the closing-gap constraints needed to reproduce the continuous evolution in time of the variables.

Chapter 3

Anti-Sloshing Motions of Serial Robots

This Chapter studies time-optimal anti-sloshing motions performed by serial robots. In particular, a novel technique, based on the discrete equivalent mass-spring-damper model, for sloshing-height estimation is presented. The model is validated by performing experiments with an industrial robot carrying a liquid-filled cylindrical container and following 2-dimensional planar paths. The extension of the formulation to 3-dimensional translational motions is proposed, and a validation campaign is carried out for this case, too. Taking into account 2-dimensional paths, a constrained-optimization problem is solved to impose limits on the sloshing height during task execution. A comparison between the non-optimized motions and the optimized ones is provided.
The work presented in this Chapter is published in [11], [12], [13].

3.1 Motivation

The transport of containers filled with liquids finds application in several industrial scenarios, e.g. in food&beverage or pharmaceutical production and packaging lines. Typically, the manipulation of such containers is assigned to linear transport systems or industrial serial robots; in many cases the required motion follows planar curves. The prediction of the liquid movement inside the container, referred to as *sloshing*, is important to prevent the liquid from overflowing. In the automotive context, the fuel movement during the launch of a spacecraft or during the cornering or breaking of vehicles, can be studied to reduce the inertia actions that may arise in such situations. For instance, the vehicle motions can be conveniently reproduced through the trajectories performed by an industrial robot following 3-dimensional paths.
The aforementioned applications justify the need for a reliable sloshing prediction model, not only for assessment purposes, but also to limit the stirring of the liquid during task execution. The latter aspect can be pursued through the offline resolution of a constrained-optimization problem.
For assessment purposes, machine-learning methodologies are presented in [14] and [15], where, starting from data collection, predictive algorithms are built to inspect the behavior of discrete liquid particles inside a cylindrical container. This technique, though very powerful, requires experiments to be run in advance, together with a not negligible computational effort.
In [16] and [17], the *Finite Element Method* (FEM) is adopted for the analysis of sloshing in rectangular containers, requiring a preliminary generation of the mesh able to replicate the liquid behavior.

The mesh-less *Smooth Particle Hydrodynamics* (SPH) method is employed in [18] and [19] to model the sloshing by discretizing the liquid in tens of thousands particles: the simulations accurately match the experimental data, at a cost of days of computation. In [20] the coefficients of the nonlinear sloshing dynamics model presented in [21] are provided to evaluate the sloshing height for 3-dimensional motions, leading to a complex formulation, which may be difficult to use.
A ready-to-use and fast alternative is represented by the development of equivalent discrete mechanical models. The literature considers two main discrete approaches for the modelling of sloshing dynamics inside a container subjected to 2-dimensional planar motion [22]: a spherical pendulum and a 2-DOF (*degree of freedom*) mass-spring-damper system. In the former case, the generalized coordinates describing the system are the angles defining the position of the pendulum mass, whereas, in the latter one, they are the mass displacements from the reference position. Although being intuitive, the use of the angular coordinates of the pendulum mass to assess the sloshing behavior of the liquid (see [23, 24]) lacks physical meaning, in particular when the knowledge of the liquid peak height is important. For this reason, in the spherical pendulum model used in [25], [26] and [27], the sloshing height is estimated by means of the tangent functions of the spherical coordinates. However, estimating the sloshing height by means of the tangent of the pendulum angles may lead to singularity conditions, when the container acceleration is high, since in this case these angles can approach 90° and the tangent tends to assume unrealistic high values.
To overcome this drawback, a novel approach, based on the mass-spring-damper model [28], is proposed in [29] for the sloshing-height estimation. This model is validated for 1-dimensional motions in [29] and the possible extension to 2-dimensional planar motions is presented, without providing an experimental validation. The latter is the objective of this Chapter, particularly referring to 2-dimensional planar motions of a cylindrical container, with accelerations up to 9.5 m/s^2 [11]. In addition, this Chapter reports an extension of the formulation to 3-dimensional motions comprising a vertical acceleration up to 5 m/s^2, also performing experiments to validate it [13].
As far as the study of anti-sloshing motion laws [30] is concerned, an appropriate filter can be designed to counteract the sloshing effect by re-orienting the container moved by a serial robot, hence exploiting all the manipulator degrees of freedom [25–27].
In [31], a 1-DOF-pendulum model is considered and an input shaper is used to suppress sloshing for a 1-dimensional excitation of the container, whereas in [23] the same approach is extended to 3-dimensional motions, by using a spherical pendulum model. Input shaping introduces a delay in the motion duration and the liquid free-surface peaks at the beginning of motion cannot be bound to remain below a specific value.
In [24], the formulation of a constrained-optimization problem taking into account a 3-dimensional motion is presented, but only 1-dimensional motion experiments are shown. The adopted model is again the spherical pendulum, and sloshing is limited by re-orienting the robot end-effector.
This research studies the time-optimal trajectory planning of an industrial robot carrying a cylindrical container filled with liquid. The objective is to impose stringent limits on the liquid sloshing prescribing 1-dimensional and 2-dimensional paths of the end-effector on a horizontal plane, without changing the end-effector orientation. The sloshing dynamics is modelled by using the mass-spring-damper model and the technique described in [29]. For the resolution of the constrained-optimization problem, different approaches are available. Pre-validated trajectory profiles may be used [32],

but this way no control on the imposed sloshing height is ensured. Alternatively, the velocity limit curve can be defined, but this requires the problem constraints to be explicitly written in terms of the path parameter s and its time derivatives $\dot{s}, \ddot{s}$ [33], which, for the problem addressed in this work, may not be easily fulfilled. For this reason, we solve the optimization problem by adopting a multiple shooting method and using CasADi [3], a software framework implemented in Matlab for nonlinear optimization and optimal control.
The Chapter is structured as follows. Section 3.2 presents the model parameters and the *equations of motion* (EOMs) in terms of the corresponding generalized coordinates. Section 3.3 provides the formulation of the sloshing-height estimation up to a 3-dimensional motion of the liquid-filled container. In Section 3.4, the constrained-optimization problem is described, showing how the sloshing constraints are written as functions of the path parameter and its time derivatives. Section 3.5 illustrates the experimental setup, together with an explanation of the experimental video post-processing; the validation campaign is described and discussed thanks to a quantitative analysis; furthermore, the experimental results comparing non-optimized and optimized motions are presented. Finally, in Section 3.6 conclusions are drawn and suggestions for future developments are given.

3.2 Sloshing Model

3.2.1 Model Parameters

We will consider a cylindrical container of radius R, filled with a liquid of height h and mass m_F. A simplified discrete mechanical model can be used to reproduce the liquid-sloshing dynamics. In particular, the mass-spring-damper model comprises a rigid mass m_0 (whose signed vertical distance from the liquid's center of gravity G is h_0) that moves rigidly with the container, and a series of moving masses m_n, with each one of them representing the equivalent mass of a sloshing mode (Figure 3.1a). Each modal mass m_n is restrained by a spring k_n and a damper c_n, and its signed vertical distance from G is h_n.
The model parameters can be determined by imposing a number of equivalence conditions with the original system [22]:

- the overall mass must be the same:

$$m_F = m_0 + \sum_{n=1}^{\infty} m_n \tag{3.1}$$

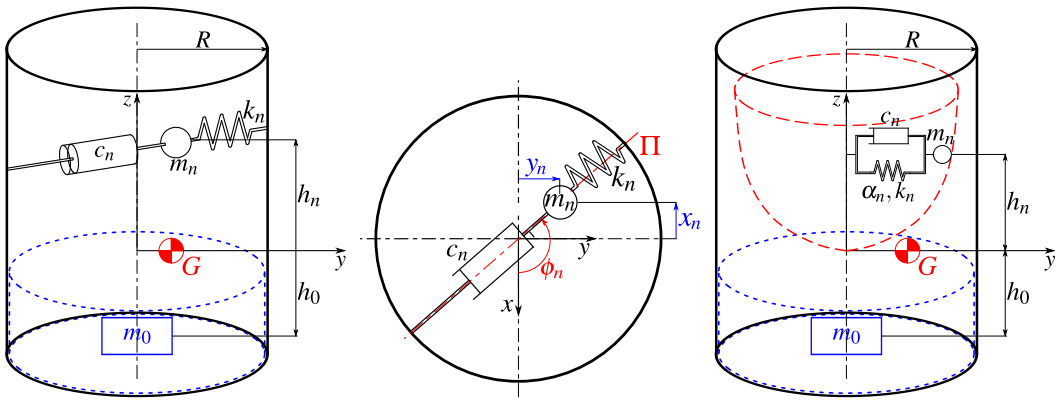
- the height of the center of gravity G must remain the same for small oscillations of the liquid:

$$m_0 h_0 + \sum_{n=1}^{\infty} m_n h_n = 0 \tag{3.2}$$

- the natural frequency associated with the n-th mode must coincide with the one that can be obtained from the continuum model:

$$\omega_n^2 = \frac{k_n}{m_n} = g\frac{\xi_{1n}}{R}\tanh\left(\xi_{1n}\frac{h}{R}\right)^1 \tag{3.3}$$

[1] This result is obtained from the formulation of the fluid field equations described in [22]. If a cylin-



(a) Model Parameters. (b) Top view showing the n-th generalized coordinates. (c) Sloshing mass sliding on a parabolic surface.

Figure 3.1: Mass-spring-damper model.

- the sloshing force acting on the container wall must be the same as the one calculated from the continuum model, leading to the determination of the n-th sloshing mass:

$$m_n = m_F \frac{2R}{\xi_{1n} h(\xi_{1n}^2 - 1)} \tanh\left(\xi_{1n} \frac{h}{R}\right). \tag{3.4}$$

In Equations (3.3) and (3.4), ξ_{1n} is the root of the derivative of the Bessel function of the first kind with respect to the radial coordinate r, for the 1^{st} circumferential mode and the n-th radial mode [34], while g is the gravity acceleration. The damping ratio $\zeta_n = \dfrac{c_n}{2\sqrt{k_n m_n}}$ can be determined by using empirical formulas [22]. In this Thesis, we will use:

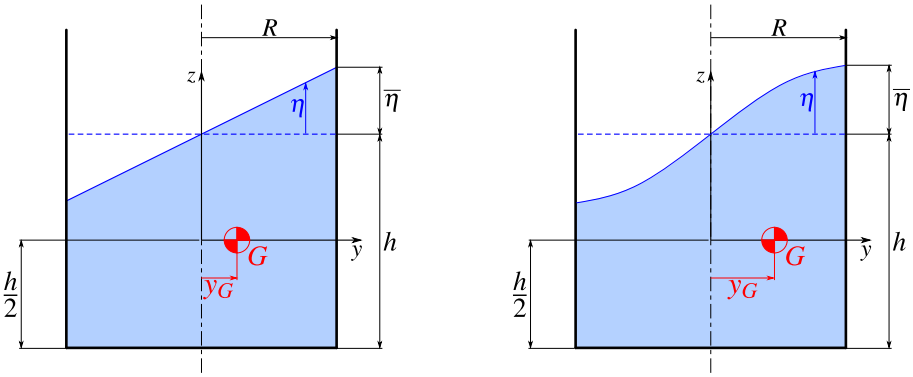
$$\zeta_n = 0.92 \sqrt{\frac{\upsilon/\rho}{\sqrt{gR^3}}} \left[1 + \frac{0.318}{\sinh(\xi_{1n} h/R)} \left(1 + \frac{1 - h/R}{\cosh(\xi_{1n} h/R)}\right)\right] \tag{3.5}$$

with υ and ρ being the dynamic viscosity and density of the liquid, respectively. In the case of an industrial robot carrying a container filled with liquid, the container is typically mounted on a tray which is attached to the robot end-effector. When the latter motion has no angular velocity, the container and the robot end-effector share the same translational acceleration, indicated with $\ddot{\mathbf{r}}_E$. For a container under 2-dimensional motion on the horizontal xy plane, the excitation is provided by the container accelerations along the x and y directions, denoted with $\{\ddot{\mathbf{r}}_E\}_0 = [\ddot{r}_x \;\; \ddot{r}_y \;\; 0]^T$. Here $\ddot{r}_x$ and $\ddot{r}_y$ indicate the projections of $\ddot{\mathbf{r}}_E$ on the x and y axes of the inertial frame F_0, respectively. The motion of the n-th sloshing mass is described by the generalized coordinates (x_n, y_n), whose definition is illustrated in Figure 3.1b. The latter are then used to compute the liquid sloshing height.

3.2.2 Equations of Motion

In general, three dynamic regimes are possible [22]:

drical container is considered and only the 1^{st} circumferential mode is taken into account, substituting the solution of the Laplace equation inside the free-surface boundary condition of the liquid yields the expression of natural frequency for the n-th radial mode.



(a) Assumption of planar free surface.

(b) Assumption of non-planar free surface.

Figure 3.2: Liquid free-surface shapes.

- small oscillations in which the liquid free surface remains planar (Figure 3.2a);
- relatively-large-amplitude oscillations in which the liquid free surface is no longer planar (Figure 3.2b);
- strongly nonlinear motion, where the liquid free surface exhibits instantaneous peaks characterized by swirling shapes.

While the third motion regime will not be object of the present study, the first and second cases can be analyzed by means of a *linear mass-spring-damper model* (L model) and a *nonlinear mass-spring-damper model* (NL model), respectively.
The NL model considers the sloshing mass m_n as sliding on a parabolic surface, with an attached nonlinear spring of order w (Figure 3.1c) [28]. The analytical expression of the parabolic surface allows the writing of the vertical coordinate z_n as a function of x_n and y_n, namely:

$$z_n = \frac{C_n}{2R}(x_n^2 + y_n^2) \tag{3.6}$$

where $C_n = \omega_n^2 \frac{R}{g}$. The time derivative of Equation (3.6) yields:

$$\dot{z}_n = \frac{C_n}{R}(\dot{x}_n x_n + \dot{y}_n y_n) \tag{3.7}$$

The nonlinear spring exerts the forces $\alpha_n k_n x_n^{2w-1}$ and $\alpha_n k_n y_n^{2w-1}$, along the x and y direction respectively. In this paper, we choose $w = 2$ and $\alpha_n = 0.58$, as suggested in [28]. If the radial generalized coordinate $r_n = \sqrt{x_n^2 + y_n^2}$ is introduced, the nonlinear-spring force in the radial direction can be written as $\alpha_n k_n r_n^{2w-1}$.
The EOMs, describing the time evolution of the generalized coordinates (x_n, y_n), can be obtained by means of the Lagrange Equations:

$$\begin{cases} \dfrac{d}{dt}\left(\dfrac{\partial T}{\partial \dot{x}_n}\right) - \dfrac{\partial T}{\partial x_n} + \dfrac{\partial V}{\partial x_n} = -\dfrac{\partial D}{\partial \dot{x}_n} \\ \dfrac{d}{dt}\left(\dfrac{\partial T}{\partial \dot{y}_n}\right) - \dfrac{\partial T}{\partial y_n} + \dfrac{\partial V}{\partial y_n} = -\dfrac{\partial D}{\partial \dot{y}_n} \end{cases} \tag{3.8}$$

where:

- the kinetic energy T of the n-th sloshing mass can be computed by taking into account its velocity $\dot{\mathbf{r}}_n = [\dot{x}_n \;\; \dot{y}_n \;\; \dot{z}_n]^T$, the container velocity $\{\dot{\mathbf{r}}_E\}_0 = [\dot{r}_x \;\; \dot{r}_y \;\; 0]^T$ and by exploiting Equation (3.7):

$$\begin{aligned} T &= \frac{1}{2} m_n [(\dot{r}_x + \dot{x}_n)^2 + (\dot{r}_y + \dot{y}_n)^2 + \dot{z}_n^2] = \\ &= \frac{1}{2} m_n \left[(\dot{r}_x + \dot{x}_n)^2 + (\dot{r}_y + \dot{y}_n)^2 + \frac{C_n^2}{R^2} (\dot{x}_n x_n + \dot{y}_n y_n)^2 \right] \end{aligned} \tag{3.9}$$

- the potential energy V considers the contribution of gravity and nonlinear-spring forces, namely:

$$V = m_n g z_n + \int_0^{r_n} \alpha_n k_n r_n^{2w-1} dr_n = m_n g \frac{C_n}{2R} (x_n^2 + y_n^2) + \frac{\alpha_n k_n}{2w} (x_n^2 + y_n^2)^w \tag{3.10}$$

- the Rayleigh function D accounts for energy dissipation:

$$D = \frac{1}{2} c_n (\dot{x}_n^2 + \dot{y}_n^2 + \dot{z}_n^2) = m_n \zeta_n \omega_n \left[\dot{x}_n^2 + \dot{y}_n^2 + \frac{C_n^2}{R^2} (\dot{x}_n x_n + \dot{y}_n y_n)^2 \right]. \tag{3.11}$$

The substitution of Equations (3.9), (3.10) and (3.11) in the system (3.8) leads to the formulation of two coupled EOMs for the NL model:

$$\begin{cases} \ddot{\overline{x}}_n + 2\omega_n \zeta_n [\dot{\overline{x}}_n + C_n^2 (\overline{x}_n^2 \dot{\overline{x}}_n + \overline{y}_n \dot{\overline{y}}_n \overline{x}_n)] + \\ \qquad + C_n^2 (\overline{x}_n \dot{\overline{x}}_n^2 + \overline{x}_n^2 \ddot{\overline{x}}_n + \overline{x}_n \dot{\overline{y}}_n^2 + \overline{x}_n \ddot{\overline{y}}_n \overline{y}_n) + \\ \qquad\qquad + \omega_n^2 \overline{x}_n [1 + \alpha_n (\overline{x}_n^2 + \overline{y}_n^2)^{w-1}] + \dfrac{\ddot{r}_x}{R} = 0 \\ \ddot{\overline{y}}_n + 2\omega_n \zeta_n [\dot{\overline{y}}_n + C_n^2 (\overline{y}_n^2 \dot{\overline{y}}_n + \overline{x}_n \dot{\overline{x}}_n \overline{y}_n)] + \\ \qquad + C_n^2 (\overline{y}_n \dot{\overline{y}}_n^2 + \overline{y}_n^2 \ddot{\overline{y}}_n + \overline{y}_n \dot{\overline{x}}_n^2 + \overline{y}_n \ddot{\overline{x}}_n \overline{x}_n) + \\ \qquad\qquad + \omega_n^2 \overline{y}_n [1 + \alpha_n (\overline{x}_n^2 + \overline{y}_n^2)^{w-1}] + \dfrac{\ddot{r}_y}{R} = 0 \end{cases} \tag{3.12}$$

where $\overline{x}_n = x_n / R$, $\overline{y}_n = y_n / R$. As far as the L model is concerned, the linearization of the EOMs in Equation (3.12) provides two decoupled EOMs in the generalized coordinates (x_n, y_n) of the n-th mode:

$$\begin{cases} \ddot{x}_n + 2\zeta_n \omega_n \dot{x}_n + \omega_n^2 x_n + \ddot{r}_x = 0 \\ \ddot{y}_n + 2\zeta_n \omega_n \dot{y}_n + \omega_n^2 y_n + \ddot{r}_y = 0 \end{cases} \tag{3.13}$$

3.2.3 Extension to 3-Dimensional Motion

If an excitation $\ddot{r}_z$ along the z-axis of F_0 is added to the one on the xy plane, the container is subject to a 3-dimensional motion, namely $\{\ddot{\mathbf{r}}_E\}_0 = [\ddot{r}_x \;\; \ddot{r}_y \;\; \ddot{r}_z]^T$. As long as we assume to employ the mass-spring-damper model presented in Section 3.2.1 to reproduce the liquid behavior, the same model parameters can be used. This choice allows the derivation of a fast and easy model extension, without the complication inherent in the construction of a different discrete model. As a consequence, we assume that

the additional motion along the z-axis only influences the kinetic energy of the n-th sloshing mass:

$$\begin{aligned} T &= \frac{1}{2} m_n [(\dot{r}_x + \dot{x}_n)^2 + (\dot{r}_y + \dot{y}_n)^2 + (\dot{r}_z + \dot{z}_n)^2] = \\ &= \frac{1}{2} m_n \left\{ (\dot{r}_x + \dot{x}_n)^2 + (\dot{r}_y + \dot{y}_n)^2 + \left[\dot{r}_z + \frac{C_n}{R} (\dot{x}_n x_n + \dot{y}_n y_n) \right]^2 \right\} \end{aligned} \tag{3.14}$$

where $\dot{z}_n$ is still given by Equation (3.7). Hence, combining Equation (3.8) with Equations (3.14), (3.10) and (3.11), the NL-model EOMs for the 3-dimensional motion become:

$$\begin{cases} \ddot{\overline{x}}_n + 2\omega_n \zeta_n [\dot{\overline{x}}_n + C_n^2 (\overline{x}_n^2 \dot{\overline{x}}_n + \overline{y}_n \dot{\overline{y}}_n \overline{x}_n)] + \\ \quad + C_n^2 (\overline{x}_n \dot{\overline{x}}_n^2 + \overline{x}_n^2 \ddot{\overline{x}}_n + \overline{x}_n \dot{\overline{y}}_n^2 + \overline{x}_n \ddot{\overline{y}}_n \overline{y}_n) + \\ \quad\quad + \omega_n^2 \overline{x}_n [1 + \alpha_n (\overline{x}_n^2 + \overline{y}_n^2)^{w-1}] + \dfrac{\ddot{r}_x}{R} + \dfrac{\ddot{r}_z}{g} \omega_n^2 \overline{x}_n = 0 \\ \ddot{\overline{y}}_n + 2\omega_n \zeta_n [\dot{\overline{y}}_n + C_n^2 (\overline{y}_n^2 \dot{\overline{y}}_n + \overline{x}_n \dot{\overline{x}}_n \overline{y}_n)] + \\ \quad + C_n^2 (\overline{y}_n \dot{\overline{y}}_n^2 + \overline{y}_n^2 \ddot{\overline{y}}_n + \overline{y}_n \dot{\overline{x}}_n^2 + \overline{y}_n \ddot{\overline{x}}_n \overline{x}_n) + \\ \quad\quad + \omega_n^2 \overline{y}_n [1 + \alpha_n (\overline{x}_n^2 + \overline{y}_n^2)^{w-1}] + \dfrac{\ddot{r}_y}{R} + \dfrac{\ddot{r}_z}{g} \omega_n^2 \overline{y}_n = 0 \end{cases} \tag{3.15}$$

whereas, the L-model EOMs yield:

$$\begin{cases} \ddot{x}_n + 2\zeta_n \omega_n \dot{x}_n + \omega_n^2 x_n + \ddot{r}_x + \dfrac{\ddot{r}_z}{g} \omega_n^2 x_n = 0 \\ \ddot{y}_n + 2\zeta_n \omega_n \dot{y}_n + \omega_n^2 y_n + \ddot{r}_y + \dfrac{\ddot{r}_z}{g} \omega_n^2 y_n = 0 \end{cases} \tag{3.16}$$

3.3 Analytical Sloshing-Height Estimation

3.3.1 1-Dimensional Motion

If only a 1-dimensional excitation in the y direction is given to the container and the phenomenon of rotary sloshing is neglected [22], solely the generalized coordinate y_n is different from zero. In such a case, the conservation of the center of gravity y-coordinate, between the continuum model and the equivalent model, yields:

$$y_G m_F = \sum_{n=1}^{\infty} y_n m_n + y_0 m_0 = \sum_{n=1}^{\infty} y_n m_n \tag{3.17}$$

Considering a cylindrical container with cross section $S = \pi R^2$, filled with a liquid of height h, y_G can be computed as:

$$\begin{aligned} y_G &= \frac{1}{Sh} \iint_S \int_{-\frac{h}{2}}^{\frac{h}{2} + \eta(r,\theta,\overline{\eta}_n)} y \, dz dS = \\ &= \frac{1}{\pi R^2 h} \int_0^R \int_0^{2\pi} \int_{-\frac{h}{2}}^{\frac{h}{2} + \eta(r,\theta,\overline{\eta}_n)} r^2 \sin\theta \, dz d\theta dr, \end{aligned} \tag{3.18}$$

where the function $\eta(r,\theta,\overline{\eta}_n)$ describes the liquid free-surface shape, $\overline{\eta}_n$ is the sloshing height of the n-th mode, (r,θ) are the polar coordinates, with $x = r\cos\theta$, $y = r\sin\theta$, $dS = r\ d\theta dr$. As for the L model, the function $\eta(r,\theta,\overline{\eta}_n)$ describes a plane (Figure 3.2a):

$$\eta(r,\theta,\overline{\eta}_n) = \sum_{n=1}^{\infty} \overline{\eta}_n \frac{r}{R}\sin\theta, \tag{3.19}$$

whereas, for the NL model, the non-planar free surface can be described by means of the first-kind Bessel function (Figure 3.2b), namely:

$$\eta(r,\theta,\overline{\eta}_n) = \sum_{n=1}^{\infty} \overline{\eta}_n \frac{J_1(\xi_{1n}\frac{r}{R})}{J_1(\xi_{1n})}\sin\theta. \tag{3.20}$$

Independently from the adopted function η, the expression of y_G from Equation (3.18) can be used in Equation (3.17) to express $\overline{\eta}_n$ as a function of the model parameters and the generalized coordinates (x_n, y_n), with the latter being obtained by solving the EOMs (see Section 3.2.2). The L-model assumption of planar surface leads to:

$$y_G = \frac{1}{\pi R^2 h}\int_0^R \int_0^{2\pi} \int_{-\frac{h}{2}}^{\frac{h}{2}+\sum\overline{\eta}_n \frac{r}{R}\sin\theta} r^2\sin\theta\ dz d\theta dr = \frac{R}{4h}\sum_{n=1}^{\infty}\overline{\eta}_n \tag{3.21}$$

Regarding the NL model, by exploiting one of the Bessel function properties, i.e.

$$\int_0^R r^2 J_1(\xi_{1n}\frac{r}{R})\ dr = R^3\frac{J_1(\xi_{1n})}{\xi_{1n}^2}, \tag{3.22}$$

y_G can be evaluated as:

$$y_G = \frac{1}{\pi R^2 h}\int_0^R \int_0^{2\pi} \int_{-\frac{h}{2}}^{\frac{h}{2}+\sum\overline{\eta}_n \frac{J_1(\xi_{1n}\frac{r}{R})}{J_1(\xi_{1n})}\sin\theta} r^2\sin\theta\ dz d\theta dr = \frac{R}{h}\sum_{n=1}^{\infty}\frac{\overline{\eta}_n}{\xi_{1n}^2} \tag{3.23}$$

Inserting the results from Equation (3.21) in Equation (3.17) yields:

$$\Big(\frac{R}{4h}\sum_{n=1}^{\infty}\overline{\eta}_n\Big)m_F = \sum_{n=1}^{\infty} y_n m_n \tag{3.24}$$

hence allowing the formulation of the n-th *sloshing height* (SH) for the L model:

$$\overline{\eta}_n = \frac{4hm_n}{m_F R}y_n \tag{3.25}$$

where the n-th generalized coordinate y_n is obtained from (3.13).
The same can be done regarding the NL model, exploiting the outcome of Equation (3.23) and substituting the value of y_G in Equation (3.17), namely:

$$\Big(\frac{R}{h}\sum_{n=1}^{\infty}\frac{\overline{\eta}_n}{\xi_{1n}^2}\Big)m_F = \sum_{n=1}^{\infty} y_n m_n. \tag{3.26}$$

This way, the estimation of the SH for the NL model is:

$$\overline{\eta}_n = \frac{\xi_{1n}^2 hm_n}{m_F R}y_n \tag{3.27}$$

with y_n computed by solving the EOMs of (3.12). For the sake of convenience, these results are summarized in the leftmost column of Table 3.1.

Table 3.1: SH estimation for a 1-dimensional planar motion, without (left column) and with (right column) an excitation along the z-axis.

	$\{\ddot{\mathbf{r}}_E\}_0 = [0 \ \ddot{r}_y \ 0]^T$	$\{\ddot{\mathbf{r}}_E\}_0 = [0 \ \ddot{r}_y \ \ddot{r}_z]^T$
L model	$\overline{\eta}_n = \frac{4hm_n}{m_F R} y_n$ (3.28)	$\overline{\eta}_n = \frac{4hm_n}{m_F R} y_n$ (3.29)
	y_n from (3.13)	y_n from (3.16)
NL model	$\overline{\eta}_n = \frac{\xi_{1n}^2 hm_n}{m_F R} y_n$ (3.30)	$\overline{\eta}_n = \frac{\xi_{1n}^2 hm_n}{m_F R} y_n$ (3.31)
	y_n from (3.12)	y_n from (3.15)

3.3.2 2-Dimensional Motion

When accounting for a 2-dimensional excitation, the plane Π, on which the *maximum sloshing height* (MSH) occurs, changes its orientation instantaneously, according to a rotation about the z-axis by the angle (Figure 3.1b):

$$\phi_n = \arctan\left(\frac{y_n}{x_n}\right) \tag{3.32}$$

If the liquid behavior is analyzed on the plane Π at every instant, Equation (3.17) can be extended to the radial coordinate of G, remembering that $r_n = \sqrt{x_n^2 + y_n^2}$:

$$r_G m_F = \sum_{n=1}^{\infty} r_n m_n = \sum_{n=1}^{\infty} m_n \sqrt{x_n^2 + y_n^2} \tag{3.33}$$

Equations (3.21) and (3.23) can be used to express r_G in terms of $\overline{\eta}_n$, depending on the adopted model. The approach seen in Section 3.3.1 can be similarly followed. In particular, the expression of $r_G = \frac{R}{4h} \sum_{n=1}^{\infty} \overline{\eta}_n$ from the L model can be inserted in Equation (3.33) to write the estimation of the MSH for the L model, namely:

$$\overline{\eta}_n = \frac{4hm_n}{m_F R} \sqrt{x_n^2 + y_n^2} \tag{3.34}$$

where x_n, y_n are computed from the EOMs in (3.13). On the other hand, considering the NL-model expression of $r_G = \frac{R}{h} \sum_{n=1}^{\infty} \frac{\overline{\eta}_n}{\xi_{1n}^2}$ allows to write the formula for the n-th MSH evaluation for the NL model, i.e.:

$$\overline{\eta}_n = \frac{\xi_{1n}^2 hm_n}{m_F R} \sqrt{x_n^2 + y_n^2} \tag{3.35}$$

with x_n, y_n from (3.12). For convenience, the formulas for the n-th MSH evaluation, both for the L model and the NL model, considering a 2-dimensional motion of the container are recapitulated in the leftmost column of Table 3.2.

Table 3.2: MSH estimation for a 2-dimensional planar motion, without (left column) and with (right column) an excitation along the z-axis.

	$\{\ddot{\mathbf{r}}_E\}_0 = [\ddot{r}_x \;\; \ddot{r}_y \;\; 0]^T$	$\{\ddot{\mathbf{r}}_E\}_0 = [\ddot{r}_x \;\; \ddot{r}_y \;\; \ddot{r}_z]^T$
L model	$\overline{\eta}_n = \frac{4hm_n}{m_F R}\sqrt{x_n^2 + y_n^2}$ (3.36)	$\overline{\eta}_n = \frac{4hm_n}{m_F R}\sqrt{x_n^2 + y_n^2}$ (3.37)
	x_n, y_n from (3.13)	x_n, y_n from (3.16)
NL model	$\overline{\eta}_n = \frac{\xi_{1n}^2 hm_n}{m_F R}\sqrt{x_n^2 + y_n^2}$ (3.38)	$\overline{\eta}_n = \frac{\xi_{1n}^2 hm_n}{m_F R}\sqrt{x_n^2 + y_n^2}$ (3.39)
	x_n, y_n from (3.12)	x_n, y_n from (3.15)

3.3.3 Remarks

By looking at the leftmost columns of Tables 3.1 and 3.2, one can point out that, for equal values of the generalized coordinates (x_n, y_n), the ratio between $\overline{\eta}_n$ obtained from the L model in Equations (3.28, 3.36) and $\overline{\eta}_n$ from the NL model in Equations (3.30, 3.38) is always $4/\xi_{1n}^2$. If only the 1^{st} mode is considered, this ratio is equal to $4/\xi_{11}^2 \approx 1.18$ and shows that the assumption of a planar free surface always overestimates the sloshing height compared to the assumption of a non-planar free surface. Furthermore, while in Equations (3.28, 3.30), $\overline{\eta}_n$ has the same sign of y_n, in Equations (3.36, 3.38) $\overline{\eta}_n$ is always positive. This means that Equations (3.28, 3.30) express the trend of the sloshing height only on one side of the container , with the sloshing height on the other side being estimated as the opposite of $\overline{\eta}_n$: in this case, we will simply talk about *sloshing height* (SH). Conversely, in Equations (3.36, 3.38), $\overline{\eta}_n$ indicates the maximum peak that occurs on the container wall on a plane oriented as described in Equation (3.32): in this case, we will use the expression *maximum sloshing height* (MSH).

3.3.4 Extension to 3-Dimensional Motion

When an additional excitation along the z-axis is taken into account, the formulas for the SH and the MSH estimation are reported in the rightmost columns of Table 3.1 and 3.2. They are seemingly identical to the ones employed in the case of a planar motion along the xy plane. However, the generalized coordinates x_n, y_n are obtained by solving the EOMs in (3.15) for the NL model and in (3.16) for the L model, instead of Equations (3.12) and (3.13), respectively.
Summing up, once the liquid properties are known (in terms of container radius R, static height h and density ρ), the equivalent-discrete model parameters (see Section 3.2) can be computed, and the 3-dimensional formulation can be applied to study the liquid sloshing for a general motion of the container, namely $\{\ddot{\mathbf{r}}_E\}_0 = [\ddot{r}_x \;\; \ddot{r}_y \;\; \ddot{r}_z]^T$, by simply employing the EOMs in (3.16) if the L model is adopted, or by solving the EOMs in (3.15) if the NL model is chosen. Once that the generalized coordinates (x_n, y_n) of

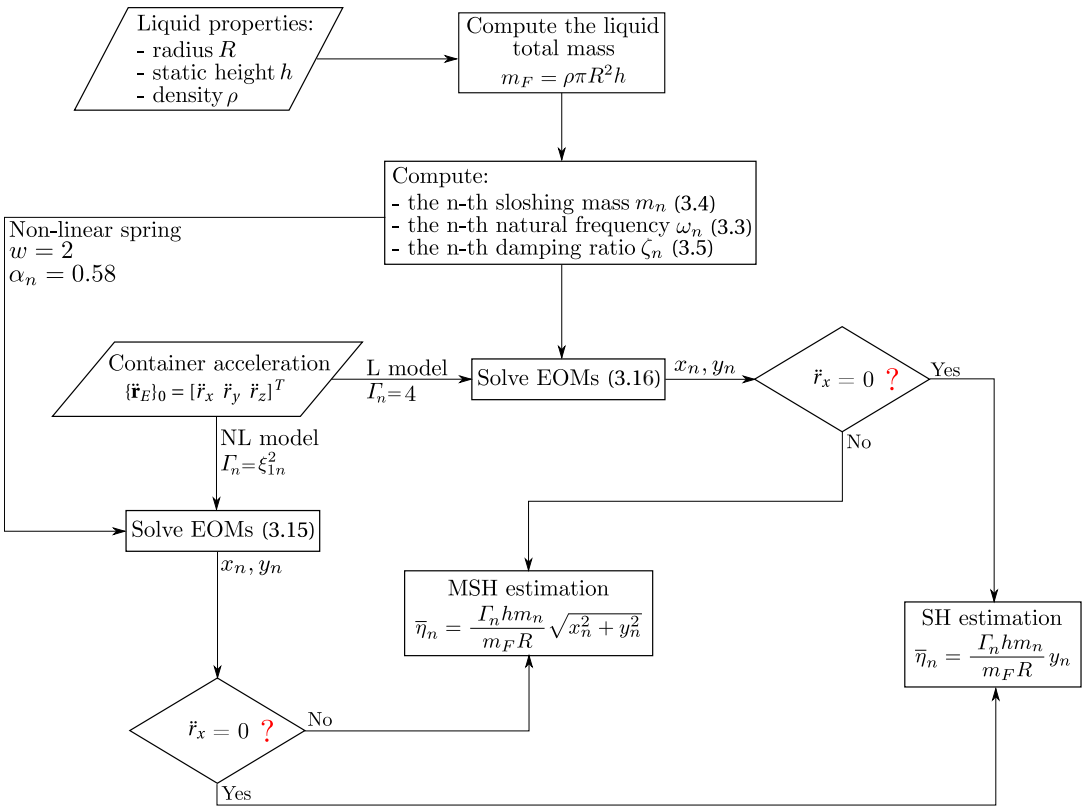


Figure 3.3: Flowchart describing the computation of the liquid sloshing height, depending on the type of container acceleration and the adopted model.

the n-th sloshing mass are computed, if the container is not subject to an acceleration along the x-axis throughout the whole motion ($\ddot{r}_x = 0$ [m/s^2]), we may refer to Table 3.1 for the SH estimation. Conversely, if both $\ddot{r}_x$ and $\ddot{r}_y$ are different from zero, the MSH may be estimated by referring to Table 3.2. The aforementioned procedure for sloshing-height estimation is illustrated by the flowchart in Figure 3.3.
Hence, the sloshing-height estimation requires the numerical solution of the EOMs to find the generalized coordinates (x_n, y_n) of the n-th sloshing mass, together with the computation of the formulas in Tables 3.1 and 3.2, depending on the adopted model. Regarding our (non-optimized) Matlab implementation, the average computational time is 0.7 secs, for either the linear and the non-linear formulations.

3.4 Time-Optimal Trajectory Planning

3.4.1 Trajectory Definition

The path and the orientation that the robot end-effector has to follow to realize the desired task are prescribed. In particular, the end-effector orientation is usually constant, so that we choose zero angular velocity and acceleration throughout the motion, namely $\boldsymbol{\omega}_E = \boldsymbol{\alpha}_E = \mathbf{0}$. The path of the reference point on the end-effector is parameterized in terms of a parameter s (arc length):

$$\mathbf{r}_E = \mathbf{r}_E(s), \qquad s \in [0,1] \tag{3.40}$$

and can be defined by using B-splines [35], [32] as

$$\mathbf{r}_E(s) = \sum_{j=0}^{m} B_j^d(s)\mathbf{p}_j, \qquad s \in [0,1] \tag{3.41}$$

where B_j^d are the B-spline basis functions of degree d and $\mathbf{p}_j$ are the $m+1$ control points. The motion law of the path parameter $s(t)$ allows the trajectory to be defined as:

$$\dot{\mathbf{r}}_E(s,\dot{s}) = \mathbf{r}'_E(s)\dot{s} \tag{3.42}$$

$$\ddot{\mathbf{r}}_E(s,\dot{s},\ddot{s}) = \mathbf{r}''_E(s)\dot{s}^2 + \mathbf{r}'_E(s)\ddot{s} \tag{3.43}$$

where $()' = \partial()/\partial s$ denotes the derivative w.r.t. the path parameter s.

3.4.2 Sloshing Limits

In order to define the constraints of the optimization problem, we use the L model for the SH (MSH) estimation, because it is more conservative (see Section 3.3.3). Furthermore, in [29], it is experimentally proven that the influence of sloshing modes higher than 1 is negligible when considering the L model. For these reasons, in the optimization-problem formulation of this paper, only the 1^{st}-mode generalized coordinates x_1, y_1 obtained from the resolution of the L-model EOMs are considered. This approach allows an easy composition of the constraints that the generalized coordinates must respect to fulfill the limit on the admissible value $\overline{\eta}_{lim}$ of the SH (MSH). Accordingly, for a 1-dimensional excitation along the y-direction, the constraint is:

$$|y_1| \leq \frac{m_F R}{4hm_1}\overline{\eta}_{lim} \tag{3.44}$$

Instead, for a 2-dimensional excitation in the xy plane, the boundaries on the generalized coordinates x_1, y_1 can be determined by assuming $|x_1| = |y_1|$, so that:

$$|x_1| \leq \frac{1}{\sqrt{2}}\frac{m_F R}{4hm_1}\overline{\eta}_{lim} \tag{3.45a}$$

$$|y_1| \leq \frac{1}{\sqrt{2}}\frac{m_F R}{4hm_1}\overline{\eta}_{lim} \tag{3.45b}$$

If compared with the constraint

$$\sqrt{x_1^2 + y_1^2} \leq \frac{m_F R}{4hm_1}\overline{\eta}_{lim} \tag{3.46}$$

whose corresponding feasible set for (x_1, y_1) is represented by a circle of radius $\frac{m_F R}{4hm_1}\overline{\eta}_{lim}$, the chosen constraints in (3.45) are represented by the square which is inscribed inside the aforementioned circle, thus providing a more restrictive feasible set for the generalized coordinates (x_1, y_1). Furthermore, the assumption $|x_1| = |y_1|$, besides being more conservative, leads to the formulation of two linear constraints, hence avoiding a nonlinear constraint and aiding the numerical solution of the optimization.

3.4.3 Problem Formulation

The jerk of the path parameter s is used as control input, in order to ensure a smooth trajectory [32]:

$$u = \dddot{s} \tag{3.47}$$

The system state is defined by a vector $\mathbf{x} \in \mathbb{R}^{13}$, namely:

$$\mathbf{x} = [s \ \dot{s} \ \ddot{s} \ \mathbf{q}^T \ x_1 \ y_1 \ \dot{x}_1 \ \dot{y}_1]^T \tag{3.48}$$

where $\mathbf{q} \in \mathbb{R}^6$ is the array of robot joint coordinates. Hence, the overall optimization problem can be formulated as

$$\min_{t_e,u} \left[\int_0^{t_e} (1 + ku^2) \, dt \right] \tag{3.49a}$$

subject to

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, u) \tag{3.49b}$$

$$\mathbf{x}(0) = [0 \ 0 \ 0 \ \mathbf{q}_0^T \ 0 \ 0 \ 0 \ 0]^T \tag{3.49c}$$

$$\mathbf{x}(t_e) = [1 \ 0 \ 0 \ \mathbf{q}_e^T \ 0 \ 0 \ 0 \ 0]^T \tag{3.49d}$$

$$|\dot{\mathbf{q}}| \leq \dot{\mathbf{q}}_{max} \tag{3.49e}$$

$$|\boldsymbol{\Theta}_B(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})\mathbf{p}_B| \leq \mathbf{Q}_{M,max} \tag{3.49f}$$

$$\mathbf{h}(x_1, y_1, \overline{\eta}_{lim}) \leq \mathbf{0} \tag{3.49g}$$

$$|u| \leq u_{max} \tag{3.49h}$$

The cost functional in (3.49a) is a trade-off between minimal time and minimal overall jerk, determined by the constant k, whose value can be conveniently tuned. Function $\mathbf{f}$ (3.49b) takes into account the integration chain of the path parameter s from the control u (3.50a), the robot inverse kinematics (3.50b) and the EOMs expressing the sloshing dynamics (3.50c), namely:

$$\frac{d}{dt} \begin{bmatrix} s \\ \dot{s} \\ \ddot{s} \end{bmatrix} = \begin{bmatrix} \dot{s} \\ \ddot{s} \\ u \end{bmatrix} \tag{3.50a}$$

$$\frac{d}{dt}\mathbf{q} = \mathbf{J}^{-1}\boldsymbol{\xi}_E(s, \dot{s}) \tag{3.50b}$$

$$\frac{d}{dt} \begin{bmatrix} x_1 \\ y_1 \\ \dot{x}_1 \\ \dot{y}_1 \end{bmatrix} = \begin{bmatrix} \dot{x}_1 \\ \dot{y}_1 \\ -2\zeta_1\omega_1\dot{x}_1 - \omega_1^2 x_1 - \ddot{r}_{E,x}(s, \dot{s}, \ddot{s}) \\ -2\zeta_1\omega_1\dot{y}_1 - \omega_1^2 y_1 - \ddot{r}_{E,y}(s, \dot{s}, \ddot{s}) \end{bmatrix} \tag{3.50c}$$

In (3.50b), the term $\mathbf{J} = \mathbf{J}(\mathbf{q})$ is the Jacobian matrix of the robot, whereas $\boldsymbol{\xi}_E = [\dot{\mathbf{r}}_E^T \ \boldsymbol{\omega}_E^T]^T$ represents the end-effector twist. Equality constraints (3.49c, 3.49d) are the initial and final conditions on the state vector $\mathbf{x}$, where $\mathbf{q}_0 = \mathbf{q}(0)$ and $\mathbf{q}_e = \mathbf{q}(t_e)$[2] are obtained by solving the inverse position analysis of the robot in correspondence of the initial and

[2]The final condition on the robot joint coordinates is relaxed into an inequality by asking that the values of the final joint angles, computed by the optimizer, fall into the neighborhood of $\mathbf{q}(t_e)$ according to a certain tolerance, rather than restraining them to the precise value of $\mathbf{q}(t_e)$.

Table 3.3: Maximum velocity and torque for the i-th joint of the robot.

i	1	2	3	4	5	6
$\dot{q}_{i,max}$ [rad/s]	4.3	4.3	5.5	6.4	7.5	17.6
$Q_{M_i,max}$ [Nm]	6.24	6.24	6.08	2.16	5.92	4.24

final poses of the end-effectors. Inequality constraints (3.49e, 3.49f) consider the limits on the maximum joint velocities $\dot{\mathbf{q}}_{max}$ and joint motor torques $\mathbf{Q}_{M,max}$, respectively. The values of $\dot{\mathbf{q}}_{max}$ and $\mathbf{Q}_{M,max}$ for the industrial robot used for experiments (Stäubli RX130L[3]) are reported in Table 3.3. The computation of the joint torques requires the knowledge of the base parameters $\mathbf{p}_B$ (which are a linear combination of independent and dependent robot dynamic parameters [36]), with $\mathbf{\Theta}_B$ being the matrix obtained from the QR decomposition of the regressor matrix [37]. The inequality (3.49g) refers to the constraints that are imposed on the generalized coordinates x_1, y_1 to limit sloshing, as defined in (3.44, 3.45). The value of u_{max} is set to 1000 $1/s^3$ [38].

3.5 Experiments

3.5.1 Experimental Setup and Sloshing Measurement

The experimental setup comprises a cylindrical container with radius $R = 50$mm and a liquid static height $h = 70$mm. The liquid is water, which is colored by adding dark brown powder, in order to obtain a better contrast for the image processing analysis. Motions are performed by an industrial robot (Stäubli RX130L) and recorded by a GoPro Hero3[4] camera attached on the same tray that hosts the container.
From the recorded videos, the extraction of the experimental sloshing height is obtained by means of a routine implemented in Matlab. In particular, once that each frame of the video is binarized according to a proper threshold level of the grayscale, the image must be processed in two different ways, depending on the excitation type:

- **SH detection:** when the excitation is given by a 1-dimensional acceleration along the y-axis with or without the addition of $\ddot{r}_z$ on the vertical direction, the trend of the SH can be examined on only one side of the container (Section 3.3.3); in this case, the SH is detected by identifying the black pixel with the highest z-coordinate on the rightmost side of the container (Figures 3.4a, 3.4b); then, the SH is evaluated as the difference (converted in mm) between the vertical coordinate of the detected pixel and the one of the pixel representing the liquid static height h.

- **MSH detection:** when the excitation is a general planar or spatial motion, the MSH can occur wherever on the container wall. The MSH is again detected as the black pixel with the highest z-coordinate, but in this case the whole lateral surface of the container is considered. Furthermore, a distinction is made between a peak occurring on the front part of the container and a peak on the rear wall, noticing that, in the former case the liquid image presents a uniform

[3] https://www.staubli.com
[4] https://gopro.com

black shape (Figure 3.4c), whereas, in the latter case, the liquid image is characterized by white regions, due to the light reflection on the liquid free surface (Figure 3.4d). If the ratio between the area occupied by the white regions and the area of the liquid image is under a certain percentage threshold, the peak is estimated in the front part of the container; conversely, the peak is detected in the rear part of the container if the ratio is greater than the specified threshold. Through this distinction, the peak can be correctly located on the container surface and the knowledge of the image depth can be used to obtain a more realistic measure of the MSH. In particular, thanks to a preliminary detection of the container pose w.r.t. the camera frame, the expression of the container lateral surface is known. Additionally, the identification of the peak in the image plane allows the computation of the line connecting the camera frame origin with the identified peak. The intersection between the line and the cylindrical lateral surface provides two points in the 3-dimensional space. If the condition of Figure 3.4c occurs, the point nearest to the camera is taken and its vertical coordinate is representative of the MSH; in case the condition of Figure 3.4c is verified, the vertical coordinate of the farthest point is used to obtain the MSH.

3.5.2 Validation Trajectories

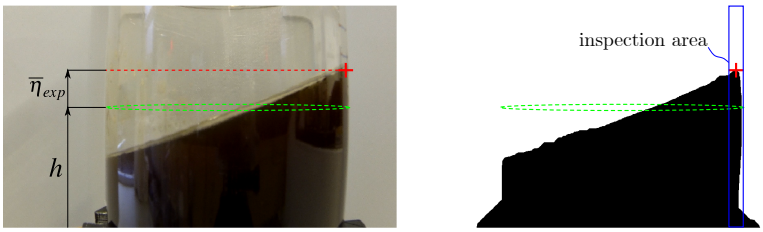
The trajectories are planned so that the robot follows three geometrical paths on the xy plane (Figure 3.5), each of them with different motion profiles, characterized by increasing container accelerations:

- a back-and-forth linear path (indicated as *l-motion*);
- an eight-shaped path (*e-motion*);
- a circular path, performed twice in succession (*c-motion*).

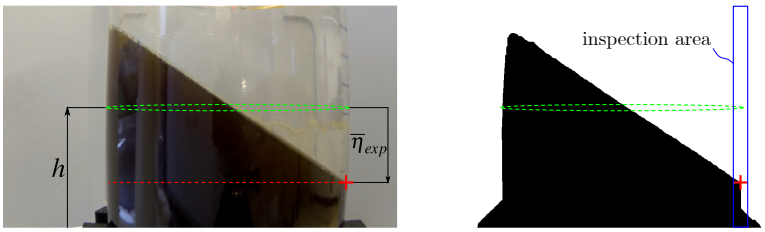
In Figure 3.6, the trends of the second time derivative $\ddot{s}$ of the path parameter are illustrated: for every path, all three motion profiles are shown. Note that the legend refers to the maximum of the container acceleration norm $\|\ddot{\mathbf{r}}_E\|_{max}$ reached during the corresponding motion.

Additionally, the 2-dimensional planar motions obtained from a modified trapezoidal motion law with 6 segments of $\ddot{s}$, are extended into 3-dimensional motions (*l3-motion*,*e3-motion*,*c3-motion*), through the inclusion of an excitation along the z-axis (Figure 3.7a). In particular, for the e3-motion and the c3-motion, the accelerations along the x and y directions are kept unchanged with respect to the corresponding e-motion and c-motion of the 2-dimensional case, respectively. The same cannot be said about the l3-motion, where the acceleration along the y-axis was slightly modified, to meet the robot limits. As a result of the extension, if 2-dimensional and 3-dimensional paths are observed from the top of the xy plane, they share the same shape, whereas, from a front perspective, the vertical coordinate changes along the path according to an excursion Δz (Figure 3.7b). On each 2-dimensional path, two different trends of $\ddot{r}_z$ are considered, thus providing two different 3-dimensional paths, characterized by two different values of Δz:

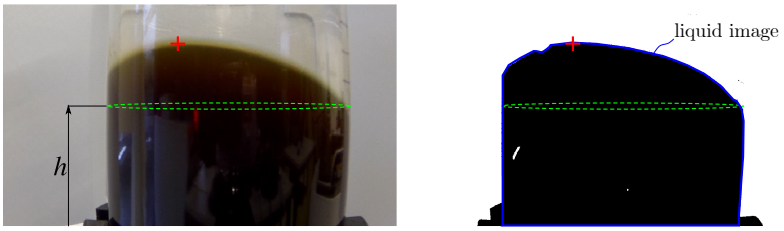
- a profile with moderate dynamics, namely $|\ddot{r}_z|_{max} \in [2,3]\mathrm{m/s}^2$;



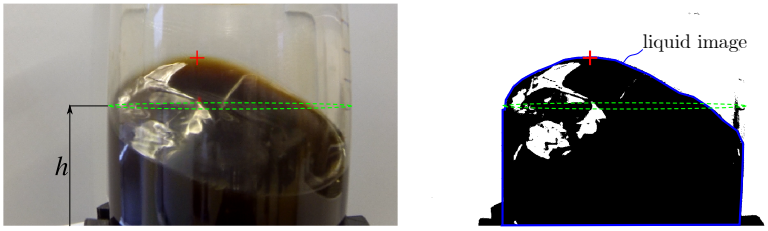
(a) SH detection: positive peak on the rightmost container wall.



(b) SH detection: negative peak on the rightmost container wall.



(c) MSH detection: front peak.



(d) MSH detection: back peak.

Figure 3.4: Snapshots from the image processing analysis of the recorded videos.

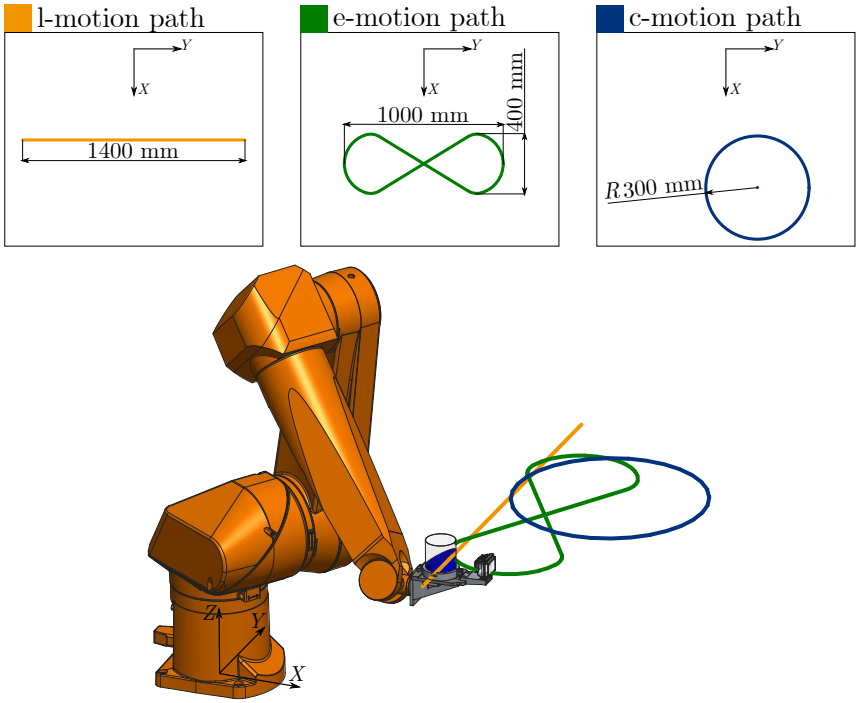
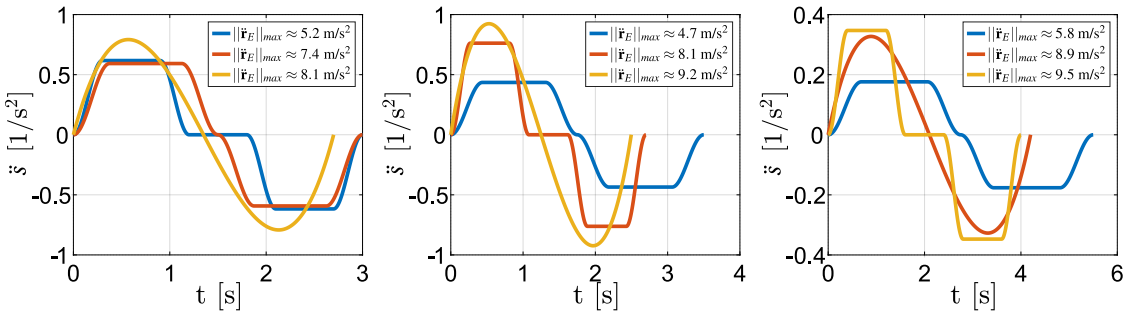


Figure 3.5: The three planar paths followed by the robot during experimental validation.



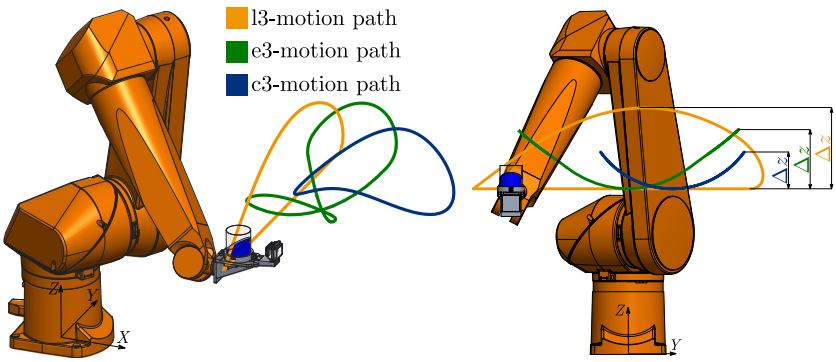
(a) l-motion. (b) e-motion. (c) c-motion.

Figure 3.6: The three motion profiles performed per each planar path.

- a profile with more prominent dynamics, namely $|\dddot{r}_z|_{max} \in [4,5]\text{m/s}^2$.

3.5.3 Experimental Results

In Figure 3.8, the 2-dimensional L and NL model predictions are compared with the results from the experimental motions, only considering the 1^{st} sloshing mode. A good adherence between the experiments and the models can be appreciated for the 1-dimensional motions (Figures 3.8a, 3.8b, 3.8c), and tracking remains reliable also for 2-dimensional motions, especially when considering lower values of $\|\ddot{\mathbf{r}}_E\|_{max}$ (Figures 3.8d, 3.8g). As the value of the 2-dimensional excitation $\ddot{\mathbf{r}}_E$ is increased, the model predictions still capture the trend of the real liquid MSH, although they seem to lose accuracy in correspondence of the peaks reached by the liquid (Figures 3.8e, 3.8f, 3.8h, 3.8i). This can be eventually attributed to two reasons:



(a) General view. (b) Front view.

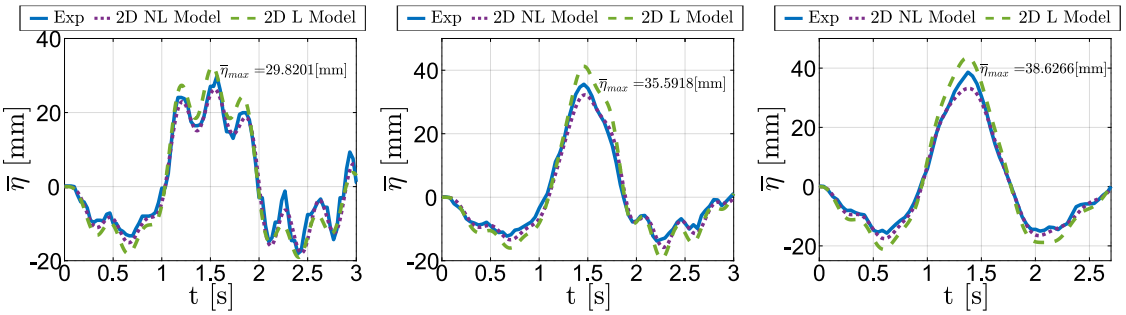
Figure 3.7: The 3-dimensional paths obtained as an extension of the 2-dimensional case.

- the high dynamics given to the container causes a regime of strongly nonlinear motions, where the liquid free surface loses the assumed shape and shows instantaneous swirly peaks, as illustrated in Figure 3.9a, that cannot be tracked by the models;
- the height of the frames that are employed for the image processing analysis, grants a greater field of view when the liquid peak occurs on the rear wall of the container (Figure 3.9b), whereas, for a peak on the front wall (Figure 3.9c), the value of the real MSH is saturated by the frame upper limit; this explains the discrepancy between the experiments and the prediction models in the red areas that are highlighted in Figures 3.8e, 3.8f, 3.8h, 3.8i.

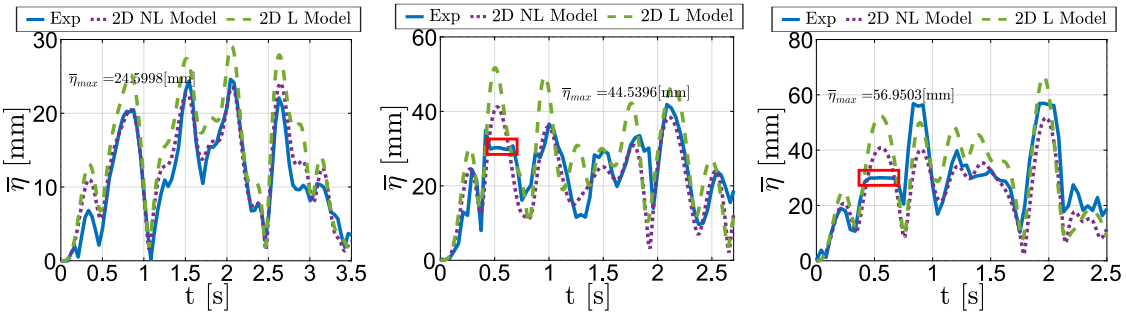
Regarding the former case, the L-model assumption of a planar surface during motion loses adherence with reality when the container acceleration is roughly greater than $\|\ddot{\mathbf{r}}_E\|_{max} \approx 8\text{m/s}^2$, with a maximum jerk of $\|\dddot{\mathbf{r}}_E\|_{max} \approx 60\text{m/s}^3$. This is reflected in a less accurate correspondence between the model prediction and the experimental results, even though the model evaluation is still reliable. It is worth observing that the maximum peaks are always overestimated by the L model. As far as the NL model is concerned, the assumption of a free surface described by means of the first-kind Bessel function finds a better correspondence with reality, if compared with the L model. The model estimation begins to lose adherence with reality when the container acceleration reaches a value of roughly $\|\ddot{\mathbf{r}}_E\|_{max} \approx 9.5\text{m/s}^2$, with a maximum jerk of $\|\dddot{\mathbf{r}}_E\|_{max} \approx 74\text{m/s}^3$. However, the global maxima predicted by the NL model are always below the experimental ones.
Table 3.4 summarizes the obtained results by reporting the accuracy index ϵ_{mod} expressing the error between the model and the experimental maxima:

$$\epsilon_{mod} = \frac{\overline{\eta}_{max,mod} - \overline{\eta}_{max,exp}}{\overline{\eta}_{max,exp}} \times 100\% \tag{3.51}$$

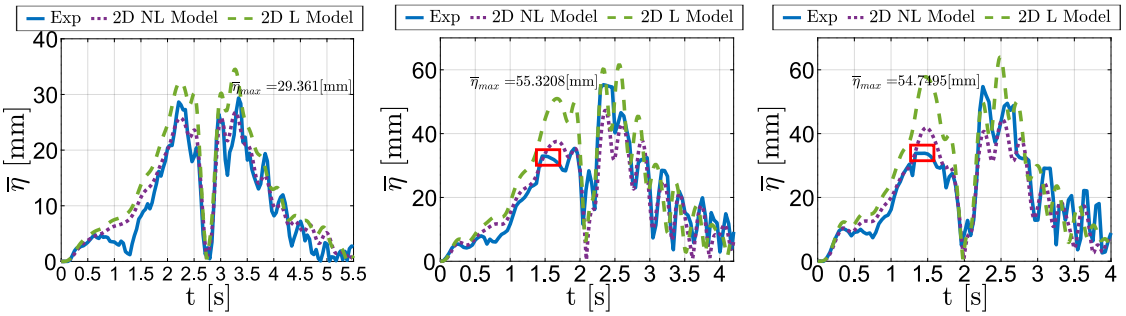
where the subscripts mod and exp denote the adopted model (2-dimensional L/NL) and the experimental results, respectively. For all motions, $|\epsilon_{2D-L}|$ is always below 18%, and $|\epsilon_{2D-NL}|$ never exceeds 19%, with the NL model granting a better tracking during the whole time period. Furthermore, the positive sign of ϵ_{2D-L} proves that the L model always overestimates the real SH and MSH peaks, as expected (see Section 3.3.3), hence



(a) l-motion: $\|\ddot{\mathbf{r}}_E\|_{max} \approx 5.2\text{m/s}^2$. (b) l-motion: $\|\ddot{\mathbf{r}}_E\|_{max} \approx 7.4\text{m/s}^2$. (c) l-motion: $\|\ddot{\mathbf{r}}_E\|_{max} \approx 8.1\text{m/s}^2$.

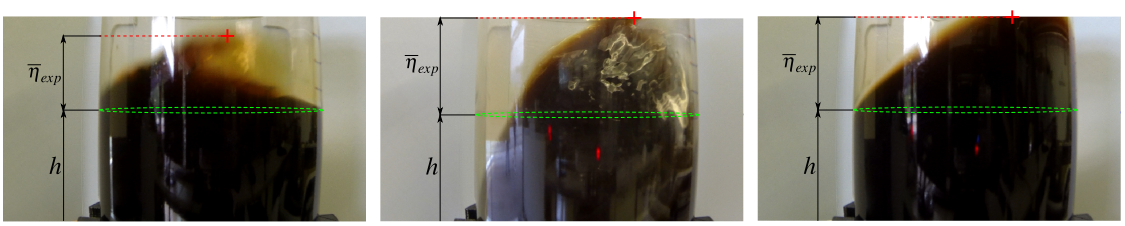


(d) e-motion: $\|\ddot{\mathbf{r}}_E\|_{max} \approx 4.7\text{m/s}^2$. (e) e-motion: $\|\ddot{\mathbf{r}}_E\|_{max} \approx 8.1\text{m/s}^2$. (f) e-motion: $\|\ddot{\mathbf{r}}_E\|_{max} \approx 9.2\text{m/s}^2$.



(g) c-motion: $\|\ddot{\mathbf{r}}_E\|_{max} \approx 5.8\text{m/s}^2$. (h) c-motion: $\|\ddot{\mathbf{r}}_E\|_{max} \approx 8.9\text{m/s}^2$. (i) c-motion: $\|\ddot{\mathbf{r}}_E\|_{max} \approx 9.5\text{m/s}^2$.

Figure 3.8: Comparison between the proposed models and the experimental results from the planar motions.



(a) Swirly peak. (b) Back peak. (c) Front peak.

Figure 3.9: Snapshots from the recorded videos, showing the different peaks reached by the liquid.

Table 3.4: Accuracy index ϵ_{mod} evaluated for the planar motions.

l-motion	$\|\ddot{\mathbf{r}}_E\|_{max} \approx 5.2\text{m/s}^2$	$\|\ddot{\mathbf{r}}_E\|_{max} \approx 7.4\text{m/s}^2$	$\|\ddot{\mathbf{r}}_E\|_{max} \approx 8.1\text{m/s}^2$
	$\epsilon_{2D-NL} = -11.9\%$	$\epsilon_{2D-NL} = -9.5\%$	$\epsilon_{2D-NL} = -14.1\%$
	$\epsilon_{2D-L} = 8.8\%$	$\epsilon_{2D-L} = 15.8\%$	$\epsilon_{2D-L} = 12.8\%$
e-motion	$\|\ddot{\mathbf{r}}_E\|_{max} \approx 4.7\text{m/s}^2$	$\|\ddot{\mathbf{r}}_E\|_{max} \approx 8.1\text{m/s}^2$	$\|\ddot{\mathbf{r}}_E\|_{max} \approx 9.2\text{m/s}^2$
	$\epsilon_{2D-NL} = -2.1\%$	$\epsilon_{2D-NL} = -7.5\%$	$\epsilon_{2D-NL} = -9.2\%$
	$\epsilon_{2D-L} = 18.3\%$	$\epsilon_{2D-L} = 16.2\%$	$\epsilon_{2D-L} = 17.1\%$
c-motion	$\|\ddot{\mathbf{r}}_E\|_{max} \approx 5.8\text{m/s}^2$	$\|\ddot{\mathbf{r}}_E\|_{max} \approx 8.9\text{m/s}^2$	$\|\ddot{\mathbf{r}}_E\|_{max} \approx 9.5\text{m/s}^2$
	$\epsilon_{2D-NL} = -8.7\%$	$\epsilon_{2D-NL} = -14.3\%$	$\epsilon_{2D-NL} = -19\%$
	$\epsilon_{2D-L} = 17.6\%$	$\epsilon_{2D-L} = 11.6\%$	$\epsilon_{2D-L} = 16.9\%$

providing a more conservative estimation.
In Figure 3.10, the results from the 3D motions are illustrated: for each motion, the 3-dimensional L and NL model predictions are compared with the 2-dimensional L and NL ones, to show the benefit obtained by employing the extended formulation. In general, the 3-dimensional models exhibit a better correspondence with respect to the 2-dimensional ones, especially when the vertical acceleration is more significant (e.g. when $|\ddot{r}_z|_{max} \in [4,5]\text{m/s}^2$). The definition of the index σ_{mod} expresses the mean absolute error between the experimental results and the model predictions:

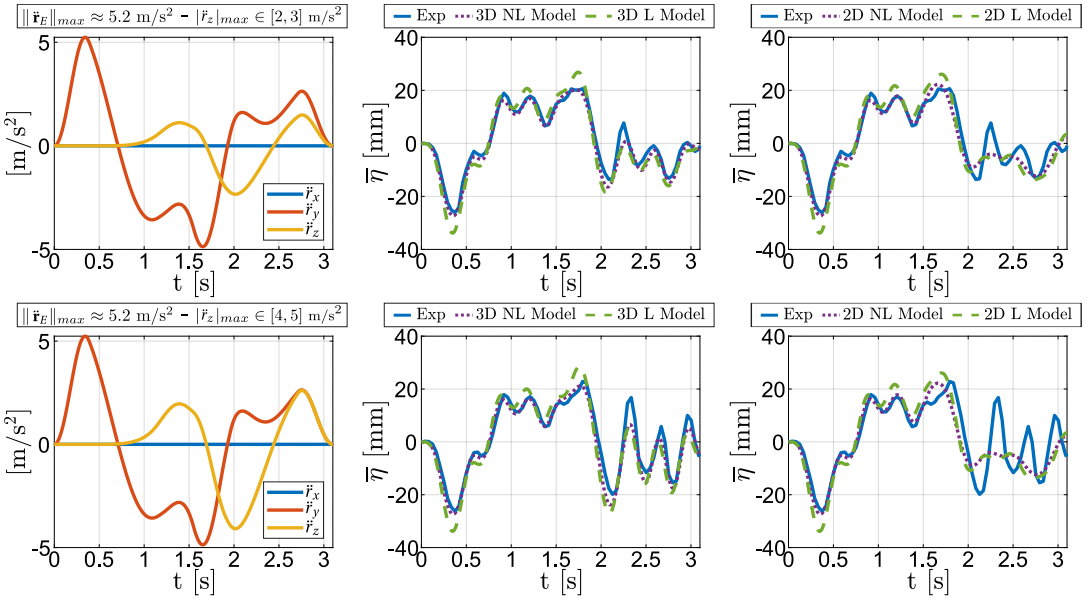
$$\sigma_{mod} = \frac{\sum_{i=1}^{N} |\overline{\eta}_{exp}(t_i) - \overline{\eta}_{mod}(t_i)|}{N} \tag{3.52}$$

where the subscripts mod and exp denote the adopted model (2-dimensional L/NL or 3-dimensional L/NL) and the experimental results, respectively, whereas t_i refers to the i-th time instant and N represents the number of samplings. The analysis of Table 3.5, in which the index σ_{mod} is evaluated for the 3-dimensional motions, confirms the qualitative evidence, i.e.:

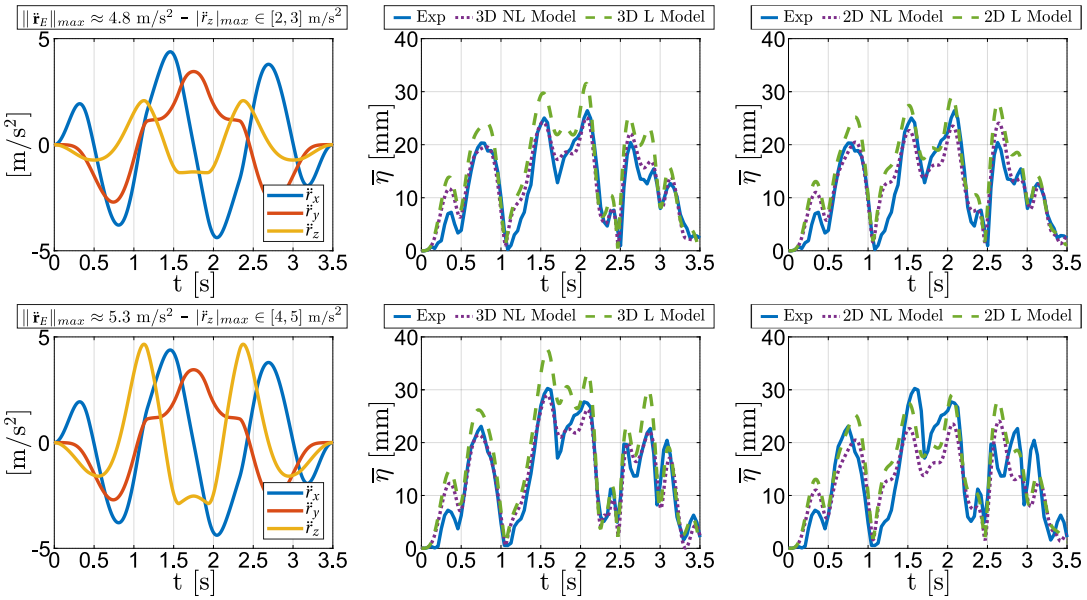
- for equal formulations (3-dimensional or 2-dimensional), the NL model grants a better tracking of the real-liquid trends with respect to the L model (for instance, in most cases, $\sigma_{3D-NL} < \sigma_{3D-L}$ and $\sigma_{2D-NL} < \sigma_{2D-L}$);
- in general, the 3-dimensional formulation provides a more reliable correspondence with reality, compared to the 2-dimensional formulation; indeed, in most cases $\sigma_{3D-NL} < \sigma_{2D-NL}$ and $\sigma_{3D-L} < \sigma_{2D-L}$.

3.5.4 Remarks

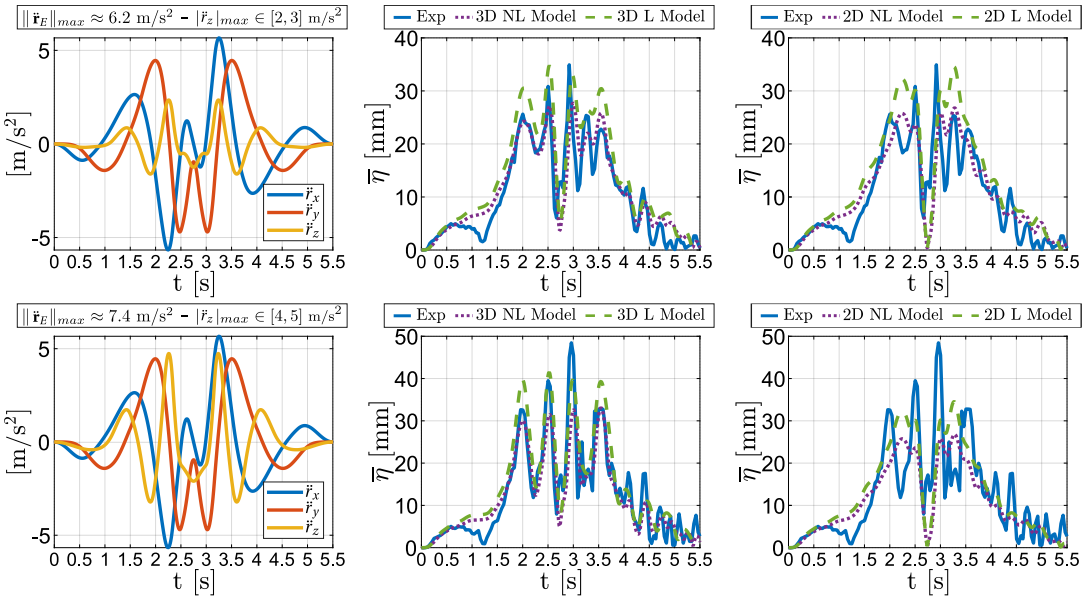
In the described experiments, the employed amount of liquid inside the container ($R = 50\text{mm}$, $h = 70\text{mm}$, with $h/R = 7/5$) has a mass of $m_F \approx 0.55$ kg. Considering only the first sloshing mode to model the liquid dynamics, from Equation (3.4), we can theoretically compute the mass m_1 which is responsible of the liquid sloshing ($m_1 \approx 0.18$ kg). If the case with the highest dynamics is taken into account (e.g. the c-motion with $\|\ddot{\mathbf{r}}_E\|_{max} \approx 9.5\text{m/s}^2$), it can be shown, from the simulations, that the inertia forces



(a) l3-motion: medium (first row) and high (second row) vertical acceleration.



(b) e3-motion: medium (first row) and high (second row) vertical acceleration.



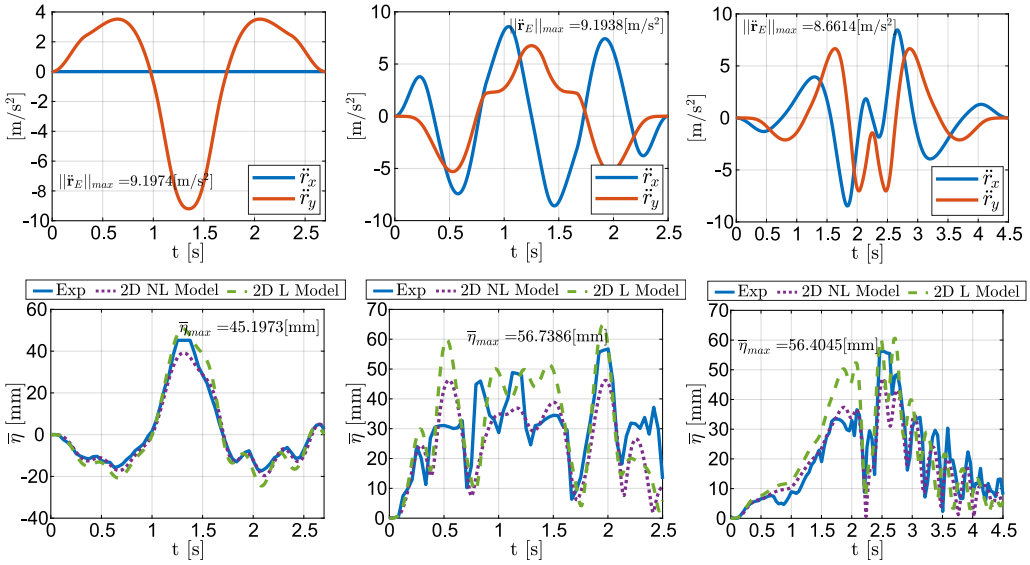
(c) c3-motion: medium (first row) and high (second row) vertical acceleration.

Figure 3.10: Results from the 3D motions.

Table 3.5: Accuracy index σ_{mod} evaluated for the 3-dimensional motions, comparing the 3-dimensional and the 2-dimensional formulations.

l3-motion ($\|\ddot{r}_z\|_{max} \in [2,3]\text{m/s}^2$)	$\sigma_{3D-NL} = 1.7\text{mm}$	$\sigma_{2D-NL} = 2.4\text{mm}$
	$\sigma_{3D-L} = 3.3\text{mm}$	$\sigma_{2D-L} = 4.1\text{mm}$
l3-motion ($\|\ddot{r}_z\|_{max} \in [4,5]\text{m/s}^2$)	$\sigma_{3D-NL} = 2.3\text{mm}$	$\sigma_{2D-NL} = 4.5\text{mm}$
	$\sigma_{3D-L} = 4.2\text{mm}$	$\sigma_{2D-L} = 6.1\text{mm}$
e3-motion ($\|\ddot{r}_z\|_{max} \in [2,3]\text{m/s}^2$)	$\sigma_{3D-NL} = 1.9\text{mm}$	$\sigma_{2D-NL} = 2.2\text{mm}$
	$\sigma_{3D-L} = 4.0\text{mm}$	$\sigma_{2D-L} = 3.3\text{mm}$
e3-motion ($\|\ddot{r}_z\|_{max} \in [4,5]\text{m/s}^2$)	$\sigma_{3D-NL} = 2.0\text{mm}$	$\sigma_{2D-NL} = 4.3\text{mm}$
	$\sigma_{3D-L} = 3.7\text{mm}$	$\sigma_{2D-L} = 4.0\text{mm}$
c3-motion ($\|\ddot{r}_z\|_{max} \in [2,3]\text{m/s}^2$)	$\sigma_{3D-NL} = 2.3\text{mm}$	$\sigma_{2D-NL} = 3.0\text{mm}$
	$\sigma_{3D-L} = 3.9\text{mm}$	$\sigma_{2D-L} = 4.2\text{mm}$
c3-motion ($\|\ddot{r}_z\|_{max} \in [4,5]\text{m/s}^2$)	$\sigma_{3D-NL} = 3.0\text{mm}$	$\sigma_{2D-NL} = 5.0\text{mm}$
	$\sigma_{3D-L} = 3.8\text{mm}$	$\sigma_{2D-L} = 5.3\text{mm}$

transmitted by the liquid on the robot end-effector are negligible, hence not influencing the robot dynamics. Conversely, as the value of m_F increases (and consequently so m_1 does, if h/R is fixed), the forces acting on the robot end-effector may become more important. This means that, in the performed experiments, the net force needed to hold and move the liquid-filled container does not influence the robot dynamics. Conversely, if a container with larger dimensions and filled with a greater amount of liquid is chosen, attention must be paid to its inertia actions. In this case, the dynamics of the liquid-filled container have to be inserted inside the formulation of the robot dynamics [39], in order to compute the necessary robot joint torques able to grant the correct accomplishment of the desired trajectory.
It is also worth mentioning the influence of the ratio h/R. In particular, for a fixed value of m_F, when h/R increases, the value of the sloshing mass m_1 drops down [22], whereas m_1 represents a more important contribution as h/R decreases. As long as the sloshing mass m_1 grows, the sloshing of the liquid becomes more significant, and the free surface might not satisfy the model assumptions. Consequently, for lower values of h/R, the prediction models may lose adherence with reality. This also finds confirmation by looking at the correlation between the first natural frequency and the ratio h/R (v. Equation (3.3)). The natural frequencies of the sloshing modes obtained by the equivalent mass-spring-damper model represent the frequencies in correspondence of which the liquid-free surface is predisposed to oscillate. When h/R decreases, ω_1 tends to lower values, meaning that the fundamental of the liquid free surface can be easily excited by the motion-law spectrum, in correspondence of the highest amplitudes [30]. This may result in a more non-linear behavior and shape of the liquid free surface, that might not be detected by the models, although an higher number of sloshing modes is considered.



(a) l-motion. (b) e-motion. (c) c-motion.

Figure 3.11: Non-optimized motions: for each type of motion, the values of $||\ddot{\mathbf{r}}_E||_{max}$ and $\overline{\eta}_{max}$ are shown.

3.5.5 Anti-Sloshing Trajectories

The same planar paths (see Figure 3.5) followed during the validation campaign described in Section 3.5.2 are employed to assess the optimization efficacy. In particular, the non-optimized trajectory planning considers a modified trapezoidal motion profile of $\ddot{s}$ from which the trend of the robot end-effctor acceleration $\ddot{\mathbf{r}}_E$ is computed (See Equation (3.43)). The results for the non-optimized motions are shown in Figure 3.11. For each motion, the first plot displays the acceleration of the robot end-effector (and thus of the container), whereas in the second one a comparison between the SH (MSH) from the experiments and that estimated by the NL model and the L model is provided. While for the 1-dimensional case (Figure 3.11a), both models exhibit a good tracking of the real SH, for the 2-dimensional motions (Figure 3.11b, 3.11c), a less accurate correspondence between the real MSH and the prediction models can be noticed. Nevertheless, the L model always overestimates the real SH and MSH peaks (indicated with $\overline{\eta}_{max}$ in the plots), hence leading to a more conservative evaluation. In Figure 3.12 the snapshots taken from the recorded videos show the worst cases that occurred during the non-optimized motions: in all situations, the liquid would have spilled out if the container were not closed (notice that the upper limit of the image coincides with the lid front view)[5].

For each path, three optimizations were carried out, each of them differing in the imposed value of $\overline{\eta}_{lim}$. Figure 3.13 reports the results obtained by solving the constrained optimization for the l-motion, the e-motion and the c-motion, imposing a limit on the SH (MSH) of $\overline{\eta}_{lim} = 20$mm. In particular, the left-most plot of each subfigure 3.13a, 3.13b, 3.13c depicts the ratio between the robot joint velocities and their maximum admissible values, namely $\dot{q}_i / \dot{q}_{i-max}$, with $i = 1,\dots,6$; the central plot represents the ration between the robot joint torques and their maximum values, namely Q_{M_i} / Q_{M_i-max}, with $i = 1,\dots,6$; finally, in the right-most plot the 1^{st} mode generalized

[5]Both non-optimized and optimized motions can be seen at the link sloshing videos. In the non-optimized ones, the liquid oscillations are not negligible, even after the task execution.

(a) l-motion. (b) e-motion. (c) c-motion.

Figure 3.12: Snapshots from the non-optimized motion videos, exhibiting the peaks reached by the liquid.

coordinates x_1, y_1, as obtained from the optimization, are shown. It can be noticed that, for each motion, among the three different constraints (see Equations (3.49e), (3.49f), (3.49g)) the conditions on the maximum robot joint velocities and torques are always inactive (i.e. they are always fulfilled with margin), whereas the conditions on the sloshing height, and hence on the 1^{st} mode generalized coordinates, are active. Indeed, at some instants, either x_1 or y_1 reaches the allowed limit, which finds correspondence with constraints (3.44, 3.45) and proves the achievement of a time-optimal solution (in the model scenario).
Figure 3.14 illustrates the experimental results of the optimized motions. In the l-motion the value $\overline{\eta}_{max}$ almost saturates the boundaries (Figure 3.14a) or exceeds them by a negligible value (Figure 3.14b, 3.14c), whereas in the e-motion (Figure 3.14d, 3.14e, 3.14f) and in the c-motion (Figure 3.14g, 3.14h, 3.14i), the MSH peaks are always below the imposed limits, as expected, since (3.45) is based on a conservative assumption (see Section 3.4.2).
In Table 3.6, an evidence of the optimization thoroughness is given by introducing two performance indices:

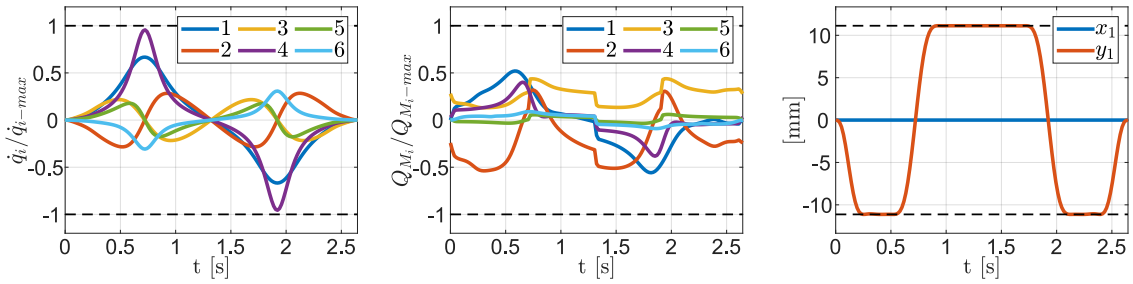
$$\%\overline{\eta} = \frac{\overline{\eta}_{max,N-OPT} - \overline{\eta}_{max,OPT}}{\overline{\eta}_{max,N-OPT}} \times 100 \tag{3.53}$$

$$\%T = \frac{T_{OPT} - T_{N-OPT}}{T_{N-OPT}} \times 100 \tag{3.54}$$

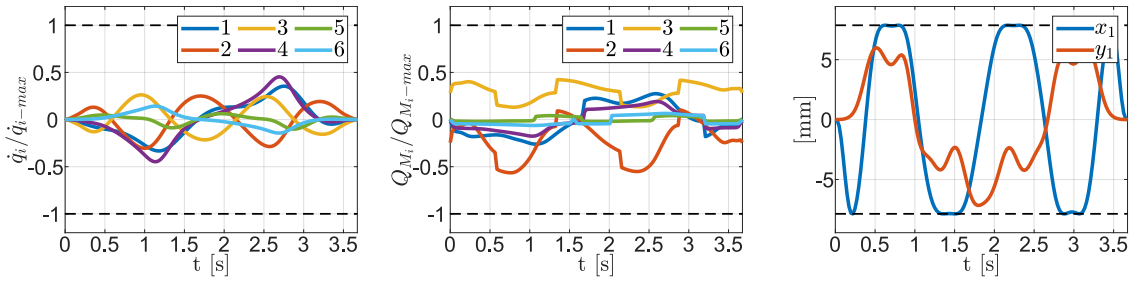
where OPT and $N-OPT$ denote the optimized and non-optimized motions, respectively. Index $\%\overline{\eta}$ gives a measure of the benefit obtained from the optimized motion w.r.t. the non-optimized one, whereas $\%T$ indicates the effort required to achieve such a benefit. It is apparent from Table 3.6 that, while the advantage in terms of $\overline{\eta}$ is always significant, the tighter is the constraint on the SH (MSH), the higher is the required effort in terms of trajectory duration.

3.6 Conclusions

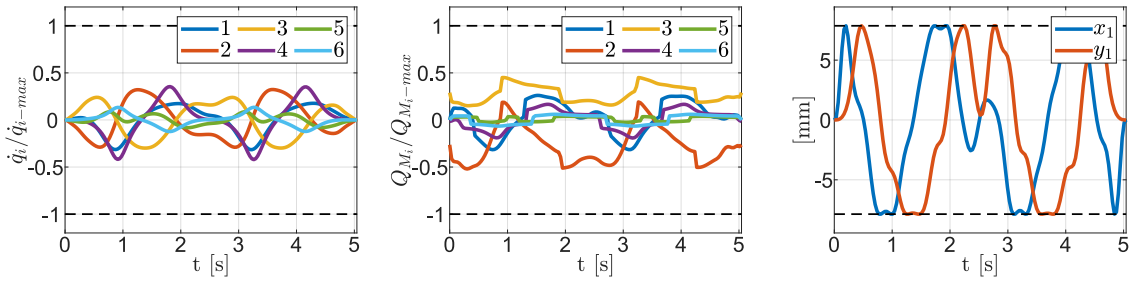
In this Chapter, a novel technique for the sloshing-height estimation of a liquid inside a cylindrical container subject to 2-dimensional planar motions was proposed [11], extending what was presented in [29]. The technique is based on simple discrete mechanical models, rather than machine-learning or complex fluid dynamics methodologies, thus granting a reliable and easy-to-compute estimation.
Experiments, considering three container planar paths performed by an industrial robot with different motion profiles, were presented and the relative results were discussed.



(a) Optimized l-motion with $\overline{\eta}_{lim} = 20$mm.



(b) Optimized e-motion with $\overline{\eta}_{lim} = 20$mm.

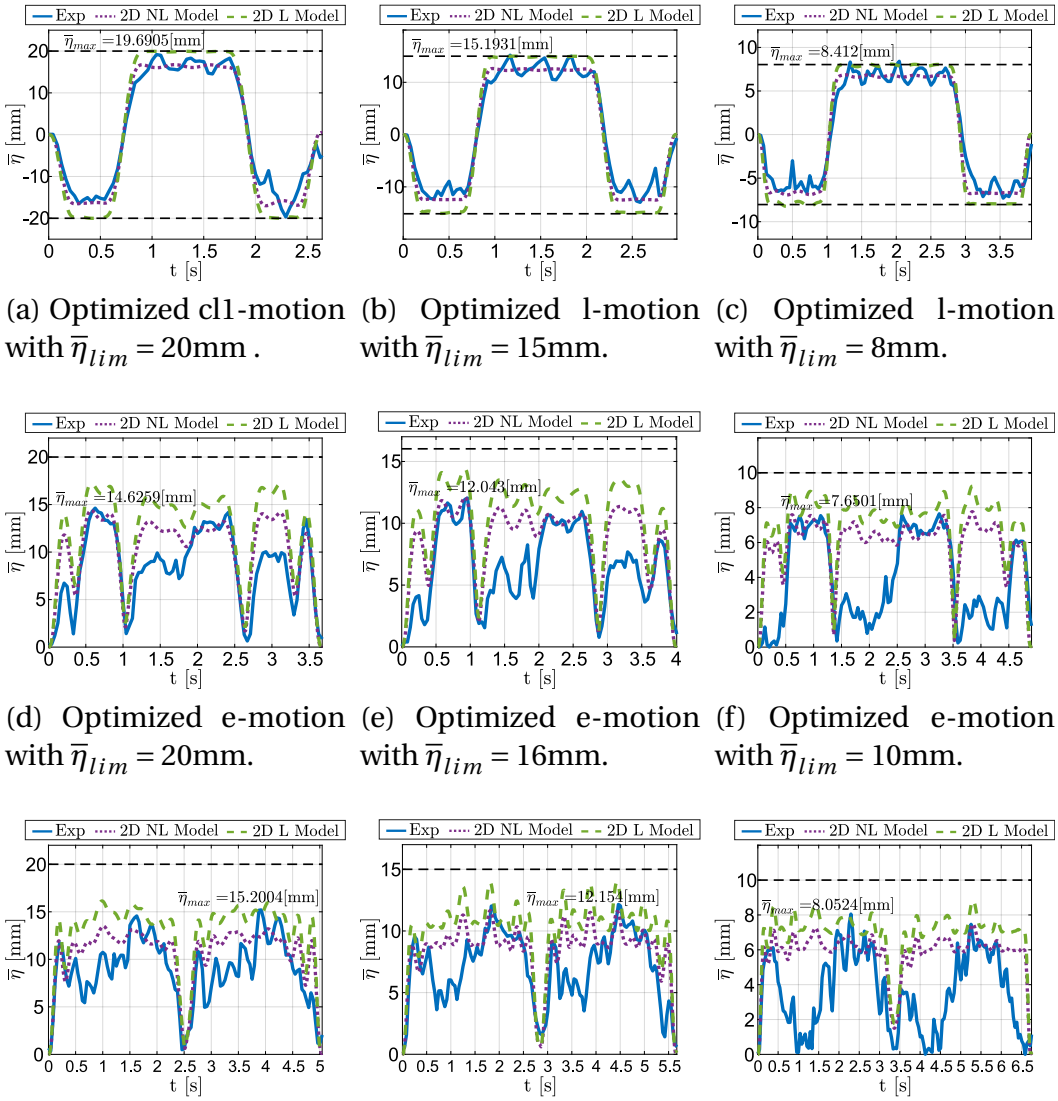


(c) Optimized c-motion with $\overline{\eta}_{lim} = 20$mm.

Figure 3.13: Results from the constrained optimization.

Table 3.6: Performance indices for the optimized motions.

l-motion	$\overline{\eta}_{lim} = 20$mm	$\overline{\eta}_{lim} = 15$mm	$\overline{\eta}_{lim} = 8$mm
	$\%\overline{\eta} = 56.4\%$	$\%\overline{\eta} = 66.4\%$	$\%\overline{\eta} = 81.4\%$
	$\%T = -2\%$	$\%T = 10.6\%$	$\%T = 46.5\%$
e-motion	$\overline{\eta}_{lim} = 20$mm	$\overline{\eta}_{lim} = 16$mm	$\overline{\eta}_{lim} = 10$mm
	$\%\overline{\eta} = 74.2\%$	$\%\overline{\eta} = 78.8\%$	$\%\overline{\eta} = 86.5\%$
	$\%T = 47\%$	$\%T = 60.4\%$	$\%T = 96.3\%$
c-motion	$\overline{\eta}_{lim} = 20$mm	$\overline{\eta}_{lim} = 15$mm	$\overline{\eta}_{lim} = 10$mm
	$\%\overline{\eta} = 73.1\%$	$\%\overline{\eta} = 78.5\%$	$\%\overline{\eta} = 85.7\%$
	$\%T = 12.1\%$	$\%T = 26\%$	$\%T = 49.9\%$



(a) Optimized cl1-motion with $\overline{\eta}_{lim} = 20$mm .
(b) Optimized l-motion with $\overline{\eta}_{lim} = 15$mm.
(c) Optimized l-motion with $\overline{\eta}_{lim} = 8$mm.
(d) Optimized e-motion with $\overline{\eta}_{lim} = 20$mm.
(e) Optimized e-motion with $\overline{\eta}_{lim} = 16$mm.
(f) Optimized e-motion with $\overline{\eta}_{lim} = 10$mm.
(g) Optimized c-motion with $\overline{\eta}_{lim} = 20$mm.
(h) Optimized c-motion with $\overline{\eta}_{lim} = 15$mm.
(i) Optimized c-motion with $\overline{\eta}_{lim} = 10$mm.

Figure 3.14: Optimized motions.

An accuracy index, expressing the error between the model and the experimental maxima of the sloshing height, was used to prove the effectiveness of the estimation, even for high values of the container acceleration (up to 9.5m/s^2).
Additionally, the 2-dimensional formulation was extended to a 3-dimensional one, to take into account an additional excitation along the vertical direction, hence resulting in a 3-dimensional motion of the container [13]. The planar paths of the 2-dimensional case were extended by adding a z-axis acceleration (up to 5m/s^2). The mean absolute error between the experimental results and the model predictions was examined to evaluate the benefit, in terms of accuracy, obtained by employing the 3-dimensional formulation instead of the 2-dimensional one.
In addition, the technique for the sloshing-height evaluation was used for the time-optimal trajectory planning of an industrial robot, with the aim of limiting liquid sloshing inside a cylindrical container subject to 1-dimensional and 2-dimensional planar motions [12]. Experimental results were described and discussed, for both non-optimized and optimized motions. Two performance indexes were introduced to as-

sess the constrained-optimization effectiveness. The main advantage of the presented approach lies in the possibility of keeping the liquid sloshing height below a specified threshold during the whole robot motion, while at the same time guaranteeing the shortest trajectory duration compatible with this constraint and the given hardware.
Future work will include a further validation study considering higher values of the vertical acceleration and the use of the proposed sloshing-height estimation for square-section containers, adapting the formulation that was presented in Section 3.3. In addition, a detailed sensitivity analysis will be addressed to assess the influence of the container dimensions on the accuracy of the model predictions (see Section 3.5.4).
Furthermore, regarding the planning of anti-sloshing trajectories, the proposed optimization approach will be extended to 3-dimensional motions and different optimization strategies will be examined, for a comparison in terms of computational time and accuracy.

Chapter 4

Cooperative Grasping

In this Chapter, the dual-arm manipulation, employing two collaborative robots (cobots) carrying a common object, is addressed. The adopted control strategy is admittance control, aimed at compliantly modulating the forces exerted by the cobot end-effectors on the object. The chosen manipulation approach is cooperative grasping, which is characterized by unilateral contact constraints, hence enabling relative motion between the end-effectors and the object. To ensure the stability of the cooperative grasping (i.e. no slipping of the object), a normal internal force, granting the fulfillment of the friction-cone conditions, can be prescribed. In this work, the trend of the internal force is inserted among the inputs of a time-optimal trajectory planning, to find the minimal internal prestress able to satisfy the friction-cone conditions and manipulate the object in minimal time. To validate this novel approach, experiments on different paths are presented and discussed.
The work presented in this Chapter is under review at [40].

4.1 Motivation

The automotive industry was the first to employ robotics within assembly lines [41]. Nowadays, industrial robots accomplish half of the tasks required in a typical automotive factory. Industrial robots usually present high-dynamics performances, which, combined with their heavy and sharp-cornered bodies, may produce large inertia actions, becoming harmful in case of collisions with things or humans. For this reason, the installation of industrial robots often requires cages and large basements, thus making them stationary and relegating industrial robots to spaces that are inaccessible to human operators.
With the advent of Industry 4.0, which puts robots and humans working side by side in shared environments, a particular class of serial robots, called collaborative robots (cobots[1]) has had a high impact on the industrial world [42]. Thanks to their lightweight architecture, the smooth silhouettes of their bodies, and the presence of force and torque sensors aboard, cobots can be easily installed in factories without altering the preexisting layout. Additionally, the fenceless installation allows cobots to safely work in cooperation with human workers and enables them to be mounted on *Autonomous*

[1]The term “cobot” was coined in 1996 by J. E. Colgate and M. Peshkin, professors at Northwestern University, who issued a homonym US patent in 1997, in which they described this new technology as "An apparatus and method for direct physical interaction between a person and a general purpose manipulator controlled by a computer."

Mobile Robots (AMRs), freely moving around to perform different tasks.
In [43, 44], a combination of a serial manipulator and a mobile platform, both characterized by collaborative features, was employed to automatize the process of raw-material feeding to a tea-packaging automatic machine. The authors showed how an ad-hoc architecture and industrial components could represent an efficient and saleable solution for accomplishing machine-tending tasks.
The integration of a lightweight two-armed robot with a mobile omnidirectional base was achieved in [45], with the objective of picking up items from shelves within a grocery store. Similarly, an anthropomorphic mobile robot was studied for bin-picking operations in [46].
Within the COORSA project [47–49], a mobile manipulation system was employed for the accomplishment of depalletizing/palletizing tasks. In particular, the combination of an autonomous platform, a collaborative robotic arm and a lifting device was used to drag items aboard the mobile vehicle.
In the work presented in [50], a forklift *Automated Guided Vehicle* (AGV) with a cobot onboard was used for the autonomous picking of goods, their placement on a EUR pallet, and the transport of the pallet towards a final destination.
The Robo-Partner EU project [51] combined the perception and dexterity of human operators with the robot strength, repeatability, and precision, in *human-robot cooperation* (HRC) scenario, aimed at the assembly of the rear axle of a passenger vehicle.
In the Valeri EU project [52], an omnidirectional platform supplemented with a rotating vertical axis on which a lightweight manipulator was mounted, was tested for the aerospace industry.
The mobile robotic system presented in [41] was tested in automotive factories, performing prototypical assembly tasks on approximately 200 cars.
As seen, the installation of cobots on mobile platforms has become current practice, not only in the academia, but also in industrial use cases. A single mobile robot can replace many stationary robots that would otherwise operate only for short periods, hence justifying such an investment [43].
However, when the object that has to be manipulated is heavy and bulky (e.g. car chassis parts or engine components), a single cobot may not be enough, due to its low payload capabilities. To overcome this drawback, a collaborative dual-arm setup, in which two cobots, mounted on a mobile platform, hold the same object, can be taken into account. The dual-arm manipulation employing two collaborative robots is an object of this PhD research and will be addressed in this Chapter.

4.2 Dual-arm Investigation

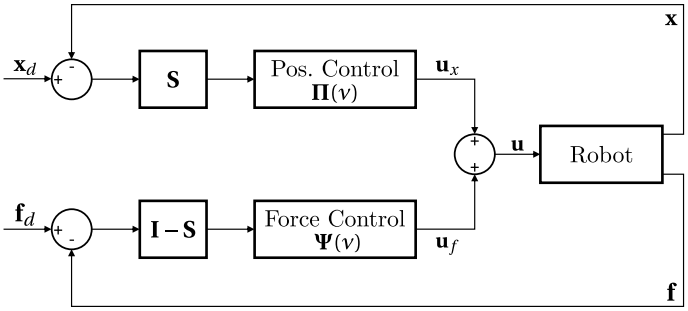
The study of dual-arm manipulation is characterized by challenges that may not be present in the single-arm manipulation case, hence lying at a higher degree of complexity. This higher complexity results in the need for more advanced system integration (e.g., force/torque sensors), together with viable control approaches.
The first aspect that has to be considered regards the type of interaction between the end-effectors of the manipulators and the object. Typically, two are the types of cooperative tasks [53]:

- *Cooperative robot manipulators holding an object with fixed grasp points*: the object is rigidly grasped by each robot, e.g. using a gripper; this implies that no

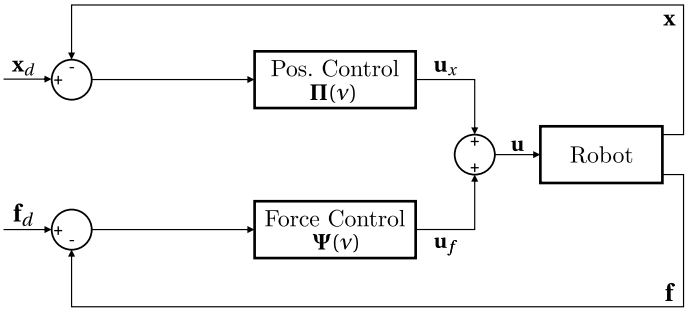
relative motion can occur in the interaction points;

- *Cooperative robot manipulators holding an object through contact points or contact areas*: relative motion between the end-effectors and the object is enabled, due to the fact that the contact constraints are unilateral; in this case, a pulling force results in slipping of the object and hence in contact loss.

The latter strategy, which will be hereafter indicated as *cooperative grasping*, has some advantages compared to the one employing grippers, namely the minor investment costs that the installation of two grippers would require and the higher flexibility w.r.t. uncertainties in the object shape and size. Grabbing an object with a gripper requires *ad hoc* geometric features, and the object needs to be small enough to be hosted within the stroke of the gripper fingers.
In [54], cooperative grasping was pursued by adopting the impedance control of two industrial manipulators carrying the object; experiments were performed and discussed.
In [55], the impedance control of two 7-DOF collaborative robots was implemented at three levels: at the object level to control the interaction forces between the object and the environment, at the end-effector level, to control the internal forces exerted on the object, and finally at body level, to limit contact between the robots and the environment. The algorithm was verified only in a simulation scenario.
The work in [56] addressed the case of a rigidly grasped object by proposing an adaptive hybrid intelligent control, based on multi-input multi-output fuzzy logic, to be independent from the knowledge of the system dynamics; results from simulations were provided.
A primary-secondary strategy was adopted in [57], where the left robot was position controlled and the right one was controlled through a force/position scheme; the motions were performed with low dynamics on objects with different shapes. Faster trajectories were executed in [58], with the two collaborative robots being admittance-controlled.
More recently, the authors of [59] addressed the problem of time-optimal cooperative grasping, by adopting the admittance control of two industrial robots.
In the scenario of cooperative grasping, the correct accomplishment of dual-arm manipulation is achieved if a stable grasp is ensured, i.e. no slipping of the object occurs. To this aim, the normal force and the tangential forces that the two robots exert on the object must be inside the static-friction cone. This is granted if an internal force (which does not contribute to the object motion) in the normal direction of the contact is taken into account.
In [59] the time-optimal trajectory planning of both manipulators was carried out, to perform the dual-arm task in minimal time. The internal force, aimed at enabling the stability of the cooperative grasping, was achieved by prescribing a constant *virtual penetration* on the object, whose value however was not optimized. The experiments showed that slipping could occur for high-dynamics motion of the object.
In our work, the virtual penetration used to generate the internal force and the path parameter defining the trajectory of the object are chosen as independent variables of the optimization. Additionally, the friction-cone constraint is inserted within the formulation of the time-optimal trajectory planning. This way the resolution of the problem leads to the optimal motion profile that the two robots have to follow to keep the contact forces inside the friction cone, while at the same time guaranteeing the shortest trajectory duration compatible with this constraint. The structure of this Chapter is as



(a) Hybrid force/position control scheme.



(b) Parallel force/position control scheme.

Figure 4.1: Schemes of the direct force control.

follows. Section 4.3 gives a brief survey about direct and indirect force control of serial robots. Section 4.4 describes the model employed for the determination of the forces exerted on the object. Section 4.5 depicts the constrained optimization problem, highlighting how the forces that are used to formulate the friction-cone inequalities are written in terms of the input variables. Section 4.6 illustrates the setup used for the experiments, together with the obtained results and the corresponding quantitative analysis.

4.3 Direct/Indirect Robot Force Control

Many robotics applications require the manipulator to interact with the external environment (e.g., pushing, grinding, polishing, etc.). In such cases, the robot not only needs to follow the prescribed path, but it also has to comply with the environment [60]. Compliant control aims to modulate the interaction between the robot and the environment to avoid the arising of large, unacceptable contact forces. This represents a keynote for the accomplishment of dual-arm manipulation, where, in the same manner as two human operators have to compliantly adjust their motion one to another while commonly lifting a weight, each of the two robots cannot apply a force on the manipulated object without being concerned of the action made by the other one.
Compliant control has evolved from hybrid to parallel force/position control and to impedance and admittance control [61].
To better understand the meaning of impedance and admittance, a 2^{nd}-order system described by a mass m_d, a stiffness k_d, and a damping c_d, can be considered. The impedance Υ and the admittance A can be seen as the equivalent dynamic quantities

of stiffness and compliance for the static relation, namely:

$$\Upsilon(v) = A(v)^{-1} = \frac{\Delta f(v)}{\Delta x(v)} = m_d v^2 + c_d v + k_d \tag{4.1}$$

with v indicating the Laplace variable, $\Delta f(v) = f_d(v) - f(v)$ and $\Delta x(v) = x_d(v) - x(v)$ being the Laplace transforms of the force and position errors, between the desired quantity (denoted by the subscript d) and the actual measured one (without subscript).
Compliant control can be divided into two categories:

- *direct force control*, in which the force feedback loop is closed by a force controller;
- *indirect force control*, where the force control is achieved by means of a motion control.

As far as direct force control is taken into account, two approaches are available:

- **hybrid force/position control** [62]: the space is divided into complementary orthogonal subspaces by the selection matrix $\mathbf{S} = \text{diag}([s_1, \ldots, s_n]^T)$, with n being the number of controlled DOFs [39]. The directions along which the motion is constrained ($s_j = 0$) must be force-controlled, whereas the free directions ($s_j = 1$) need to be motion-controlled. This approach, while granting a reliable tracking of the desired force and position within the respective subspaces, requires detailed knowledge of the environment and its applicability becomes complex in the case of non-planar contact surfaces. The control scheme is depicted in Figure 4.1a, where the hybrid command $\mathbf{u}(v)$ is

$$\mathbf{u}(v) = \mathbf{S}\,\mathbf{\Pi}(v)[\mathbf{x}_d(v) - \mathbf{x}(v)] + [\mathbf{I} - \mathbf{S}]\,\mathbf{\Psi}(v)[\mathbf{f}_d(v) - \mathbf{f}(v)], \tag{4.2}$$

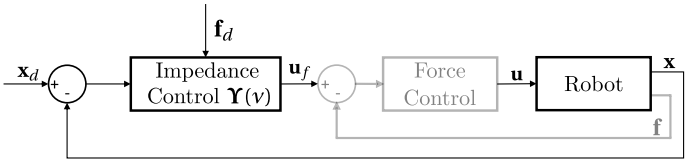
  with $\mathbf{\Pi}(v)$ and $\mathbf{\Psi}(v)$ being the Laplace transforms of the position and force control loops, respectively; $\mathbf{x}_d$ and $\mathbf{x}$ represent the desired and measured robot end-effector position; $\mathbf{f}_d$ and $\mathbf{f}$ are the desired and measured force exerted by the robot on the environment; finally $\mathbf{I} \in \mathbb{R}^{n \times n}$ is the identity matrix;

- **parallel force/position control** [63, 64]: unlike hybrid force/position control, in parallel force/position control the outputs of the force and motion controllers are superimposed, hence eventually acting along the same directions. The advantages of this approach are represented by a reliable tracking of the desired force and position, together with robustness in the presence of environment model uncertainties and planning errors; however, this technique grant slower dynamics w.r.t. to the hybrid one. In the control scheme (see Figure 4.1b) the parallel command is computed as

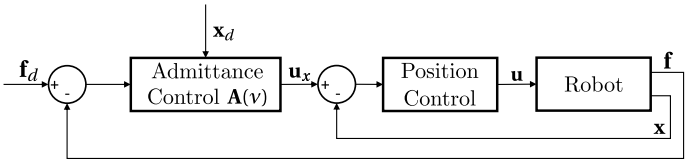
$$\mathbf{u}(v) = \mathbf{\Pi}(v)[\mathbf{x}_d(v) - \mathbf{x}(v)] + \mathbf{\Psi}(v)[\mathbf{f}_d(v) - \mathbf{f}(v)], \tag{4.3}$$

  with the employed quantities having the same meaning as the ones in Equation (4.2).

For indirect force control, two complementary strategies can be considered:



(a) Implicit (black lines) and explicit (black and gray lines) impedance control scheme.



(b) Admittance control scheme.

Figure 4.2: Schemes of the indirect force control.

- **impedance control** [65]: in the case of implicit impedance control (only black lines of Figure 4.2a), the position error $\Delta\mathbf{x}$ is combined with the desired force $\mathbf{f}_d$ to obtain the command on the force $\mathbf{u}_f$:

$$\mathbf{u}_f(\nu) = \boldsymbol{\Upsilon}(\nu)[\mathbf{x}_d(\nu) - \mathbf{x}(\nu)] + \mathbf{f}_d(\nu). \tag{4.4}$$

  The command $\mathbf{u}_f$ represents the force that the robot end-effector needs to apply on the environment to cancel the position error and track the value of the desired force.
  This approach does not require force measurement; however, its implicitness is given by the need to control the robot through a force command, which may not be a standard feature of the employed manipulator. For this reason, an inner force control loop is usually inserted within the control scheme (black and gray lines of Figure 4.2a), to achieve an explicit impedance control;

- **admittance control**: the force error $\Delta\mathbf{f}$ is combined with the desired position $\mathbf{x}_d$ to obtain the command on the motion $\mathbf{u}_x$ (see Figure 4.2b):

$$\mathbf{u}_x(\nu) = \mathbf{A}(\nu)[\mathbf{f}_d(\nu) - \mathbf{f}(\nu)] + \mathbf{x}_d(\nu). \tag{4.5}$$

  The output of the admittance block (i.e. $\mathbf{u}_x$) is the correction on the motion that the robot needs to perform to cancel the force error and follow the desired trajectory $\mathbf{x}_d$.
  The command $\mathbf{u}_x$ can be directly fed as input to the manipulator or it can be employed as input of an inner position control, providing the so-called explicit admittance control. In both cases, the manipulator receives a motion command, which represents a standard feature of industrial or collaborative robots.

When cooperative grasping has to be performed, the choice of the force-control strategy to be pursued is crucial. With the perspective of achieving maximum flexibility in the manipulation task, objects of different sizes and shapes are the main target, hence not granting detailed knowledge of their characteristics. This randomness excludes the hybrid force/position control approach. On the other hand, the robustness to uncertainties ensured by the parallel force/position control can represent a viable alternative. In this scenario, the two manipulators cannot act at the same hierarchy

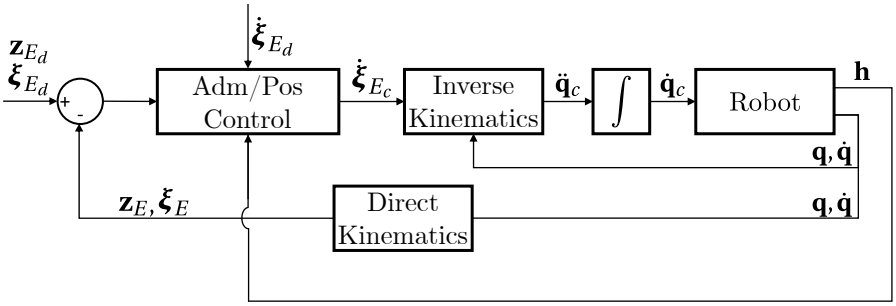


Figure 4.3: Admittance control scheme adopted for the dual-arm manipulation.

level, but a primary-secondary strategy has to be foreseen. This implies that one of the two manipulators (*primary*) is position-controlled, whereas the other one (*secondary*) is controlled through parallel force/position control. In the case of cooperative grasping, the stability of the contact may be invalidated by the fact that the primary robot only follows the desired trajectory and the secondary one has to chase it and simultaneously apply an internal force on the object to avoid slipping.
For this reason, in the addressed work, attention is given to indirect force control, which allows the two manipulators to be controlled at the same hierarchy level. In particular, since the equipment of a force sensor does not represent a discriminant (both admittance control and explicit impedance control require force measurement), admittance control is chosen. Admittance control provides the direct translation of a force input into a command at motion level, which represents standard practice in robot control. Conversely, impedance control would need an additional inner force loop (gray lines in Figure 4.2a) to convert the obtained force command in a command that can be eventually fed to the robot.
The adopted scheme is reported in Figure 4.3: the same controller is applied to both manipulators. For the sake of clarity, subscipts referring to one or the other robot are omitted. The first loop is a combination of admittance and position control. The input is represented by the desired end-effector pose $\mathbf{z}_{E_d} = \begin{bmatrix}\mathbf{r}_{E_d}^T & \mathcal{Q}_{E_d}^T\end{bmatrix}^T$, with $\mathcal{Q}_{E_d}^T$ being the quaternion representing the desired rotation matrix $\mathbf{R}_{E_d}$, the end-effector twist $\boldsymbol{\xi}_{E_d} = \begin{bmatrix}\dot{\mathbf{r}}_{E_d}^T & \boldsymbol{\omega}_{E_d}^T\end{bmatrix}^T$ and the first-time derivative of the twist $\dot{\boldsymbol{\xi}}_{E_d} = \begin{bmatrix}\ddot{\mathbf{r}}_{E_d}^T & \dot{\boldsymbol{\omega}}_{E_d}^T\end{bmatrix}^T$. The feedback on the measured end-effector wrench $\mathbf{h}$ and on the actual kinematics of the robot end-effector $\mathbf{z}_E = \begin{bmatrix}\mathbf{r}_E^T & \mathcal{Q}_E^T\end{bmatrix}^T$, $\boldsymbol{\xi}_E = \begin{bmatrix}\dot{\mathbf{r}}_E^T & \boldsymbol{\omega}_E^T\end{bmatrix}^T$, is employed to obtain the motion command $\dot{\boldsymbol{\xi}}_{E_c}$. Note that the desired force is implicitly inserted within the desired position $\mathbf{r}_{E_d}$ through the concepts of *virtual penetration* and *contact stiffness*, as it will be clarified at the end of Section 4.4. The actual values of $\mathbf{z}_E$ and $\boldsymbol{\xi}_E$ are obtained online by computing the direct kinematics of the robot [66]. The control law of this first block is:

$$\dot{\boldsymbol{\xi}}_{E_c} = \mathbf{M}_E^{-1}(\mathbf{D}_E \Delta\boldsymbol{\xi}_E + \mathbf{K}_E \Delta\mathbf{z}_E - \mathbf{h}) + \dot{\boldsymbol{\xi}}_{E_d} \tag{4.6}$$

where $\Delta\mathbf{z}_E = \begin{bmatrix}(\mathbf{r}_{E_d} - \mathbf{r}_E)^T & \boldsymbol{\epsilon}_d^T\end{bmatrix}^T$ denotes the pose error of the end-effector, with $\boldsymbol{\epsilon}_d$ being the vectorial part of the quaternion $\mathcal{Q}_{E_d,E}$ extracted from $\mathbf{R}_E^T\mathbf{R}_{E_d}$ [67], [68] ($\mathbf{R}_{E_d}$ and $\mathbf{R}_E$ are the desired and actual rotation matrices of the robot end-effector w.r.t. the base frame, respectively). The error on the end-effector twist is expressed as $\Delta\boldsymbol{\xi}_E = \begin{bmatrix}(\dot{\mathbf{r}}_{E_d} - \dot{\mathbf{r}}_E)^T & (\boldsymbol{\omega}_{E_d} - \boldsymbol{\omega}_E)^T\end{bmatrix}^T$.
Note that there is no direct correspondence between the control schemes of Figures 4.2b and 4.3. Indeed, since the normal force that the robots have to exert on the object is prescribed by using a virtual penetration on the object, this combined view of force

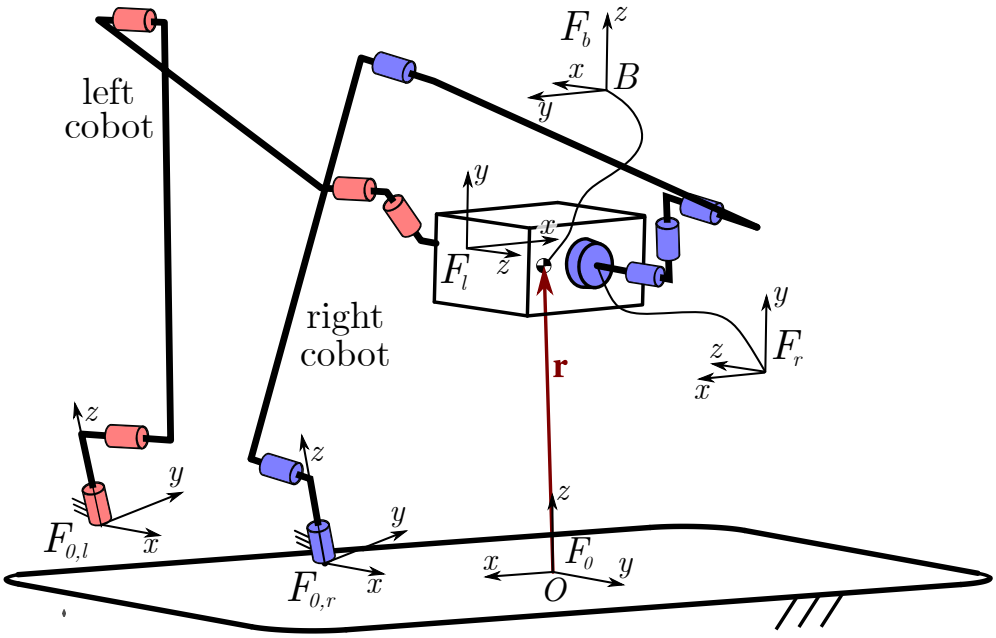


Figure 4.4: General scheme of the dual-arm cooperative grasping.

exertion is reflected in the control scheme indicated as "Adm/Pos Control". In the latter, the input is represented by the desired motion $\mathbf{z}_{E_d}, \boldsymbol{\xi}_{E_d}, \dot{\boldsymbol{\xi}}_{E_d}$, where the desired force is hidden inside the variable $\mathbf{z}_{E_d}$. Through the feedback on the measured wrench $\mathbf{h}$, the force error is computed by subtraction between the part of $\mathbf{z}_{E_d}$ expressing the desired force and the measured one. Instead of using the latter result and adding it to the desired motion to obtain an intermediate command (as it is done for $\mathbf{u}_x$ in Equation (4.5)) to feed to the position-control loop (see Figure 4.2b), the position and velocity errors are directly taken into account at the same stage of the admittance control. Matrices $\mathbf{M}_E$, $\mathbf{D}_E$, $\mathbf{K}_E$ are diagonal matrices, whose values can be conveniently tuned to achieve the desired compliant behavior. In particular, the matrix $\mathbf{M}_E$ is representative of the object inertia, whereas $\mathbf{D}_E$ and $\mathbf{K}_E$ have to replicate the damping and stiffness desired behavior. Trial-and-error procedures can be performed to obtain the matrix values that grant the zeroing of the position error at the end of the motion. The commanded motion, expressed by $\dot{\boldsymbol{\xi}}_{E_c}$, is transformed in joint space through the robot inverse kinematics, by computing the robot Jacobian matrix $\mathbf{J}(\mathbf{q})$ and its time derivative $\dot{\mathbf{J}}(\mathbf{q}, \dot{\mathbf{q}})$ in correspondence of the actual configuration and velocities of the robot joints $\mathbf{q}, \dot{\mathbf{q}}$ (measured by the encoders). This gives the commanded joint accelerations

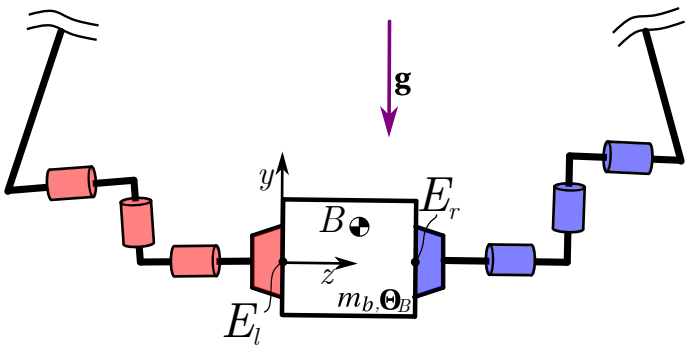
$$\ddot{\mathbf{q}}_c = \mathbf{J}^{-1}(\mathbf{q})\left[\dot{\boldsymbol{\xi}}_{E_c} - \dot{\mathbf{J}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}\right]. \tag{4.7}$$

Finally, an integrator is used to compute the commanded joint velocities $\dot{\mathbf{q}}_c$ to feed the robot.

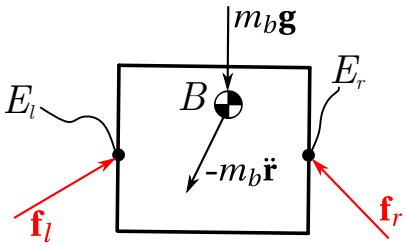
4.4 Force Model

The object motion is prescribed in terms of the desired trajectory given by:

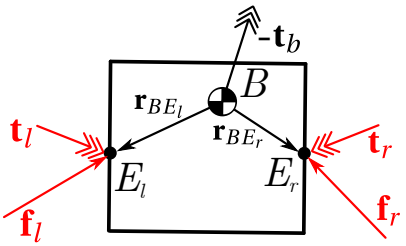
- the position $\mathbf{r}(t)$, where $\mathbf{r}$ indicates the position vector connecting the origin O of the inertial frame F_0 to the object *centre of mass* (COM) B (Figure 4.4);



(a) Front view of the cooperative grasping.



(b) Scheme of the object translational equilibrium.



(c) Scheme of the object rotational equilibrium.

Figure 4.5: Schemes employed for the equilibrium of the object.

- the orientation $\mathbf{R}(t)$, with $\mathbf{R}$ being the rotation matrix between the frame F_0 and the frame F_b with origin in B and attached to the object (Figure 4.4).

The object motion is achieved as long as the following net wrench $\mathbf{h}_b = [\mathbf{f}_b^T \ \mathbf{t}_b^T]^T$ is applied on the object:

$$\mathbf{f}_b = -m_b\mathbf{g} + m_b\ddot{\mathbf{r}} \tag{4.8a}$$

$$\mathbf{t}_b = \boldsymbol{\Theta}_B\dot{\boldsymbol{\omega}} + \tilde{\boldsymbol{\omega}}\boldsymbol{\Theta}_B\boldsymbol{\omega} \tag{4.8b}$$

where m_b is the mass of the object and $\boldsymbol{\Theta}_B$ is the inertia matrix w.r.t. the COM B, $\mathbf{g}$ is the gravity acceleration, $\ddot{\mathbf{r}} = \frac{d^2}{dt^2}\mathbf{r}$ and $\dot{\boldsymbol{\omega}} = \frac{d}{dt}\boldsymbol{\omega}$ are the object translational and angular accelerations, respectively, and the symbol $\sim$ denotes the skew-symmetric representation of a 3-dimensional vector. Due to the equilibrium of the object, the wrenches $\mathbf{h}_l = [\mathbf{f}_l^T \ \mathbf{t}_l^T]^T$ and $\mathbf{h}_r = [\mathbf{f}_r^T \ \mathbf{t}_r^T]^T$ exerted by the two cobots on the object must produce the desired net wrench, namely (Figure 4.5):

$$\mathbf{f}_l + \mathbf{f}_r = -m_b\mathbf{g} + m_b\ddot{\mathbf{r}} \tag{4.9a}$$

$$\mathbf{t}_l + \mathbf{t}_r + \tilde{\mathbf{r}}_{BE_l}\mathbf{f}_l + \tilde{\mathbf{r}}_{BE_r}\mathbf{f}_r = \boldsymbol{\Theta}_B\dot{\boldsymbol{\omega}} + \tilde{\boldsymbol{\omega}}\boldsymbol{\Theta}_B\boldsymbol{\omega} \tag{4.9b}$$

Note that the subscripts l and r stand for *left* and *right*, respectively. The grasp kinetostatics of the cobots on the object can be written in compact form:

$$\mathbf{W}\mathbf{h}_{l,r} = \mathbf{h}_b \tag{4.10}$$

where $\mathbf{h}_{l,r}$ is the vector $\mathbf{h}_{l,r} = [\mathbf{h}_l^T \ \mathbf{h}_r^T]^T = [\mathbf{f}_l^T \ \mathbf{t}_l^T \ \mathbf{f}_r^T \ \mathbf{t}_r^T]^T \in \mathbb{R}^{12}$ and $\mathbf{W} \in \mathbb{R}^{6\times 12}$ is the grasp matrix

$$\mathbf{W} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \tilde{\mathbf{r}}_{BE_l} & \mathbf{I} & \tilde{\mathbf{r}}_{BE_r} & \mathbf{I} \end{bmatrix} \tag{4.11}$$

Given the hyperstatic nature of the problem, which is characterized by the 12 scalar unknowns $\mathbf{h}_{l,r}$ contained in the 6 scalar equations (4.11), its resolution is under-determined, namely:

$$\mathbf{h}_{l,r} = \mathbf{W}^{\perp}\mathbf{h}_b + \mathbf{V}\mathbf{h}_p \tag{4.12}$$

Here, $\mathbf{W}^{\perp} = \mathbf{G}\mathbf{W}^T(\mathbf{W}\mathbf{G}\mathbf{W}^T)^{-1} \in \mathbb{R}^{12\times 6}$ is a pseudo-inverse of $\mathbf{W}$ [39], whose weighting matrix is $\mathbf{G}$, and $\mathbf{V} \in \mathbb{R}^{12\times 6}$ is an orthogonal complement matrix such that $\mathbf{W}\mathbf{V} = \mathbf{0}$. Additionally, $\mathbf{h}_p = [\mathbf{f}_p^T\ \mathbf{t}_p^T]^T$ represents a desired penetrating wrench that generates internal prestressing. The way the two cobots distribute the wrenches $\mathbf{h}_l$ and $\mathbf{h}_r$ on the object is influenced by the weighting matrix $\mathbf{G}$, whose choice has to be done to avoid the arising of null-space components. For this reason, the employed weighting matrix $\mathbf{G}$ is [69]

$$\mathbf{G} = \begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \end{bmatrix} \tag{4.13}$$

and consequently the pseudo-inverse $\mathbf{W}^{\perp}$ results in

$$\mathbf{W}^{\perp} = \begin{bmatrix} \frac{1}{2}\mathbf{I} & \mathbf{0} \\ -\frac{1}{2}\tilde{\mathbf{r}}_{BE_l} & \frac{1}{2}\mathbf{I} \\ \frac{1}{2}\mathbf{I} & \mathbf{0} \\ -\frac{1}{2}\tilde{\mathbf{r}}_{BE_r} & \frac{1}{2}\mathbf{I} \end{bmatrix} \tag{4.14}$$

The orthogonal complement matrix $\mathbf{V}$ can be chosen as

$$\mathbf{V} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\tilde{\mathbf{r}}_{BE_l} & \mathbf{I} \\ -\mathbf{I} & \mathbf{0} \\ \tilde{\mathbf{r}}_{BE_r} & -\mathbf{I} \end{bmatrix} \tag{4.15}$$

From Equations (4.12), (4.14) and (4.15), the expressions of $\mathbf{f}_l$, $\mathbf{t}_l$, $\mathbf{f}_r$ and $\mathbf{t}_r$ can be isolated, i.e.

$$\mathbf{f}_l = \frac{1}{2}(-m_b\mathbf{g} + m_b\ddot{\mathbf{r}}) + \mathbf{f}_p \tag{4.16a}$$

$$\mathbf{t}_l = \frac{1}{2}(\boldsymbol{\Theta}_B\dot{\boldsymbol{\omega}} + \tilde{\boldsymbol{\omega}}\boldsymbol{\Theta}_B\boldsymbol{\omega}) - \tilde{\mathbf{r}}_{BE_l}\left(\frac{1}{2}\mathbf{f}_b + \mathbf{f}_p\right) + \mathbf{t}_p \tag{4.16b}$$

$$\mathbf{f}_r = \frac{1}{2}(-m_b\mathbf{g} + m_b\ddot{\mathbf{r}}) - \mathbf{f}_p \tag{4.16c}$$

$$\mathbf{t}_r = \frac{1}{2}(\boldsymbol{\Theta}_B\dot{\boldsymbol{\omega}} + \tilde{\boldsymbol{\omega}}\boldsymbol{\Theta}_B\boldsymbol{\omega}) - \tilde{\mathbf{r}}_{BE_r}\left(\frac{1}{2}\mathbf{f}_b - \mathbf{f}_p\right) - \mathbf{t}_p \tag{4.16d}$$

Equations (4.16) represent an equal distribution of the reactions among the cobot end-effectors. In particular, Equations (4.16a) and (4.16c) imply that each cobot may apply half of the net wrench $\mathbf{f}_b$ to make the object follow its desired trajectory, setting aside the prestress $\mathbf{f}_p$. Regarding Equations (4.16b) and (4.16d), in general, no internal grasp torque is desired, implying that $\mathbf{t}_p$ can be set to zero, namely $\mathbf{t}_p = \mathbf{0}$. In addition, it can be stated that the torques $\mathbf{t}_l$ and $\mathbf{t}_r$ exerted by the cobots on the object are not involved in the friction-cone analysis. For this reason, only the exerted forces $\mathbf{f}_l$ and $\mathbf{f}_r$ will be hereafter taken into account. In the case of a pure-translation motion of the object, the

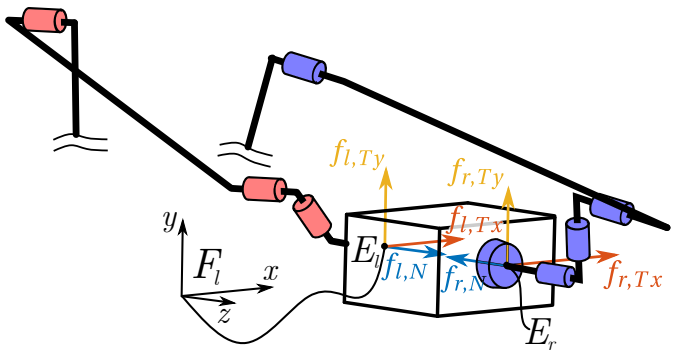


Figure 4.6: Projection of the forces $\mathbf{f}_l$ and $\mathbf{f}_r$ on the coordinate frame F_l.

angular velocity and acceleration are kept equal to zero, i.e. $\boldsymbol{\omega} = \dot{\boldsymbol{\omega}} = \mathbf{0}$. In this scenario, a suitable coordinate frame F_l attached to the object can be chosen. The origin of F_l is in correspondence of E_l, the interface point between the left robot and the object. F_l is oriented such that its y-axis is always along the vertical direction (this is possible since the orientation of the object is kept constant throughout the motion), and the z-axis points towards the pushing direction of the left robot on the object (Figure 4.5a). Accordingly, the projection of vector $\mathbf{g}$ on F_l is $\{\mathbf{g}\}_l = [0 \; -g \; 0]^T$. For the sake of simplicity, we can assume that the only component of the internal prestress which is other than zero is the one in the z-axis direction of frame F_l, thus resulting in $\{\mathbf{f}_p\}_l = [0 \; 0 \; f_p]^T$. Furthermore, the projections of the forces $\mathbf{f}_l$ and $\mathbf{f}_r$ on F_l are composed of the tangential forces along the x and y directions and by the normal force along the z-axis. For the sake of clarity, taking advantage of the illustration in Figure 4.6, these projections will be indicated as:

$$\{\mathbf{f}_l\}_l = \begin{bmatrix} f_{l,Tx} \\ f_{l,Ty} \\ f_{l,N} \end{bmatrix}, \quad \{\mathbf{f}_r\}_l = \begin{bmatrix} f_{r,Tx} \\ f_{r,Ty} \\ f_{r,N} \end{bmatrix} \tag{4.17}$$

By exploiting the definitions in Equation (4.17), the projections of Equations (4.16a) and (4.16c) on F_l yield:

$$\{\mathbf{f}_l\}_l = \begin{bmatrix} f_{l,Tx} \\ f_{l,Ty} \\ f_{l,N} \end{bmatrix} = \frac{m_b}{2} \begin{bmatrix} \ddot{r}_x \\ g + \ddot{r}_y \\ \ddot{r}_z \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ f_p \end{bmatrix} \tag{4.18a}$$

$$\{\mathbf{f}_r\}_l = \begin{bmatrix} f_{r,Tx} \\ f_{r,Ty} \\ f_{r,N} \end{bmatrix} = \frac{m_b}{2} \begin{bmatrix} \ddot{r}_x \\ g + \ddot{r}_y \\ \ddot{r}_z \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ f_p \end{bmatrix} \tag{4.18b}$$

The presented force model holds as far as none of the two robot loses contact with the manipulated object, i.e. there is no slipping of the object. This is not sure in advance, as, if the internal prestress is not taken into account ($f_p = 0$N), the resolution of Equations (4.18) may lead to a contact force $\mathbf{f}_j$ (with $j = l, r$) pulling away from the object. This is the main reason why the value of f_p must be chosen to ensure a stable grasp on the object. In other words, f_p plays a role in granting that the friction-cone condition is satisfied on both sides of the contact, namely:

$$\sqrt{f_{j,Tx}^2 + f_{j,Ty}^2} \leq \mu |f_{j,N}|, \quad j = l, r \tag{4.19}$$

with μ being the static friction coefficient between the cobot extremities and the object.

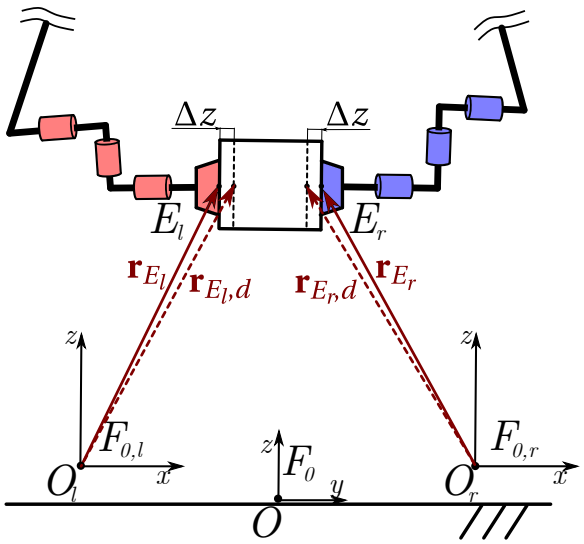


Figure 4.7: Physical meaning of the virtual penetration Δz.

In general, when addressing robot indirect force control, prescribing a specified interaction force can be translated into requiring the robot to virtually go beyond the contact surface of the manipulated object [39]. Considering the sole position variables as characterizing the robot operational space, Fig. 4.7 shows the condition above. Indicating with $\mathbf{r}_{E_j}$ $(j = l, r)$ the reference position of E_j, if the robot desired target $\mathbf{r}_{E_j,d}$ coincides with $\mathbf{r}_{E_j}$ at position level, the robot is asked to apply no force on the object (i.e. $\mathbf{f}_p = \mathbf{0}$). Conversely, if the target position $\mathbf{r}_{E_j,d}$ is beyond the contact surface, the robot is obliged to exert a certain internal prestress on the object (i.e. $\mathbf{f}_p \neq \mathbf{0}$).
According to the elastic model, the internal prestress depends on the difference between the desired position $\mathbf{r}_{E_j,d}$ and the reference one $\mathbf{r}_{E_j}$:

$$\mathbf{f}_p = \mathbf{K}(\mathbf{r}_{E_j,d} - \mathbf{r}_{E_j}) \tag{4.20}$$

where $\mathbf{K}$ represents the matrix of contact stiffness at the interface. In the scenario analyzed in this research, given the absence of internal prestress along the tangential directions ($f_{p,x} = f_{p,y} = 0$N), the command of going beyond the contact surface only applies along the z-axis, according to a virtual penetration Δz. This allows expressing $\mathbf{r}_{E_j,d}$ $(j = l, r)$ in the corresponding end-effector frames F_l and F_r, with the latter having the origin at the interface point E_r and axes oriented as in Figure 4.4:

$$\{\mathbf{r}_{E_j,d}\}_j = \{\mathbf{r}_{E_j}\}_j + [0 \quad 0 \quad \Delta z]^T, \quad j = l, r \tag{4.21}$$

Substituting (4.21) into ((4.20), the internal prestress along the normal direction can be written as a function of the virtual penetration Δz, namely

$$f_p = k\Delta z \tag{4.22}$$

where the value of the penetration stiffness k can be experimentally identified.

4.5 Time-Optimal Trajectory Planning

4.5.1 Trajectory Definition

The path and the orientation of the object are prescribed. In particular, the object orientation is constant, so that its angular velocity and acceleration are equal to zero dur-

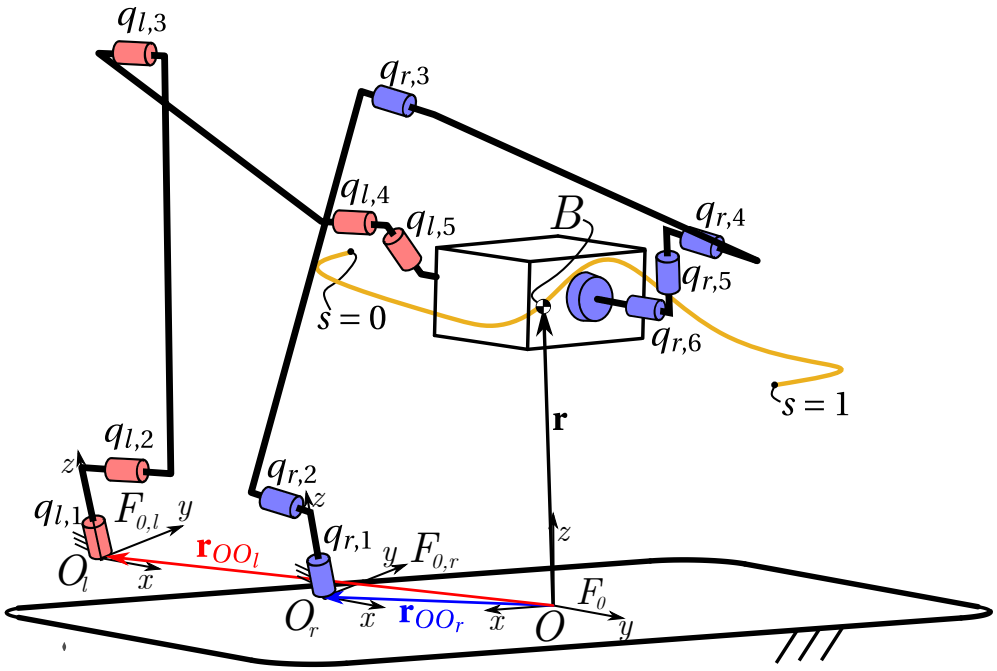


Figure 4.8: Scheme for a general dual-arm trajectory planning.

ing the entire motion, namely $\boldsymbol{\omega} = \boldsymbol{\alpha} = \mathbf{0}$. The path of the object COM is parameterized in terms of a parameter s (arc length):

$$\mathbf{r} = \mathbf{r}(s), \qquad s \in [0,1] \tag{4.23}$$

and can be defined by using B-splines [35], [32] as

$$\mathbf{r}(s) = \sum_{j=0}^{m} B_j^d(s)\mathbf{p}_j, \qquad s \in [0,1] \tag{4.24}$$

where B_j^d are the B-spline basis functions of degree d and $\mathbf{p}_j$ are $m+1$ control points. The motion law of the path parameter $s(t)$ allows the trajectory to be defined:

$$\dot{\mathbf{r}}(s,\dot{s}) = \mathbf{r}'(s)\dot{s} \tag{4.25}$$

$$\ddot{\mathbf{r}}(s,\dot{s},\ddot{s}) = \mathbf{r}''(s)\dot{s}^2 + \mathbf{r}'(s)\ddot{s} \tag{4.26}$$

with $()' = \partial()/\partial s$ denoting the derivative w.r.t. the path parameter s.
Assuming that both cobots do not lose contact with the object, the reference position of their end-effectors can be obtained as:

$$\mathbf{r}_{E_j}(s) = \mathbf{r}(s) + \mathbf{r}_{BE_j} - \mathbf{r}_{OO_j}, \qquad j = l,r \tag{4.27}$$

where $\mathbf{r}_{BE_j}$ is the position vector from the object COM B to the corresponding interface point E_j ($j = l,r$) and $\mathbf{r}_{OO_j}$ is the position vector connecting the origin of F_0 with the origin of F_j, which is the base frame of the k-th ($j = l,r$) cobot (v. Figure 4.8). Since the prescribed motion foresees a constant orientation of the object, the velocity and the acceleration of the cobot end-effector reference points are:

$$\dot{\mathbf{r}}_{E_j}(s,\dot{s}) = \dot{\mathbf{r}}(s,\dot{s}), \qquad j = l,r \tag{4.28}$$

$$\ddot{\mathbf{r}}_{E_j}(s,\dot{s},\ddot{s}) = \ddot{\mathbf{r}}(s,\dot{s},\ddot{s}), \qquad j = l,r \tag{4.29}$$

4.5.2 Force Limits

Once the motion of the object and the cobot end-effectors is written in terms of the path parameter s and its time derivatives $\dot{s}, \ddot{s}$, the modeled forces (v. Equations (4.18a), (4.18b)) can be expressed as a function of the same motion law, that is:

$$\begin{bmatrix} f_{l,Tx}(s,\dot{s},\ddot{s}) \\ f_{l,Ty}(s,\dot{s},\ddot{s}) \\ f_{l,N}(s,\dot{s},\ddot{s},\Delta z) \end{bmatrix} = \frac{m_b}{2} \begin{bmatrix} \ddot{r}_x(s,\dot{s},\ddot{s}) \\ g + \ddot{r}_y(s,\dot{s},\ddot{s}) \\ \ddot{r}_z(s,\dot{s},\ddot{s}) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ k\Delta z \end{bmatrix} \tag{4.30a}$$

$$\begin{bmatrix} f_{r,Tx}(s,\dot{s},\ddot{s}) \\ f_{r,Ty}(s,\dot{s},\ddot{s}) \\ f_{r,N}(s,\dot{s},\ddot{s},\Delta z) \end{bmatrix} = \frac{m_b}{2} \begin{bmatrix} \ddot{r}_x(s,\dot{s},\ddot{s}) \\ g + \ddot{r}_y(s,\dot{s},\ddot{s}) \\ \ddot{r}_z(s,\dot{s},\ddot{s}) \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ k\Delta z \end{bmatrix} \tag{4.30b}$$

By substituting the expressions of Equations (4.30) inside the friction-cone inequality (4.19), the no-slipping conditions for both cobots can be written as functions of the motion law $(s, \dot{s}, \ddot{s})$ and the virtual penetration Δz:

$$f_{l,Tx}^2 + f_{l,Ty}^2 \leq \mu^2 \left(\frac{m_b}{2} \ddot{r}_z + k\Delta z \right)^2 \tag{4.31a}$$

$$f_{r,Tx}^2 + f_{r,Ty}^2 \leq \mu^2 \left(\frac{m_b}{2} \ddot{r}_z - k\Delta z \right)^2 \tag{4.31b}$$

The fulfillment of the friction-cone inequalities (4.31) requires a trade-off between the fastest motion law and the minimum virtual penetration, in order to grant the fewest internal prestress possible on the object that is able, at the same time, to avoid slipping of the object. This suggests that the problem can be represented by the independent variables s and Δz, which can be stored inside the vector $\boldsymbol{\sigma} = [s \quad \Delta z]^T$.

4.5.3 Problem Formulation

The jerk of the vector $\boldsymbol{\sigma}$ is chosen as the control input, in order to ensure smoothness of the trajectory and of the internal force $f_p = k\Delta z$:

$$\mathbf{u} = \dddot{\boldsymbol{\sigma}} = [\dddot{s} \quad \dddot{\Delta z}]^T \tag{4.32}$$

The system state is defined as a vector $\mathbf{x} \in \mathbb{R}^{18}$, namely:

$$\mathbf{x} = \left[\boldsymbol{\sigma}^T \ \dot{\boldsymbol{\sigma}}^T \ \ddot{\boldsymbol{\sigma}}^T \ \mathbf{q}_l^T \ \mathbf{q}_r^T \right]^T \tag{4.33}$$

where $\mathbf{q}_l \in \mathbb{R}^6$ and $\mathbf{q}_r \in \mathbb{R}^6$ are the arrays of the two cobot joint coordinates (Figure 4.8). The overall optimization problem can be formulated as [12], [38]

$$\min_{t_e, \mathbf{u}, \Delta z} \left[\int_0^{t_e} (1 + k_1 \Delta z + k_2 \mathbf{u}^T \mathbf{u}) \, dt \right] \tag{4.34a}$$

subject to

$$\dot{\mathbf{x}} = \boldsymbol{\Gamma}(\mathbf{x}, \mathbf{u}) \tag{4.34b}$$

$$\mathbf{x}(0) = [0 \ \Delta z(0) \ 0 \ 0 \ 0 \ 0 \ \mathbf{q}_l(0)^T \ \mathbf{q}_r(0)^T]^T \tag{4.34c}$$

$$\mathbf{x}(t_e) = [1 \ \Delta z(t_e) \ 0 \ 0 \ 0 \ 0 \ \mathbf{q}_l(t_e)^T \ \mathbf{q}_r(t_e)^T]^T \tag{4.34d}$$

$$|\dot{\mathbf{q}}_j| \leq 0.9 \, \dot{\mathbf{q}}_{j,max} \qquad j = l, r \tag{4.34e}$$

$$\underline{f}_p \leq k\Delta z \leq \overline{f}_p \tag{4.34f}$$

$$\boldsymbol{\Phi}(\{\mathbf{f}_l\}_l, \{\mathbf{f}_r\}_l) \leq \mathbf{0} \tag{4.34g}$$

$$|\mathbf{u}| \leq \mathbf{u}_{max} \tag{4.34h}$$

The cost functional (4.34a) is a trade-off between minimal time t_e of the motion, and minimal overall virtual penetration and jerk. The latter are weighted by the constants k_1 and k_2, whose values can be conveniently tuned. Function $\mathbf{\Gamma}$ (4.34b) takes into account the integration chain of the vector $\boldsymbol{\sigma}$ from the control $\mathbf{u}$ (4.35a), and the cobot inverse kinematics (4.35b, 4.35c), namely:

$$\frac{d}{dt}\begin{bmatrix}\boldsymbol{\sigma}\\ \dot{\boldsymbol{\sigma}}\\ \ddot{\boldsymbol{\sigma}}\end{bmatrix} = \begin{bmatrix}\dot{\boldsymbol{\sigma}}\\ \ddot{\boldsymbol{\sigma}}\\ \mathbf{u}\end{bmatrix} \tag{4.35a}$$

$$\frac{d}{dt}\mathbf{q}_l = \mathbf{J}_l^{-1}\boldsymbol{\xi}_{E_l}(s,\dot{s}) \tag{4.35b}$$

$$\frac{d}{dt}\mathbf{q}_r = \mathbf{J}_r^{-1}\boldsymbol{\xi}_{E_r}(s,\dot{s}) \tag{4.35c}$$

In (4.35b) and (4.35c), the terms $\mathbf{J}_l = \mathbf{J}_l(\mathbf{q}_l)$ and $\mathbf{J}_r = \mathbf{J}_r(\mathbf{q}_r)$ represent the Jacobian matrices of the cobots, whereas $\boldsymbol{\xi}_{E_l} = [\dot{\mathbf{r}}_{E_l}^T\ \boldsymbol{\omega}^T]^T$ and $\boldsymbol{\xi}_{E_r} = [\dot{\mathbf{r}}_{E_r}^T\ \boldsymbol{\omega}^T]^T$ are the end-effector twists, in which the cobots share the same angular velocity $\boldsymbol{\omega} = \mathbf{0}$ of the object.
Equality constraints (4.34c, 4.34d) are the initial and final conditions on the state vector $\mathbf{x}$. The initial and final values of the virtual penetration, namely the 2nd elements of $\mathbf{x}(0)$ and $\mathbf{x}(t_e)$, are left free, so that the solver is asked to find the values needed to satisfy the inequalities (4.31) in static conditions ($\dot{s} = \ddot{s} = 0$)

$$\Delta z(0) \geq \frac{m_b g}{2k\mu} \tag{4.36a}$$

$$\Delta z(t_e) \geq \frac{m_b g}{2k\mu} \tag{4.36b}$$

The values of $\mathbf{q}_j(0)$ and $\mathbf{q}_j(t_e)$[2], with $j = l, r$, are obtained by solving the inverse position analysis of the cobots in correspondence of the initial and final poses of the end-effectors [70]. Inequality constraints (4.34e) consider the limits on the maximum joint velocities $\dot{\mathbf{q}}_{j,max}$ of the cobots, with the addition of a coefficient equal to 0.9, to be on the side of safety. The inequality (4.34f) is employed to limit the internal prestress on the object, with $\underline{f}_p$ and $\overline{f}_p$ representing the minimum and the maximum value of f_p, respectively. The constraint (4.34g) represents the friction-cone conditions (4.31a,4.31b) that must be respected to avoid slipping on both sides of the cooperative grasping.

4.6 Experiments

4.6.1 Experimental Setup

The experimental setup (depicted in Figure 4.9) consists of two UR10 e-Series from Universal Robots[3], both equipped with an integrated force/torque sensor at the end-effector. The two cobots are mounted aboard a mobile platform from MiR[4], which

[2]The final conditions on the cobot joint coordinates are relaxed into inequalities by asking that the values of the final joint angles, computed by the optimizer, fall into the neighborhood of $\mathbf{q}_j(t_e)$ according to a certain tolerance, rather than restraining them to the precise value of $\mathbf{q}_j(t_e)$.

[3]www.universal-robots.com

[4]www.mobile-industrial-robots.com

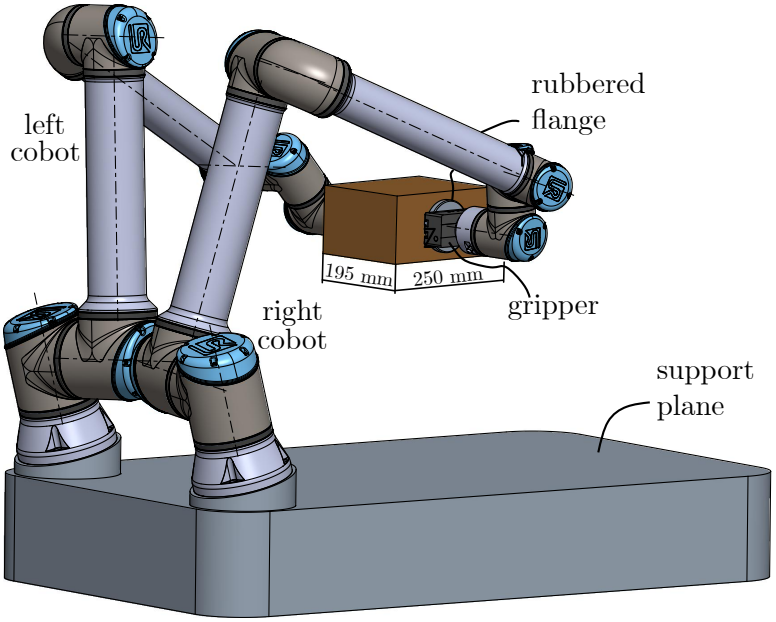
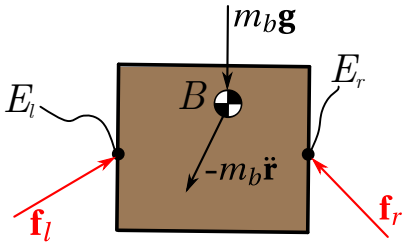
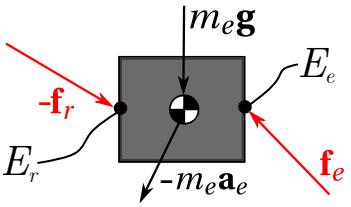


Figure 4.9: Dual-arm setup employed for the experiments.



(a) Scheme of the object translational equilibrium.



(b) Scheme of the gripper translational equilibrium.

Figure 4.10: Schemes employed for the equilibrium of the manipulated object and the right-cobot gripper.

remains still during the tests. Each cobot end-effector is interfaced with a 3D-printed flange, on which a rubbered tape is applied. The manipulated object is a cardboard box with dimensions $195 \times 130 \times 250$mm and a mass $m_b = 1$kg.
The friction coefficient is estimated by employing an inclined-plane test. The object that has to be manipulated is placed on a rubbered plane, whose inclination can be manually varied thanks to a revolute joint. By acting on the revolute joint in quasi-static conditions, as soon as slipping of the object is detected, the friction angle is read on the angular scale attached to the plane. The static friction between the cardboard and the rubbered flange is hence estimated as $\mu = \tan 32°$.

4.6.2 Adopted Model

While the left-cobot end-effector only presents the aforementioned rubbered flange, the right-cobot end-effector also mounts a gripper that could not be removed during the experiments; the gripper mass is $m_e = 2.21$kg (Figure 4.9). Since m_e is not negligible w.r.t. m_b, the force modeling presented in Section 4.4 has to be slightly revised. To maintain coherence with the symbols employed in Section 4.4, the forces exerted on the object are still indicated with $\mathbf{f}_l$ and $\mathbf{f}_r$.

While the experimental value of $\mathbf{f}_l$ can be easily read as the force measured by the integrated sensor of the left cobot, the value of $\mathbf{f}_r$ depends on the force measured by the right cobot, butthe two quantities are not equal. In fact, the force read by the right-cobot sensor is equal to $\mathbf{f}_e$, which is the reaction force of the right-cobot end-effector on the gripper. By writing the equilibrium of the gripper as (Figure 4.10b)

$$-\mathbf{f}_r + \mathbf{f}_e = -m_e\mathbf{g} + m_e\mathbf{a}_e \tag{4.37}$$

with $\mathbf{a}_e$ indicating the acceleration of the mass m_e, the force acting on the object from the right side is

$$\mathbf{f}_r = \mathbf{f}_e + m_e\mathbf{g} - m_e\mathbf{a}_e \tag{4.38}$$

The equilibrium of the manipulated object is still given by Equation (4.9a), namely (Figure 4.10a)

$$\mathbf{f}_l + \mathbf{f}_r = -m_b\mathbf{g} + m_b\ddot{\mathbf{r}} \tag{4.39}$$

Under the assumption of zero angular velocity of the object, the acceleration of the gripper and the object are the same, namely $\mathbf{a}_e = \ddot{\mathbf{r}}$ and, adding Equations (4.38) and (4.39) side by side, we obtain

$$\mathbf{f}_l + \mathbf{f}_e = -(m_b + m_e)\mathbf{g} + (m_b + m_e)\ddot{\mathbf{r}} \tag{4.40}$$

Equation (4.40) implies that the forces $\mathbf{f}_l$ and $\mathbf{f}_e$, as expected, have to generate the net force able to grant the translational motion of both the object and the gripper. The equilibriums expressed in Equations (4.37) and (4.39) can be written in matrix form as

$$\begin{bmatrix} \mathbf{0} & -\mathbf{I} & \mathbf{I} \\ \mathbf{I} & \mathbf{I} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{f}_l \\ \mathbf{f}_r \\ \mathbf{f}_e \end{bmatrix} = \begin{bmatrix} -m_e\mathbf{g} + m_e\ddot{\mathbf{r}} \\ -m_b\mathbf{g} + m_b\ddot{\mathbf{r}} \end{bmatrix} \tag{4.41}$$

Analogously to the case presented in Section 4.4, the problem is hyperstatic with 9 unknowns in 6 scalar equations. To solve it, a force distribution among the ∞^3 solutions can be chosen. In this case, it is assumed that, along the two tangential directions (x, y-axes of F_r) the cobot on the right withstands the whole weight and inertia of the gripper plus half of the net force needed to lift the object. As a consequence, the tangential forces exerted on the object on both sides of the grasping (i.e. at the interfaces E_l and E_r) equally distribute the weight and the inertia of the object. Regarding the normal direction, each of the two cobots (in correspondence of the interfaces E_l and E_e) is subjected to half of the inertia of both the object and the gripper, disregarding the internal prestress $f_p = k\Delta z$. To better clarify this force distribution, we can consider the projections of the force $\mathbf{f}_l$, $\mathbf{f}_r$ and $\mathbf{f}_e$ on the coordinate frame F_l:

$$\begin{bmatrix} f_{l,Tx} \\ f_{l,Ty} \\ f_{l,N} \end{bmatrix} = \begin{bmatrix} \frac{m_b}{2}\ddot{r}_x \\ \frac{m_b}{2}(\ddot{r}_y + g) \\ \frac{m_b + m_e}{2}\ddot{r}_z + k\Delta z \end{bmatrix} \tag{4.42a}$$

$$\begin{bmatrix} f_{r,Tx} \\ f_{r,Ty} \\ f_{r,N} \end{bmatrix} = \begin{bmatrix} \frac{m_b}{2}\ddot{r}_x \\ \frac{m_b}{2}(\ddot{r}_y + g) \\ \frac{m_b - m_e}{2}\ddot{r}_z - k\Delta z \end{bmatrix} \tag{4.42b}$$

$$\begin{bmatrix} f_{e,Tx} \\ f_{e,Ty} \\ f_{e,N} \end{bmatrix} = \begin{bmatrix} \left(m_e + \frac{m_b}{2}\right)\ddot{r}_x \\ \left(m_e + \frac{m_b}{2}\right)(\ddot{r}_y + g) \\ \frac{m_b + m_e}{2}\ddot{r}_z - k\Delta z \end{bmatrix} \tag{4.42c}$$

Since the gripper is attached to the right-cobot end-effector, a calibration procedure (made available by the robot manufacturer) made the weight of the gripper transparent to the force/torque sensor measurement. This means that the read force is not $\mathbf{f}_e$, but $\mathbf{f}'_e = \mathbf{f}_e + m_e\mathbf{g}$, whose projection on F_l gives

$$\begin{bmatrix} f'_{e,Tx} \\ f'_{e,Ty} \\ f'_{e,N} \end{bmatrix} = \begin{bmatrix} \left(m_e + \frac{m_b}{2}\right)\ddot{r}_x \\ m_e\ddot{r}_y + \frac{m_b}{2}(\ddot{r}_y + g) \\ \frac{m_b + m_e}{2}\ddot{r}_z - k\Delta z \end{bmatrix} \tag{4.43}$$

Hence, the sum of the reaction forces measured by both sensors theoretically equals the inertia of the object and the gripper plus the weight of the sole object, i.e.:

$$\mathbf{f}_l + \mathbf{f}'_e = -m_b\mathbf{g} + (m_b + m_e)\ddot{\mathbf{r}} \tag{4.44}$$

4.6.3 Optimization Problem Resolution

As far as the resolution of the time-optimal trajectory planning problem is concerned (v. Section 4.5), the values of some parameters must be chosen or identified.
The penetration stiffness k is identified by employing the controller of Section 4.3 in zero-motion conditions. In practice, the two cobots are commanded to hold the position of their end-effectors on the object, while their targets are given by $\{\mathbf{r}_{E_{j,d}}\}_j =$

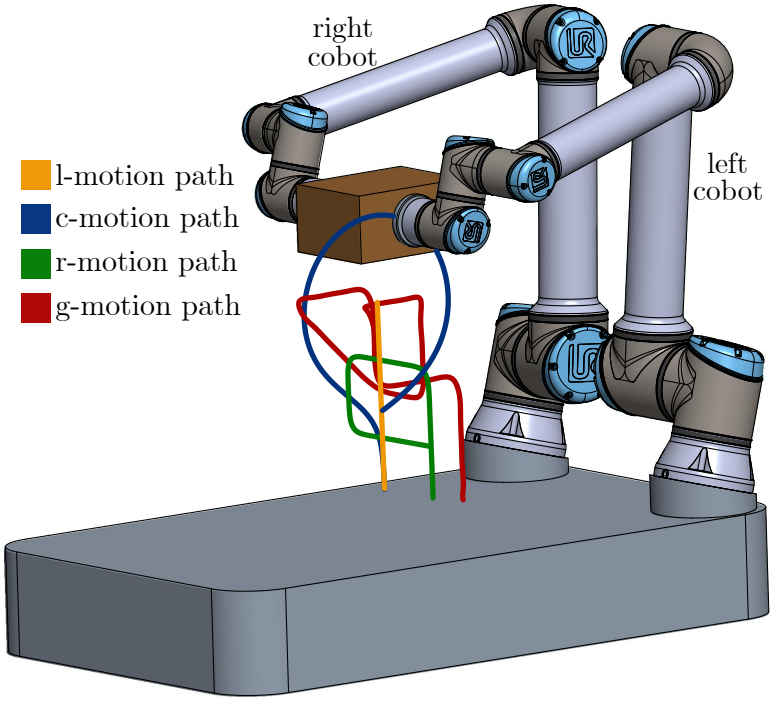


Figure 4.11: The four paths prescribed for the dual-arm tests.

Table 4.1: Maximum velocity for the i-th joint, considering the left and the right cobots.

i	1	2	3	4	5	6
$\dot{q}_{l,i-max}$ [rad/s]	1.05	1.05	1.05	1.05	1.05	1.05
$\dot{q}_{r,i-max}$ [rad/s]	1.05	1.05	1.05	1.05	1.05	1.05

$\{\mathbf{r}_{E_j}\}_j + [0\ 0\ \Delta z]^T$ $(j = l, r)$, and a linear trend of the virtual penetration Δz from 0mm to 100mm is chosen[5]. By reading the normal forces exerted by the cobots on the object, the value of k is estimated as $k \approx 1000$N/m.
The two cobots, as already mentioned, are two UR10e, hence having the same nominal kinematic parameters and limits. In particular, the values of the joint velocity limits employed in the inequality (4.34e) are the same for both cobots, and they are reported in Table 4.1.
Regarding the inequality (4.34f), whose aim is to limit the internal prestress on the object, its maximum and minimum values are chosen equal to $\overline{f}_p = 30$N and $\underline{f}_p = 5$N, respectively.
Given the force-model adjustment described in Section 4.6.2, due to the presence of the gripper m_e attached to the right-cobot end-effector, the constraint (4.34g) aimed at satisfying the friction-cone conditions has to be adapted to the Equations (4.42a), (4.42b). By inserting the assumed force distributions of Equations (4.42a), (4.42b) inside the inequality (4.19), the function $\mathbf{\Phi}$ (v. (4.34g)) can be written as

$$\mathbf{\Phi} = \begin{bmatrix} f_{l,Tx}^2 + f_{l,Ty}^2 - \mu^2 \left(\frac{m_b + m_e}{2} \ddot{r}_z + k\Delta z \right)^2 \\ f_{r,Tx}^2 + f_{r,Ty}^2 - \mu^2 \left(\frac{m_b - m_e}{2} \ddot{r}_z - k\Delta z \right)^2 \end{bmatrix} \tag{4.45}$$

The initial and final conditions of the cobot joint angles are computed by solving the inverse position analysis, where the translational vector of the homogeneous transformation matrices $\mathbf{T}_{E_j}(0)$ and $\mathbf{T}_{E_j}(t_e)$ depends on the desired position of the object COM and on the position vector between the object COM B and the interface E_j $(j = l, r)$. The assumption is that the interaction points E_j $(j = l, r)$ are the centroids of the object contact surface and that the object mass is equally distributed, hence making the COM B coincide with the object geometric center. As a consequence, the expression of the position vector $\mathbf{r}_{BE_j}$ projected on F_j $(j = l, r)$ simplifies in

$$\{\mathbf{r}_{BE_j}\}_j = \begin{bmatrix} 0 & 0 & -\frac{l}{2} \end{bmatrix}^T, \quad j = l, r \tag{4.46}$$

Figure 4.11 shows the four different paths considered by the trajectory planning:

- an up-and-down linear path (*l-motion*);
- a circular path (*c-motion*);
- a rectangular path (*r-motion*);
- a general path (*g-motion*).

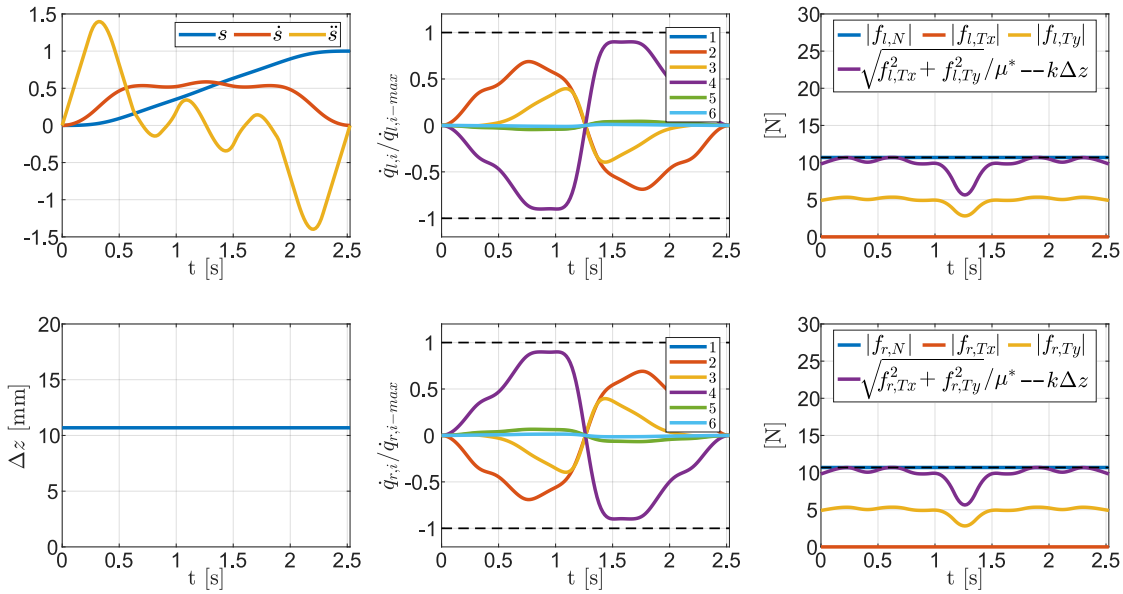
[5]Note that these values correspond to a virtual penetration and are not related to the actual deformation of the object. For the latter aspect, the measurement of the object deformation was not performed, given its irrelevance for the admittance-control implementation.

For the definition of the four paths, the B-spline degree is set to $d = 4$ (v. Equation (4.24)). The expression of the object path $\mathbf{r}(s)$ as a function of the path parameter s is then used in Equation (4.27) to write the reference paths that the two end-effectors have to follow during motion. The actual target of each cobot is hence written as a combination of the reference path $\mathbf{r}_{E_j}(s)$ and the virtual penetration Δz as explained in Equation (4.21). The trend of the virtual penetration Δz and hence of the internal prestress $f_p = k\Delta z$ is optimized within the resolution of the constrained time-optimal trajectory planning. Regarding the latter aspect, for each path, two optimizations, that consider different trends of Δz and thus of f_p, are carried out:

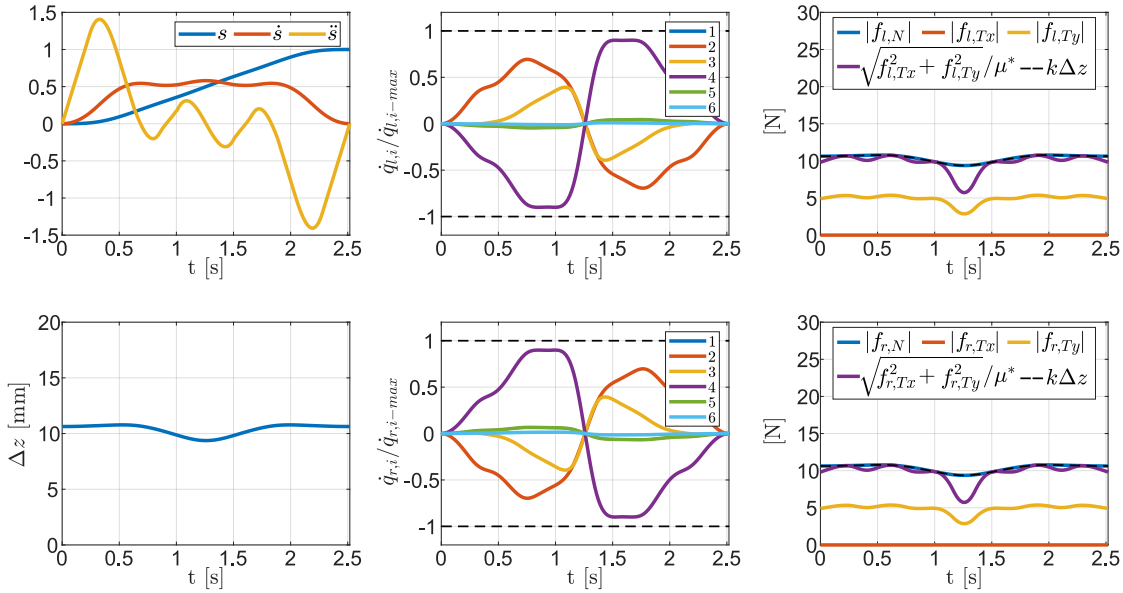
- the first one imposes a constant optimal value of Δz (which is found by the solver) throughout the whole motion;
- the second one considers a variable optimal Δz (the optimal trend is computed by the solver) during the trajectory execution.

For the friction-cone constraint, a safety factor of 0.8 is employed: this results in a static friction μ^* used within the optimization equal to $\mu^* = 0.8 \tan 32° = 0.5$.
The constrained time-optimal trajectory planning is solved by adopting a multiple-shooting method and using CasADi [3], a software framework implemented in Matlab for nonlinear optimization and optimal control. The employed algorithm is the interior-point method.
Figures 4.12, 4.13, 4.14, 4.15 show the results obtained by solving the constrained optimization for the l-motion, the c-motion, the r-motion and the g-motion, respectively. For each motion, the subfigures 4.12a, 4.13a, 4.14a, 4.15a illustrate the resolution considering a constant value of the virtual penetration Δz and hence of the internal prestress f_p, whereas in subfigures 4.12b, 4.13b, 4.14b, 4.15b the resolution with a varying virtual penetration Δz and hence a varying internal prestress f_p is depicted. The first column of each subfigure shows the optimized trend of the motion law $s, \dot{s}, \ddot{s}$ and of the virtual penetration Δz. The second column represents the ratio between the cobot joint velocities and their maximum values, namely $\dot{q}_{j,i}/\dot{q}_{j,i-max}$, with $j = l, r$ and $i = 1, \ldots, 6$. The last column reports the absolute values of the normal force $|f_{j,N}|$ and the tangential forces $|f_{j,Tx}|, |f_{j,Ty}|$, acting on the object on both sides of the grasping ($j = l, r$); furthermore, in the same plots, the purple line indicates the quantity $f_{j,T}/\mu^*$, with $f_{j,T} = \sqrt{f_{j,Tx}^2 + f_{j,Ty}^2}$, whereas the black dashed line stands for the internal prestress $f_p = k\Delta z$.
As far as the required motion lies on a plane that is orthogonal to the z-axis of frame F_l (i.e. $\ddot{r}_z = 0$ m/s^2), the normal force $|f_{j,N}|$ is only characterized by the internal prestress $f_p = k\Delta z$. This is the case of the l-motion and the c-motion, where, looking at the third column of Figures 4.12 and 4.13, the blue line always coincides with the black dashed one. On the contrary, when the object motion has a component along the z-axis of frame F_l (i.e. $\ddot{r}_z \neq 0$ m/s^2), the normal force is composed of the internal prestress f_p and of a combination of the inertia force of the object and the gripper along the z-axis, as expressed in Equations (4.42a) and (4.42b). This is the reason why, for the r-motion and the g-motion, in the third column of Figures 4.14 and 4.15, the blue line does not replicate the black dashed one.

In all the friction-cone plots (see the third column of Figures 4.12, 4.13, 4.14, 4.15), the purple line is always under the blue one, indicating that the no-slipping constraint

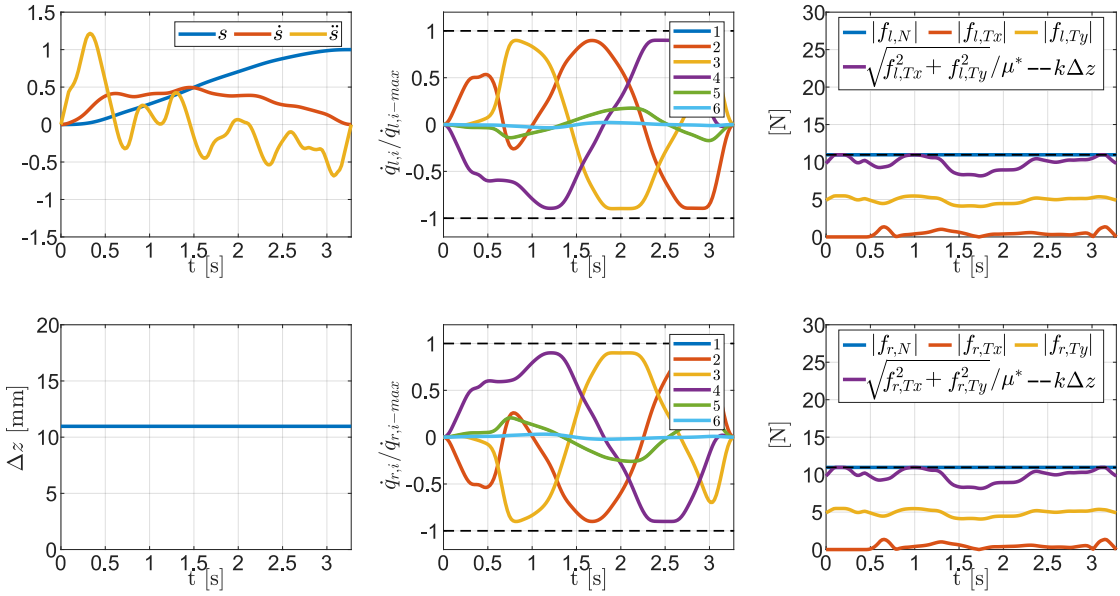


(a) Constant virtual penetration Δz.

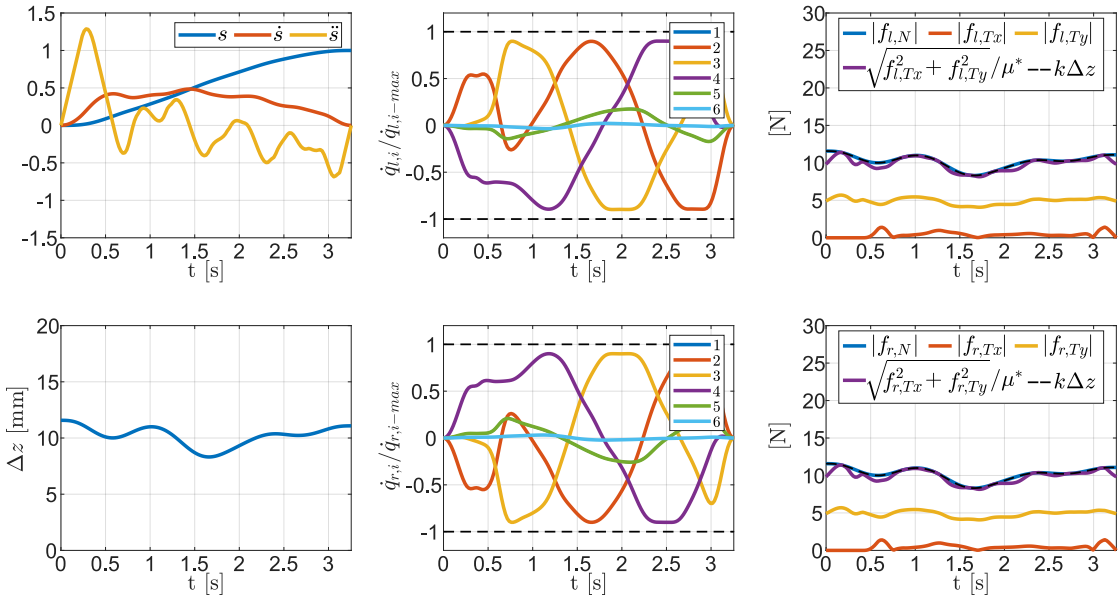


(b) Varying virtual penetration Δz.

Figure 4.12: Optimization results for the l-motion.

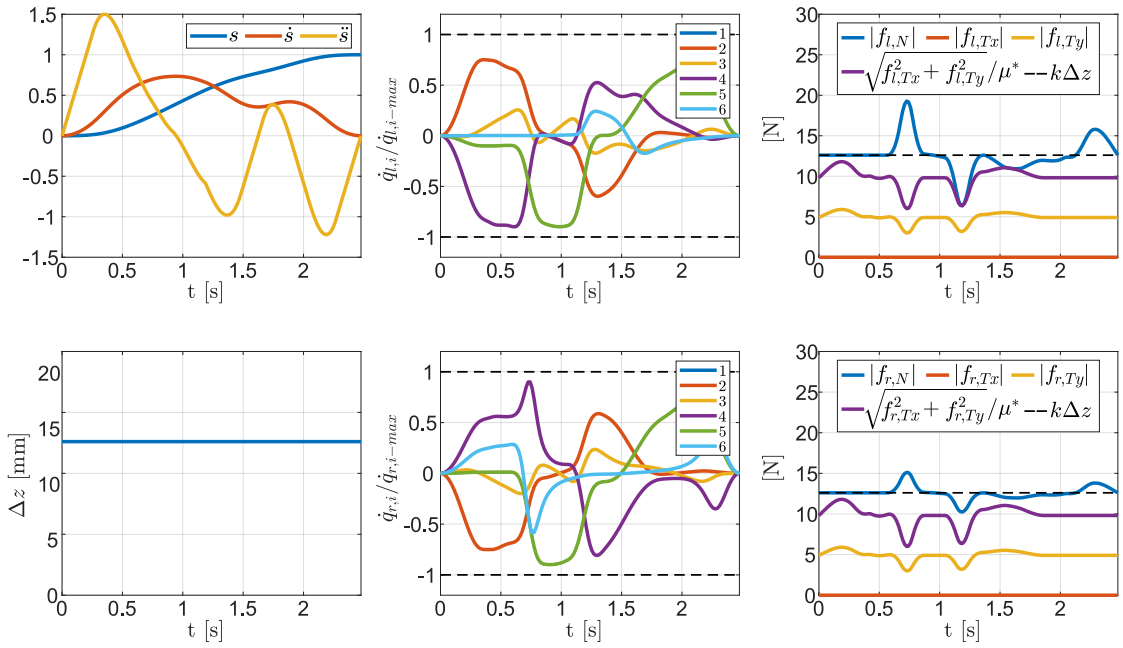


(a) Constant virtual penetration Δz.

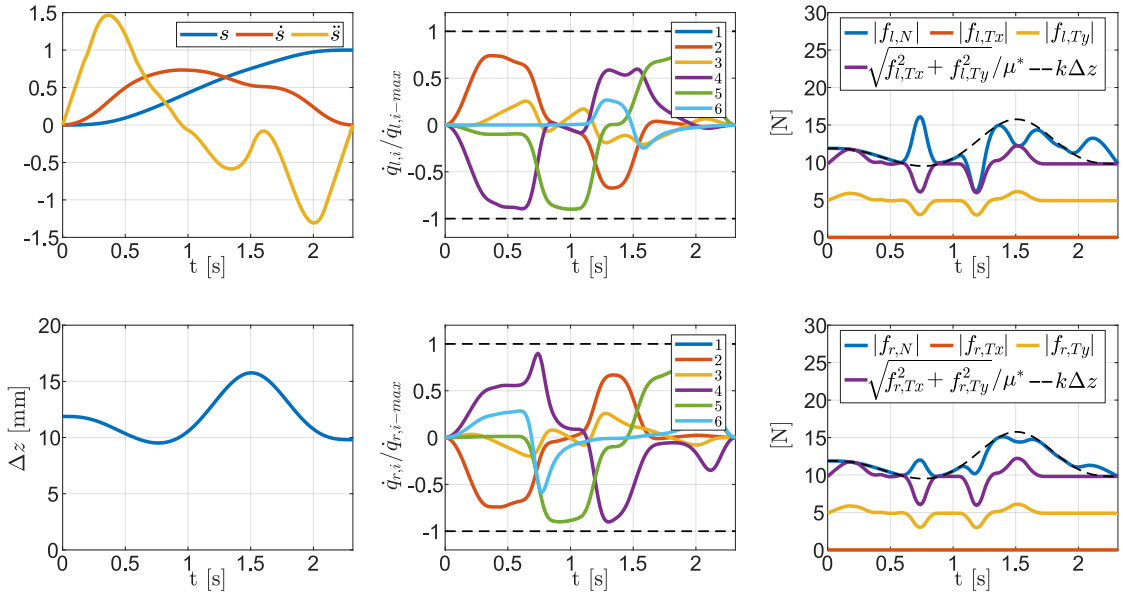


(b) Varying virtual penetration Δz.

Figure 4.13: Optimization results for the c-motion.

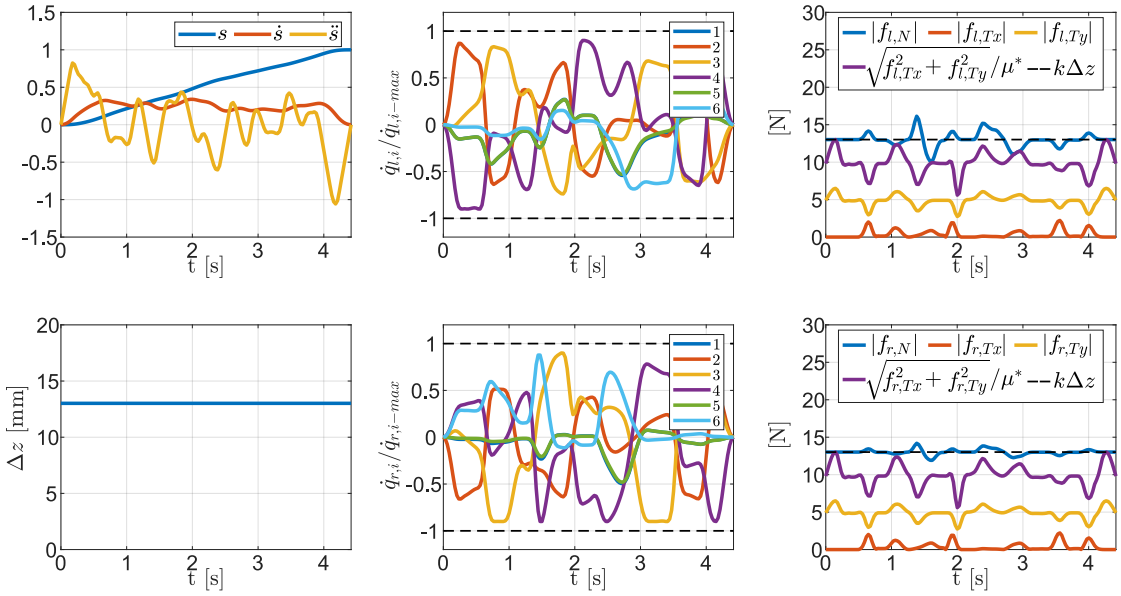


(a) Constant virtual penetration Δz.

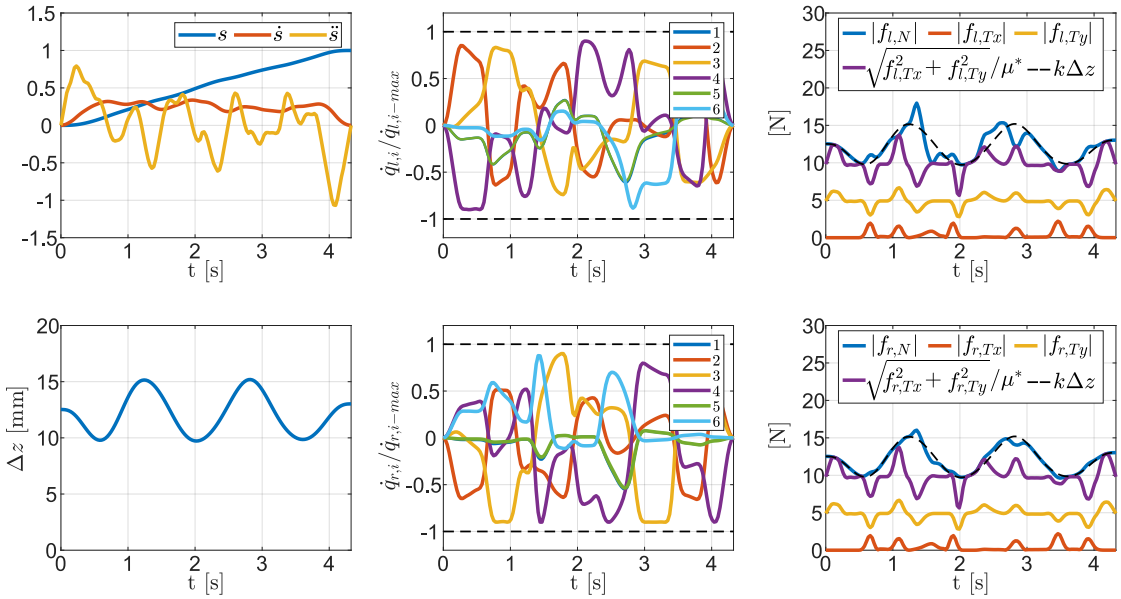


(b) Varying virtual penetration Δz.

Figure 4.14: Optimization results for the r-motion.



(a) Constant virtual penetration Δz.



(b) Varying virtual penetration Δz.

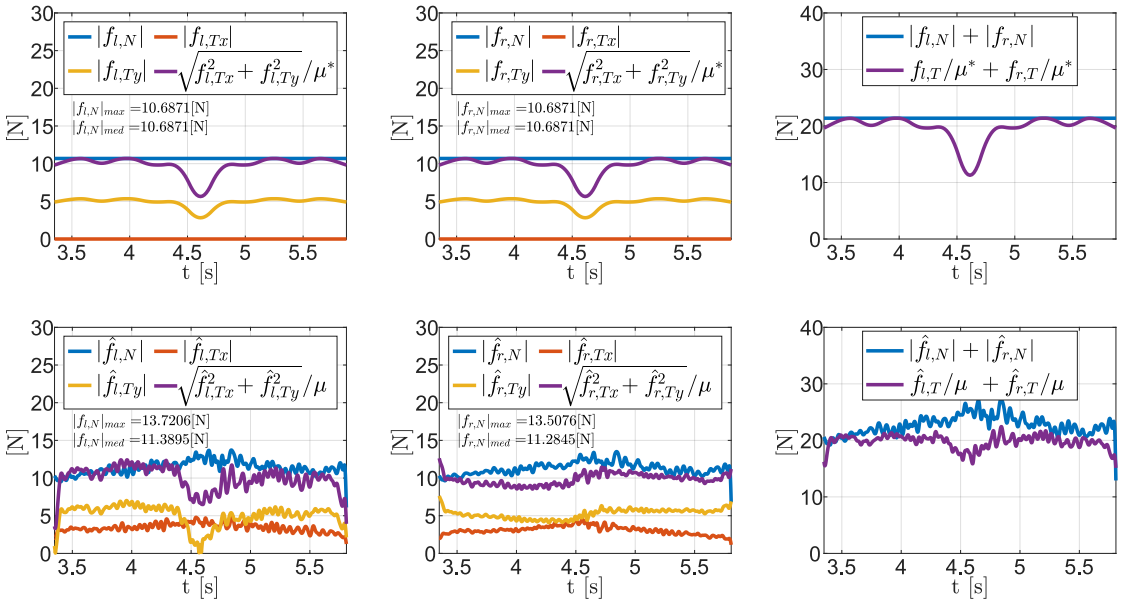
Figure 4.15: Optimization results for the g-motion.

Table 4.2: Initial, maximum and mean values of the internal prestress f_p obtained as a result of the time-optimal trajectory planning, with an indication of the final time t_e of the corresponding trajectory.

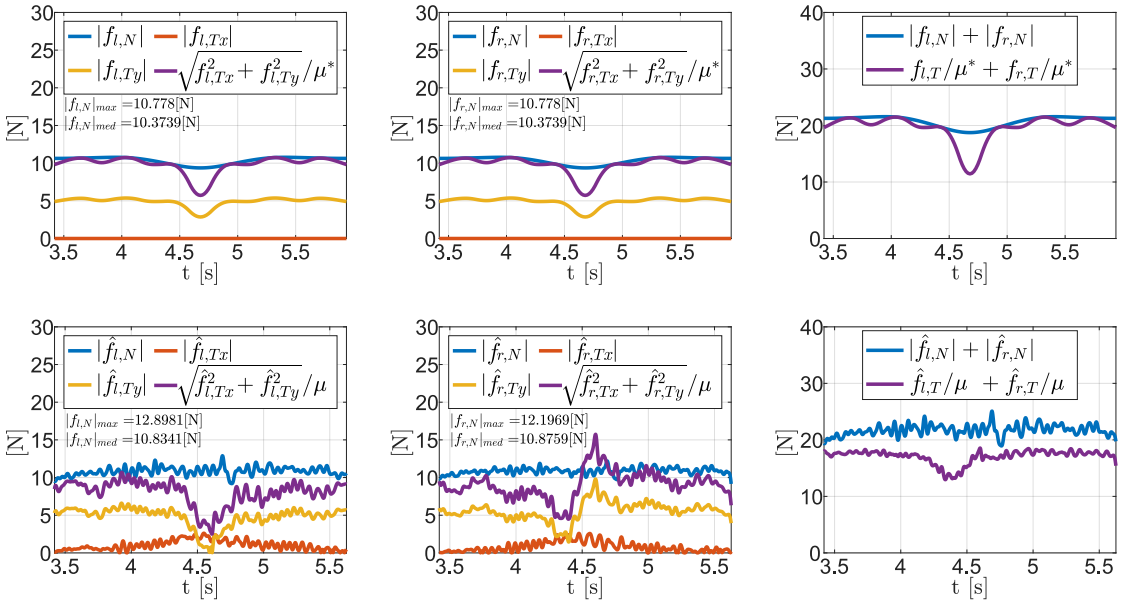
	Constant Δz	Varying Δz
l-motion	$t_e = 2.52$s	$t_e = 2.52$s
	$f_p(0) = 10.7$N	$f_p(0) = 10.6$N
	$f_{p,max} = 10.7$N	$f_{p,max} = 10.8$N
	$f_{p,med} = 10.7$N	$f_{p,med} = 10.4$N
c-motion	$t_e = 3.27$s	$t_e = 3.25$s
	$f_p(0) = 11.0$N	$f_p(0) = 11.6$N
	$f_{p,max} = 11.0$N	$f_{p,max} = 11.6$N
	$f_{p,med} = 11.0$N	$f_{p,med} = 10.2$N
r-motion	$t_e = 2.47$s	$t_e = 2.31$s
	$f_p(0) = 12.6$N	$f_p(0) = 11.9$N
	$f_{p,max} = 12.6$N	$f_{p,max} = 15.8$N
	$f_{p,med} = 12.6$N	$f_{p,med} = 11.8$N
g-motion	$t_e = 4.41$s	$t_e = 4.33$s
	$f_p(0) = 13.0$N	$f_p(0) = 12.5$N
	$f_{p,max} = 13.0$N	$f_{p,max} = 15.2$N
	$f_{p,med} = 13.0$N	$f_{p,med} = 12.1$N

(inequality (4.34g) with the function $\mathbf{\Phi}$ expressed in Equation (4.45)) is satisfied in the model scenario. Furthermore, in the second column of Figures 4.12, 4.13, 4.14, 4.15, the ratios between the joint velocities and their maximum values always have a certain distance from the lower and upper dashed lines, hence reflecting the safety factor of 0.9 employed in the inequality (4.34e). Table 4.2 summarizes the main results of the optimizations, with an indication of the final time t_e of the corresponding trajectory, together with the initial, maximum and mean values of the obtained internal prestress f_p for each motion. Looking at Table 4.2, it can be stated that, for the l-motion and the c-motion, no apparent advantage in terms of trajectory duration t_e is gained by choosing the optimization with a varying Δz instead of the one with a constant Δz, and even for the other two motions (r-motion and g-motion), the time saving is almost imperceptible (6% and 2%, respectively). Furthermore, for a given motion type, the maximum value $f_{p,max}$ reached by the internal prestress in the model scenario is higher in the trajectory with a varying Δz w.r.t. the one with a constant penetration. Conversely, the trajectory with a varying virtual penetration grants a smaller mean value $f_{p,max}$ of the internal prestress throughout the motion. In addition, the initial value of f_p is chosen, in every optimization, by the solver in order to satisfy the friction-constraint (4.19) in static conditions (i.e. $\ddot{\mathbf{r}} = \mathbf{0}$), namely:

$$f_p(0) = k\Delta z(0) \geq \frac{m_b g}{\mu^*} \tag{4.47}$$

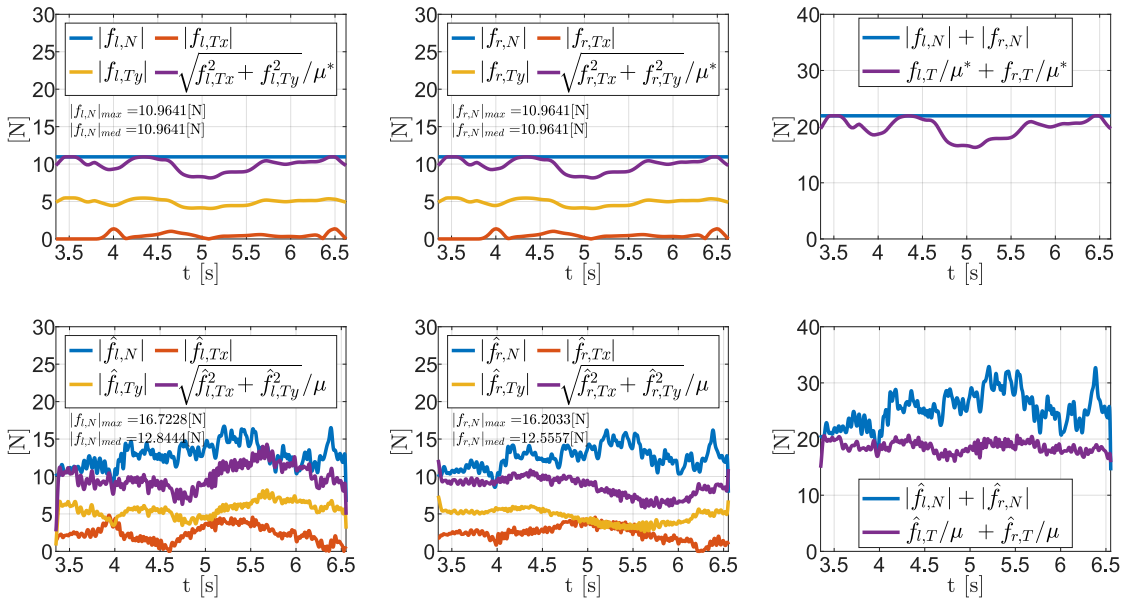


(a) Constant virtual penetration Δz.

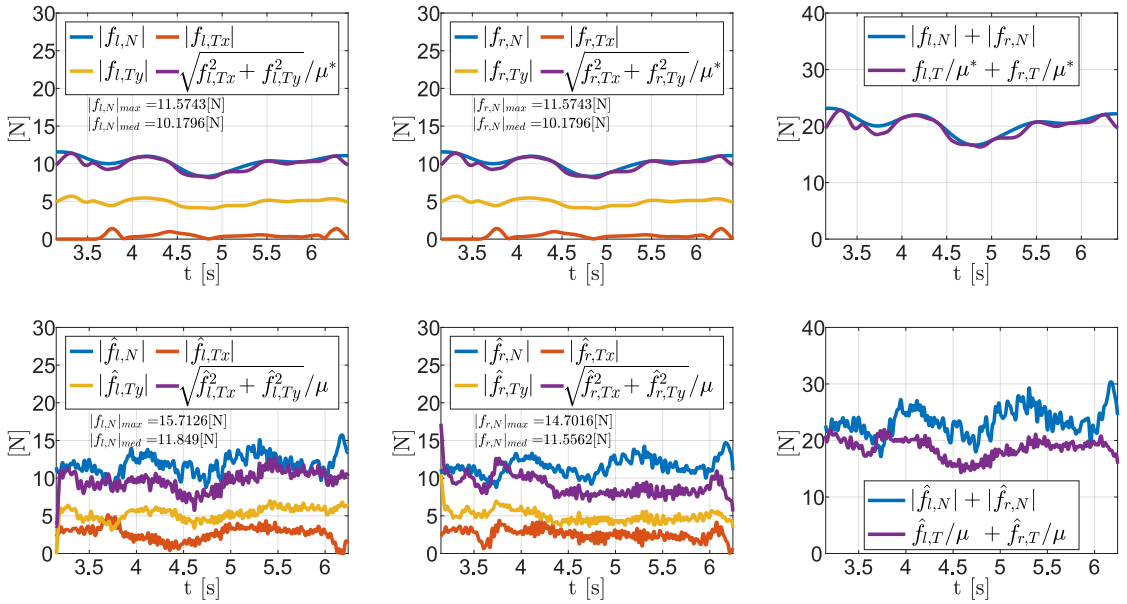


(b) Varying virtual penetration Δz.

Figure 4.16: Assessment of the friction-cone condition for the l-motion.

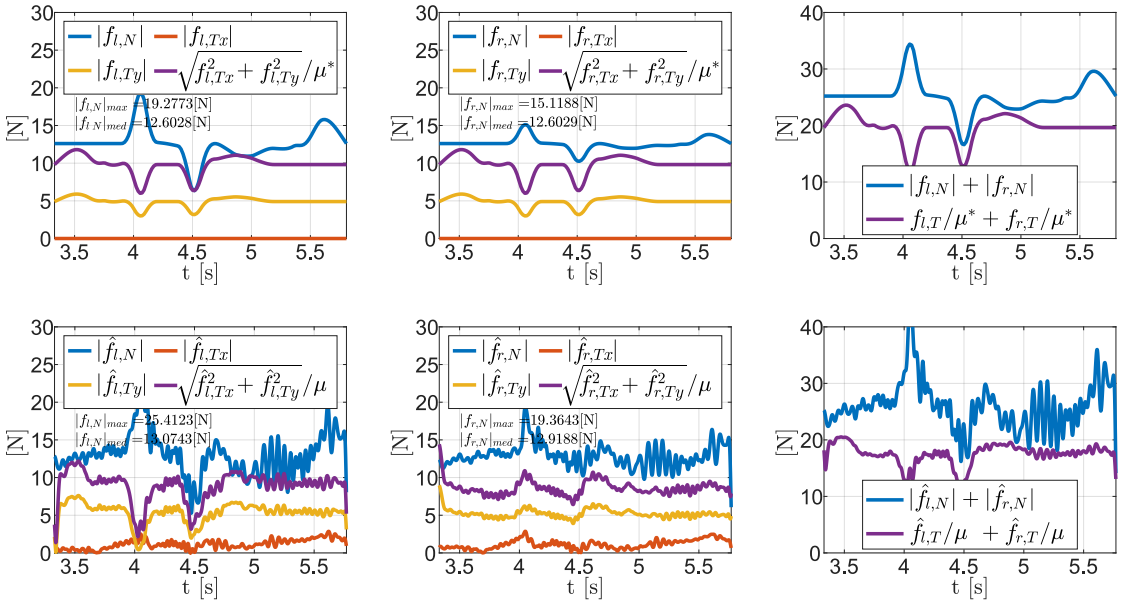


(a) Constant virtual penetration Δz.

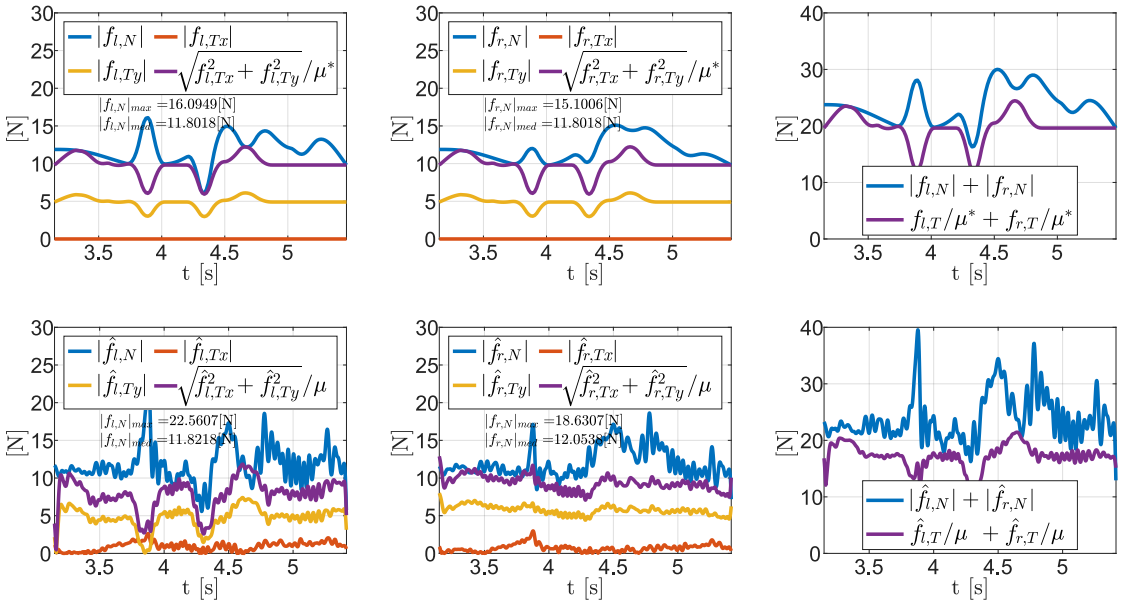


(b) Varying virtual penetration Δz.

Figure 4.17: Assessment of the friction-cone condition for the c-motion.

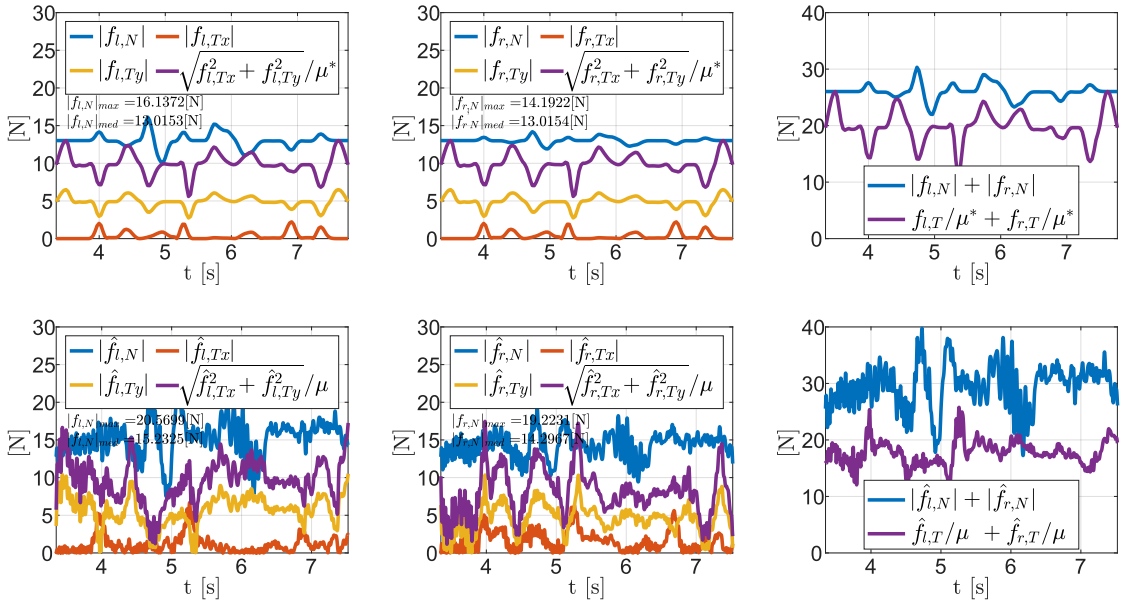


(a) Constant virtual penetration Δz.

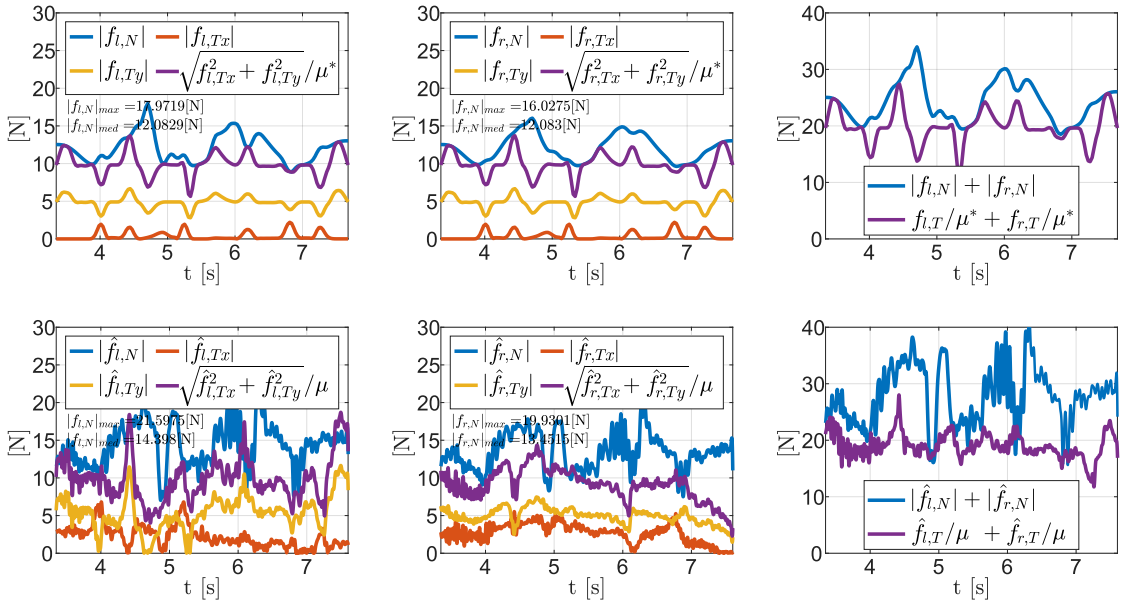


(b) Varying virtual penetration Δz.

Figure 4.18: Assessment of the friction-cone condition for the r-motion.



(a) Constant virtual penetration Δz.



(b) Varying virtual penetration Δz.

Figure 4.19: Assessment of the friction-cone condition for the g-motion.

4.6.4 Experimental Results

During the experiments, performed within the R&D department of IMA Spa[6], the two cobots are controlled by using ROS[7], with an admittance-controller node implemented in C++. The preliminary operations aimed at placing the object in correspondence of its desired initial pose are the following:

- the left and right cobots are sent to the corresponding initial configurations $\mathbf{q}_l(0)$ and $\mathbf{q}_r(0)$;
- the left cobot, through a linear motion along the opposite direction of its end-effector z-axis, is retracted to leave the space for manual placement of the object;
- with the same opposite linear motion, the left cobot moves towards the object until a contact force of 1N along the z-axis is detected;
- the two sensor measurements are tared to zero.

The value of 1N for the detection of the contact is chosen to avoid an higher internal prestress on the object, which may compromise the reliability of the experiment.
This way, the two cobots, in correspondence of their initial reference positions $\mathbf{r}_{E_j}(0)$ ($j = l, r$), are led to exert a negligible internal prestress on the object (v. Figure 4.7). A transitory procedure of roughly 3s is then started to bring the prestress to the initial value of $f_p(0)$. This procedure is needed since, while during the preliminary operations, the object is lying on a support plane and its gravity force is counterbalanced by the plane itself, hence not requiring an internal prestress to satisfy the friction-cone conditions, as soon as the desired motion starts and the object has to be lifted by the cobots, the internal prestress must equal the value $f_p(0)$ to avoid slipping.
This also justifies why, in Figure from 4.16 to 4.27, where the results from the experiments are illustrated, the time reported on the x-axis starts from a value greater than 3s. In particular, Figures 4.16, 4.17, 4.18, 4.19 show the comparison between the model and the experimental forces exerted on the object during the l-motion, c-motion, r-motion and g-motion, respectively. For each of the figures, the cases with a constant and varying Δz are provided. To better explain the content of the aforementioned figures, the first column compares the tangential and normal forces exerted by the left cobot on the object in the model scenario with the ones read by the sensor of the left cobot; the second column shows the comparison between the forces exerted on the right interface of the object in the model scenario with the ones obtained from the measurements of the right-cobot sensor. In addition, a purple line indicates the quantity $|f_{j,T}|/\mu^*$ for the model case and $|\hat{f}_{j,T}|/\mu$ for the experimental measurement to allow a visual check on the friction-cone condition, where the symbol $\hat{}$ is employed for measured quantities.
To sum up, indicating with $\hat{\mathbf{f}}_l$ and $\hat{\mathbf{f}}'_e$ the forces measured by the left and right sensors, respectively, the comparison between the modeled $\mathbf{f}_l$ and the measured $\hat{\mathbf{f}}_l$ is represented in the first column, whereas the modeled $\mathbf{f}_r$ is compared with the quantity $\hat{\mathbf{f}}'_e - m_e\ddot{\mathbf{r}}$ (v. Section 4.6.2) in the second column. Finally, the third column represents the sum of the sides of the friction-cone inequalities (see (4.19)), for both the model and the experimental scenarios. The comparison of the blue line (corresponding to the sum $|f_{l,N}| + |f_{r,N}|$) with the purple line (corresponding to the sum $f_{l,T}/\mu^* + f_{r,T}/\mu^*$

[6] https://ima.it
[7] https://www.ros.org

for the model and $f_{l,T}/\mu + f_{r,T}/\mu$ for the experiments) provides a necessary but not sufficient condition for the fulfillment of the no-slipping conditions. This means that, slipping is detected when the purple line exceeds the blue one, whereas, when the blue line is over the purple one, slipping is prevented as long as the friction-cone inequalities are individually respected.
The experimental plots of Figures 4.16, 4.17, 4.18, 4.19 show that the blue line is always over the purple one, except for some isolated time instants, hence confirming that the friction inequalities are respected during the execution of the trajectories. This also finds evidence by looking the recorded videos[8], where no apparent slipping of the object occurs. Table 4.3 reports the accuracy indexes $\varepsilon_{j,max}$ and $\varepsilon_{j,med}$ that take into account the maxima and the mean values of the normal forces acting on the object and are defined as

$$\varepsilon_{j,max} = \frac{|f_{j,N}|_{max-exp} - |f_{j,N}|_{max-mod}}{|f_{j,N}|_{max-mod}} \times 100\%, \quad j = l, r \tag{4.48}$$

$$\varepsilon_{j,med} = \frac{|f_{j,N}|_{med-exp} - |f_{j,N}|_{med-mod}}{|f_{j,N}|_{med-mod}} \times 100\%, \quad j = l, r \tag{4.49}$$

with $|f_{j,N}|_{med-p}$ ($p = exp, mod$) being the mean value of $f_{j,N}$ ($j = l, r$) during the motion time period, namely

$$|f_{j,N}|_{med-p} = \frac{\sum_{i=1}^{N_{tot}} |f_{j,N,p}(t_i)|}{N_{tot}}, \quad j = l, r, \quad p = exp, mod \tag{4.50}$$

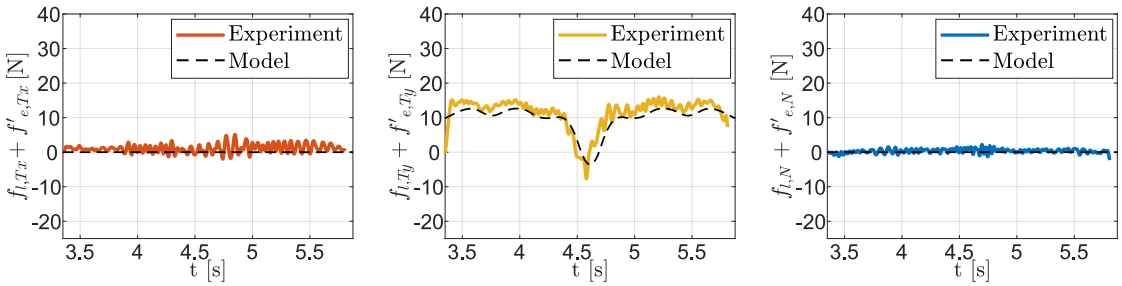
where mod and exp denote the model and the experimental quantities, respectively, and N_{tot} is the total number of samples. It can be noted that, in the majority of the cases, for a given motion type, the trajectory with a varying Δz grants lower values of the accuracy indexes $\varepsilon_{j,max}$ and $\varepsilon_{j,med}$ ($j = l, r$), w.r.t. the case characterized by a constant Δz. Furthermore, the value of $\varepsilon_{j,med}$ is always below the 19.1%, proving that the general trend of the normal forces exerted on the object remains in line with the commanded trend, even when the percentage error of the maxima $\varepsilon_{j,max}$ is high. For instance, the worst case is represented by the c-motion with a constant Δz, where, even, if the values of $\varepsilon_{l,max}$ and $\varepsilon_{r,max}$ equal 52.5% and 47.8%, the general trends are still acceptable ($\varepsilon_{l,med}$ = 17.1% and $\varepsilon_{r,med}$ = 14.5%). Regarding the correspondence of the force components between the model and the experiment, the plots on the first column exhibit a good adherence of the predicted forces with the measured ones, even in correspondence of the boundary friction conditions when the purple and the blue lines touch. The same cannot be stated for the plots of the second column, where the tangential forces present some disparities between model and experimental values. This can be attributable to the fact that, while the first column compares model quantities with directly measured ones, on the second column the model forces exerted on the object on the right side of the contact are compared with experimental quantities that are not directly measured, but they are obtained as a combination of a measured force (i.e. $\hat{\mathbf{f}}'_e$) and a model term (i.e. $m_e\ddot{\mathbf{r}}$).

Figures 4.20, 4.21, 4.22, 4.23 illustrate, for each motion, with both a constant and a varying Δz, the sum of the reaction forces $\mathbf{f}_l$ and $\mathbf{f}'_e$, projected on the coordinate frame

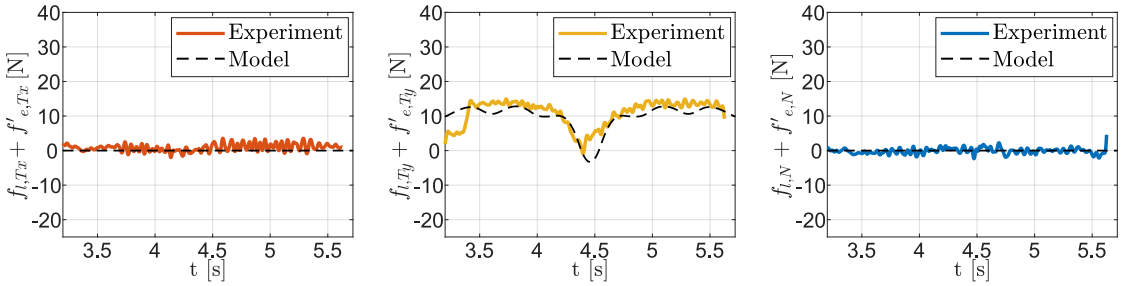
[8]The performed motions can be seen at the link dual-arm videos.

Table 4.3: Normal force accuracy indexes.

	Constant Δz	Varying Δz
l-motion	$\varepsilon_{l,max} = 28.4\%$	$\varepsilon_{l,max} = 19.7\%$
	$\varepsilon_{l,med} = 6.6\%$	$\varepsilon_{l,med} = 4.4\%$
	$\varepsilon_{r,max} = 26.4\%$	$\varepsilon_{r,max} = 13.2\%$
	$\varepsilon_{r,med} = 5.6\%$	$\varepsilon_{r,med} = 4.8\%$
c-motion	$\varepsilon_{l,max} = 52.5\%$	$\varepsilon_{l,max} = 35.7\%$
	$\varepsilon_{l,med} = 17.1\%$	$\varepsilon_{l,med} = 16.4\%$
	$\varepsilon_{r,max} = 47.8\%$	$\varepsilon_{r,max} = 27.0\%$
	$\varepsilon_{r,med} = 14.5\%$	$\varepsilon_{r,med} = 13.5\%$
r-motion	$\varepsilon_{l,max} = 31.8\%$	$\varepsilon_{l,max} = 40.2\%$
	$\varepsilon_{l,med} = 3.7\%$	$\varepsilon_{l,med} = 0.2\%$
	$\varepsilon_{r,max} = 28.1\%$	$\varepsilon_{r,max} = 23.4\%$
	$\varepsilon_{r,med} = 2.5\%$	$\varepsilon_{r,med} = 2.1\%$
g-motion	$\varepsilon_{l,max} = 27.5\%$	$\varepsilon_{l,max} = 20.2\%$
	$\varepsilon_{l,med} = 17.0\%$	$\varepsilon_{l,med} = 19.1\%$
	$\varepsilon_{r,max} = 35.4\%$	$\varepsilon_{r,max} = 24.3\%$
	$\varepsilon_{r,med} = 9.8\%$	$\varepsilon_{r,med} = 11.3\%$

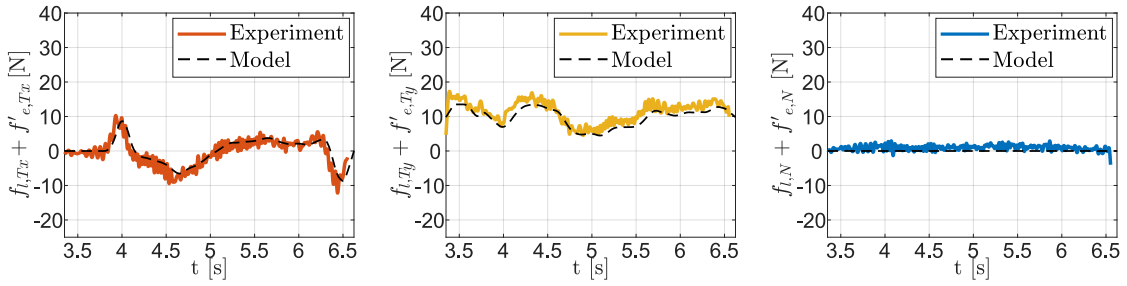


(a) Constant virtual penetration Δz.

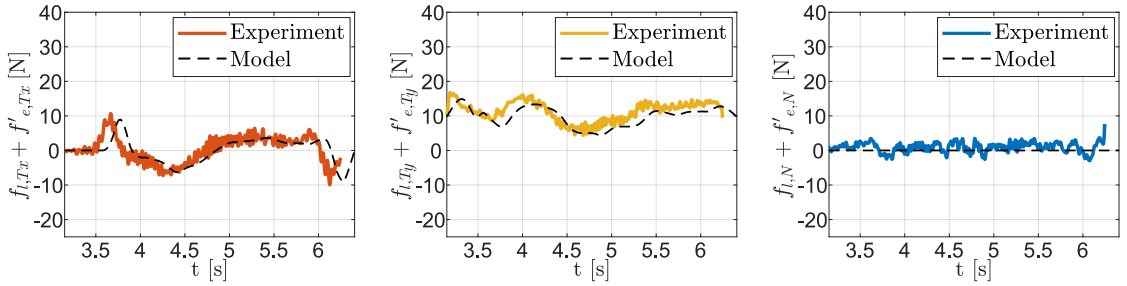


(b) Varying virtual penetration Δz.

Figure 4.20: Assessment on the application of the net wrench on both the object and the gripper for the l-motion.

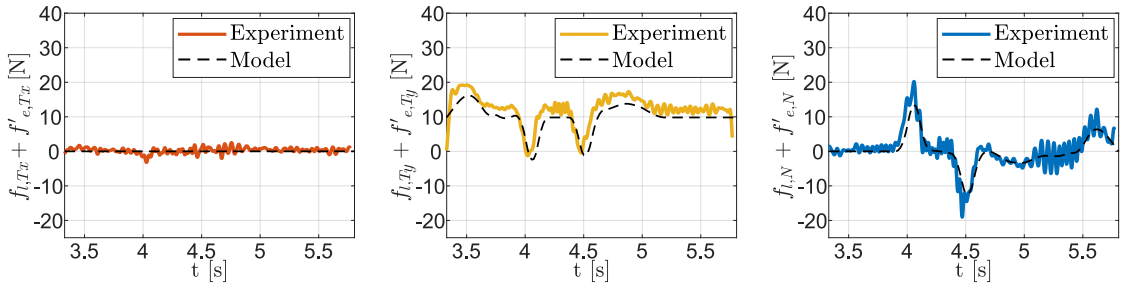


(a) Constant virtual penetration Δz.

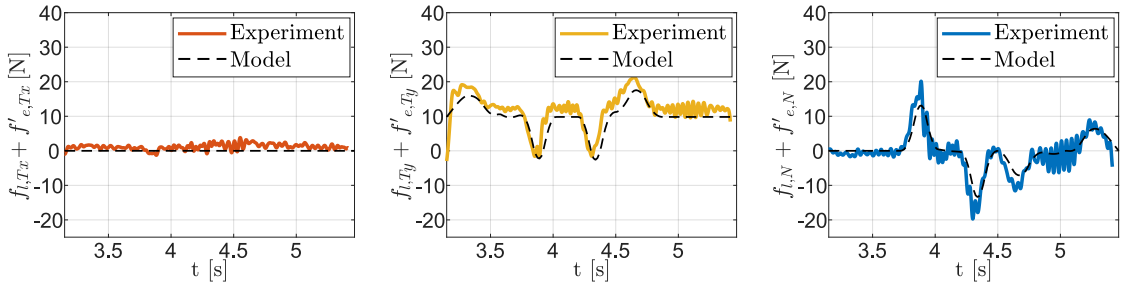


(b) Varying virtual penetration Δz.

Figure 4.21: Assessment on the application of the net wrench on both the object and the gripper for the c-motion.

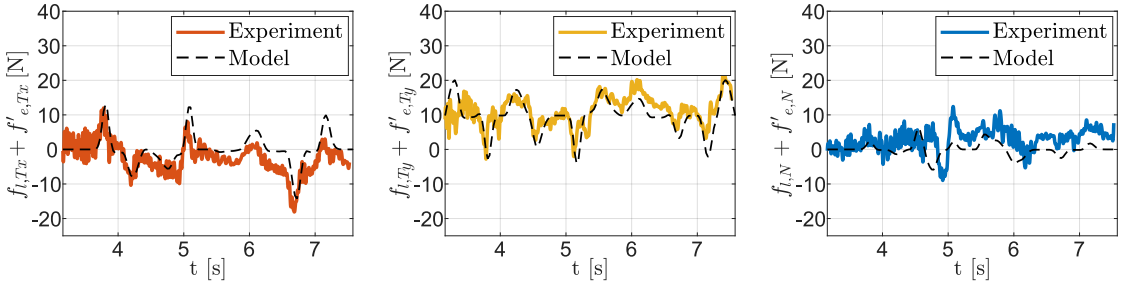


(a) Constant virtual penetration Δz.

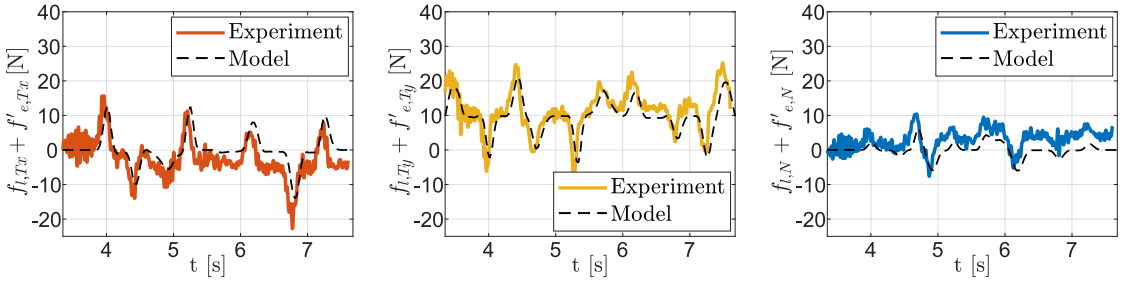


(b) Varying virtual penetration Δz.

Figure 4.22: Assessment on the application of the net wrench on both the object and the gripper for the r-motion.



(a) Constant virtual penetration Δz.



(b) Varying virtual penetration Δz.

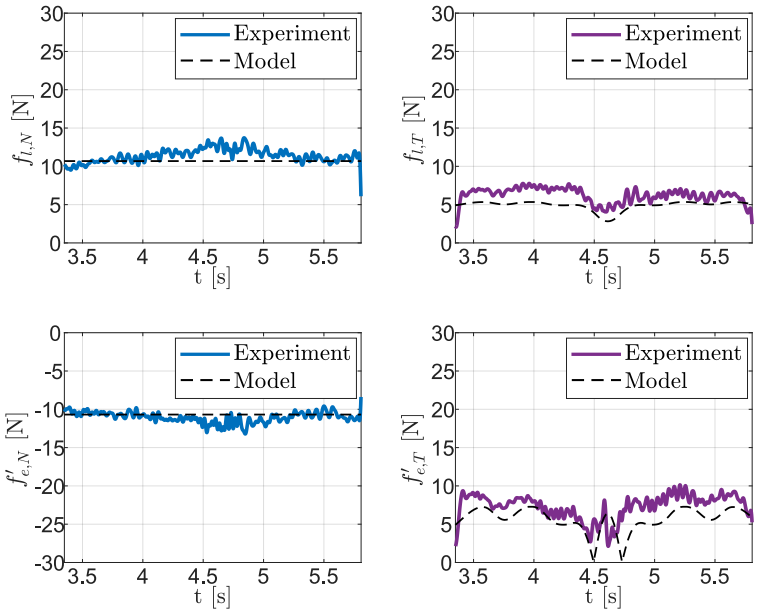
Figure 4.23: Assessment on the application of the net wrench on both the object and the gripper for the g-motion.

F_l. Recalling Equation (4.44), this sum only depends on the motion law and, thus, is not affected by either the subjective choices related to force distributions or the presence of the gripper on the right cobot. In each subfigure, the left-most plot displays the sum $f_{l,Tx} + f'_{e,Tx}$, the central plot the sum $f_{l,Ty} + f'_{e,Ty}$ and the right-most plot reports the quantity $f_{l,N} + f'_{e,,N}$. In particular, the coloured solid lines indicate the experimental trends, and the black dashed lines are the model predictions. The good correspondence between the model and the experiment demonstrates that the net force exerted by the cobots on the object and the gripper equals the desired dynamics, as mentioned in Equation (4.44), hence making the object follow the prescribed trajectory.

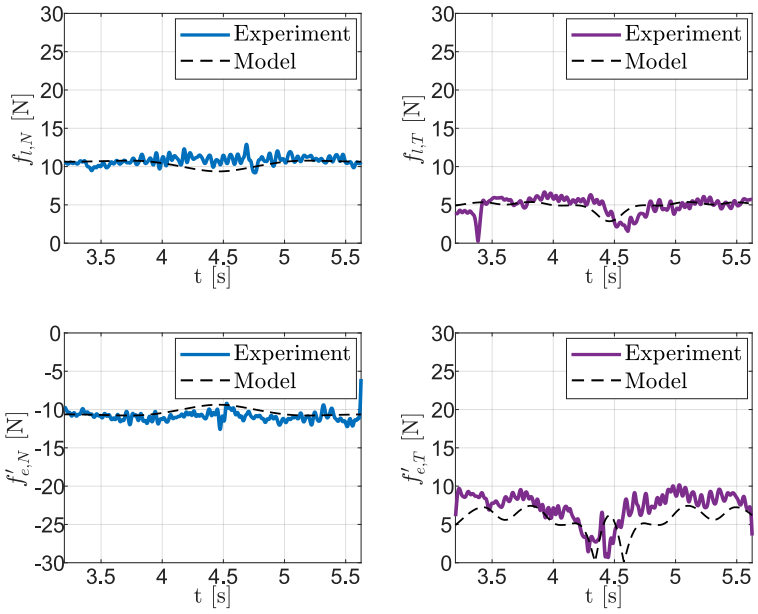
In Figures 4.24, 4.25, 4.26, 4.27, the trends of the normal forces exerted by the cobots $f_{l,N}$ and $f'_{e,N}$ and of the tangential norms $f_{l,T}$ and $f'_{e,T}$ are reported. Analogously to the case of Figures from 4.20 to 4.23, the solid lines indicate the experimental quantities, compared with the model ones in black dashed lines. For all motions, the model captures the experimental trends of the forces, with some tolerable discrepancies, especially if the results of the c-motion and the g-motion are taken into account. Nonetheless, the good adherence of the model predictions with reality proves the reasonableness of the assumption on the force distribution described in Section 4.6.2. This also finds confirmation by looking at Table 4.4, where the mean value of the relative errors $\gamma_{l,d}$ and $\gamma_{e,d}$ between the experiment and the model of the normal force and the tangential norm ($d = N, T$) on both cobots, is reported. For a better explanation, the relative errors are computed as follows:

$$\gamma_{l,d}(t_i) = \frac{|f_{l,d-exp}(t_i) - f_{l,d-mod}(t_i)|}{|f_{l,d-mod}(t_i)|} \times 100\%, \quad d = N, T \tag{4.51}$$

$$\gamma_{e,d}(t_i) = \frac{|f'_{e,d-exp}(t_i) - f'_{e,d-mod}(t_i)|}{|f'_{e,d-mod}(t_i)|} \times 100\%, \quad d = N, T \tag{4.52}$$

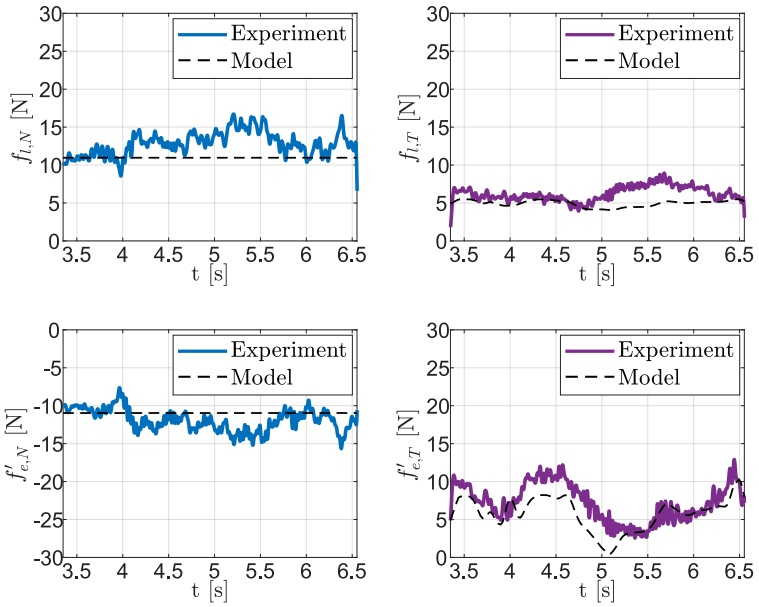


(a) Constant virtual penetration Δz.

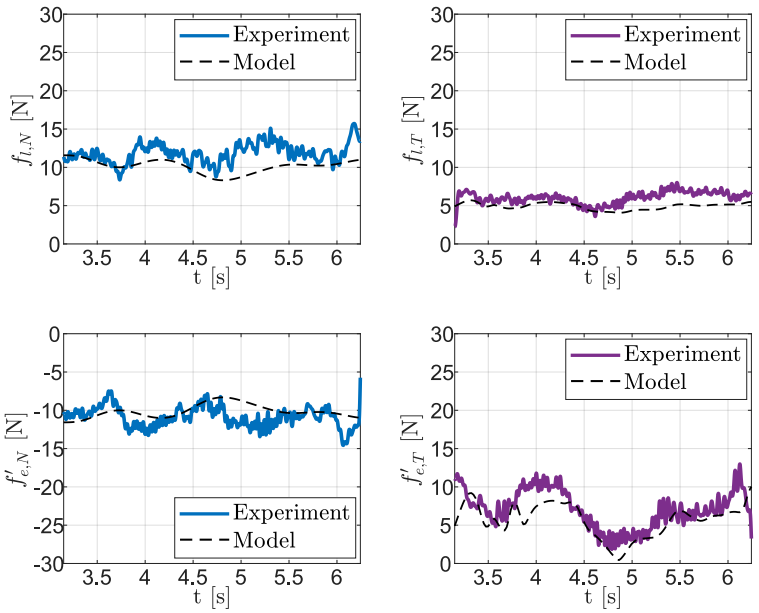


(b) Varying virtual penetration Δz.

Figure 4.24: Assessment on the assumed force distribution for the l-motion.

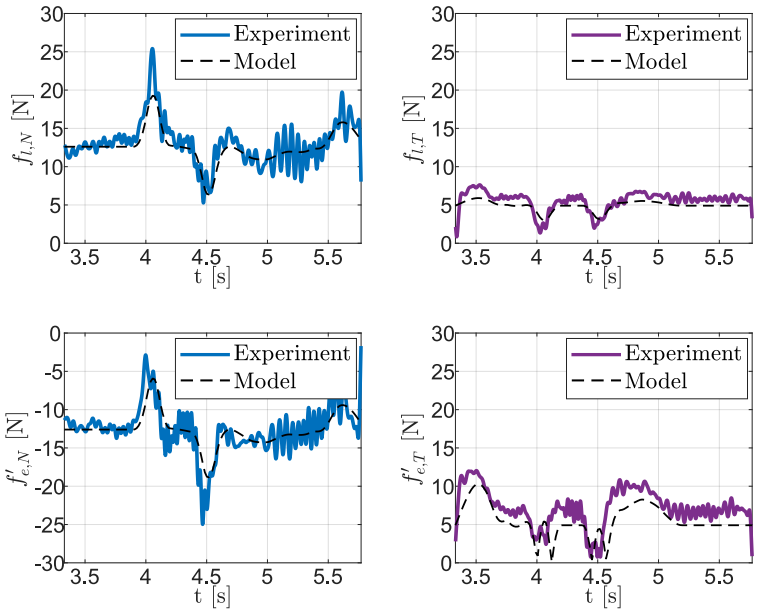


(a) Constant virtual penetration Δz.

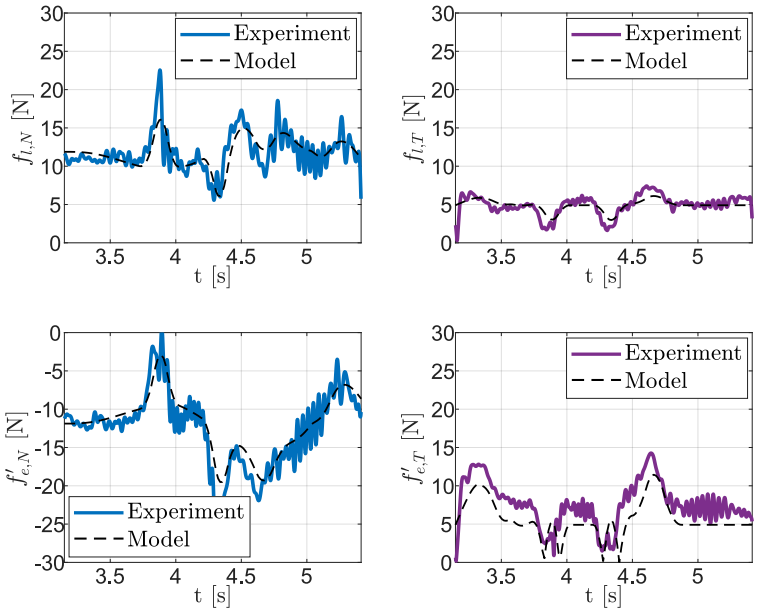


(b) Varying virtual penetration Δz.

Figure 4.25: Assessment on the assumed force distribution for the c-motion.

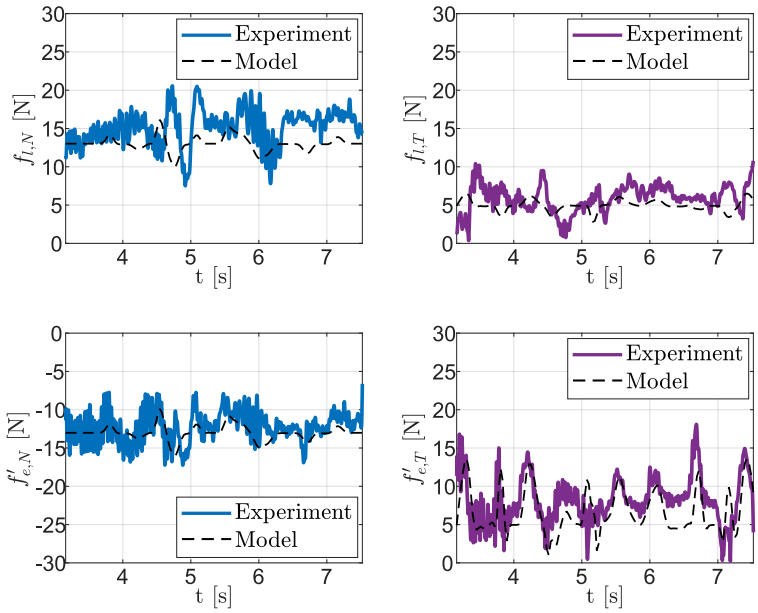


(a) Constant virtual penetration Δz.

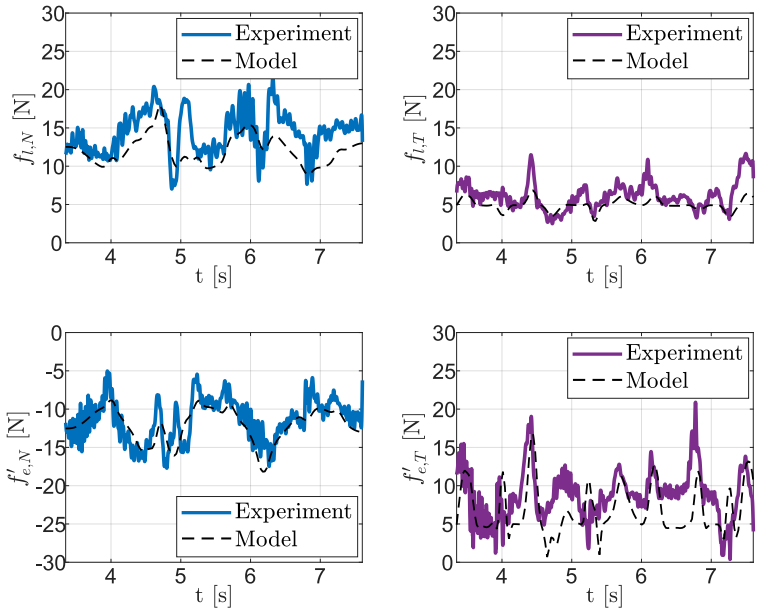


(b) Varying virtual penetration Δz.

Figure 4.26: Assessment on the assumed force distribution for the r-motion.



(a) Constant virtual penetration Δz.



(b) Varying virtual penetration Δz.

Figure 4.27: Assessment on the assumed force distribution for the g-motion.

Table 4.4: Mean value of the relative error, considering the normal and tangential forces read by the two sensors.

	Constant Δz	Varying Δz
l-motion	$\gamma_{l,N-med} = 8.3\%$	$\gamma_{l,N-med} = 6.8\%$
	$\gamma_{l,T-med} = 29.8\%$	$\gamma_{l,T-med} = 17.6\%$
	$\gamma_{e,N-med} = 5.6\%$	$\gamma_{e,N-med} = 6.8\%$
	$\gamma_{e,T-med} = -$	$\gamma_{e,T-med} = 88.4\%$
c-motion	$\gamma_{l,N-med} = 18.5\%$	$\gamma_{l,N-med} = 18.5\%$
	$\gamma_{l,T-med} = 28.5\%$	$\gamma_{l,T-med} = 24.0\%$
	$\gamma_{e,N-med} = 12.8\%$	$\gamma_{e,N-med} = 13.6\%$
	$\gamma_{e,T-med} = 57.0\%$	$\gamma_{e,T-med} = 48.1\%$
r-motion	$\gamma_{l,N-med} = 9.9\%$	$\gamma_{l,N-med} = 11.2\%$
	$\gamma_{l,T-med} = 18.5\%$	$\gamma_{l,T-med} = 13.9\%$
	$\gamma_{e,N-med} = 11.2\%$	$\gamma_{e,N-med} = 14.8\%$
	$\gamma_{e,T-med} = -$	$\gamma_{e,T-med} = -$
g-motion	$\gamma_{l,N-med} = 21.7\%$	$\gamma_{l,N-med} = 22.5\%$
	$\gamma_{l,T-med} = 37.7\%$	$\gamma_{l,T-med} = 27.9\%$
	$\gamma_{e,N-med} = 12.1\%$	$\gamma_{e,N-med} = 13.5\%$
	$\gamma_{e,T-med} = 53.9\%$	$\gamma_{e,T-med} = 76.2\%$

As far as the left cobot is considered, the value of $\gamma_{l,N-med}$ is always below the 22.5%, confirming, as in Table 4.3, that the experimental trend of the normal force exerted by the left cobot is reliable w.r.t. to the predicted one. A better result is obtained regarding the normal force of the right cobot, with a maximum value of $\gamma_{e,N-med}$ that reaches the 14.8%. The correspondence between the experiment and the model loses reliability if the mean value of the relative error along the tangential direction is considered. In particular, for the left cobot, $\gamma_{l,T-med}$ reaches a value of 37.7% in correspondence of the g-motion with a constant Δz. A different investigation is needed for the right cobot, for which $\gamma_{e,T-med}$ presents high values, that are omitted in some cases (v. the symbol $-$ in Table 4.4). The fact that the tangential norm of the force $\mathbf{f}'_e$ in the model scenario is near to small values in correspondence of some time instants (due to the fact that $\mathbf{f}'_e$ does not take into account the gripper gravity) makes the ratio $\gamma_{e,T}$ very unstable, namely

$$\lim_{|f'_{e,T-mod}(t_i)|\to 0} \gamma_{e,T}(t_i) = \infty. \tag{4.53}$$

4.7 Conclusive Remarks

In this Chapter, the prescription of a virtual penetration on the manipulated object was adopted to accomplish the dual-arm task. The virtual penetration, aimed at exerting an internal force along the normal-contact direction between the cobot end-effectors and the object, was inserted within the inputs of a constrained-optimization problem. The resolution of the time-optimal trajectory planning considered, for each path followed by the two cobots, an optimized constant value and an optimized varying trend of the virtual penetration. The motions were performed by employing two collaborative robots, which were controlled in ROS, and an admittance-control strategy was

implemented.
The experiments showed that no evident slipping of the object occurred even considering fast motions, hence making an advance in the state of the art where the value of the virtual penetration was found by trial-and-error procedures and slipping of the object was detected. The correct exertion of the net wrench on the object was confirmed by the force measurements, thus ensuring that the object followed the desired trajectory. A quantitative analysis was reported, highlighting that the accuracy index regarding the application of the normal force on the object was always under an acceptable threshold. This proved the efficacy of the control, especially considering the case with an optimized varying trend of the virtual penetration. In this case, the lower accuracy index not only proved a more reliable tracking of the commanded normal force w.r.t. the case with a constant virtual penetration, but it also granted a lower internal prestress on the object. Finally, the assumption on the force distribution among the two cobots, needed when hyperstatic problems, such as the one regarding dual-arm manipulation, have to be taken into account, turned out to be reasonable by comparing the experimental results with the model ones.
Further developments will see the extension of the experiments to trajectories with a varying orientation of the object. In addition, a deeper investigation on the force distribution will be addressed, by exploiting previous research [71, 72].
To conclude, point-to-point motions (in which only the initial and final poses are fixed) will be considered to release the object from following a prescribed path. This approach can be pursued offline, by making the optimization algorithm find the shortest path to be followed in minimal time and fulfilling the friction-cone conditions, also taking into account collision-avoidance constraints [73, 74].

Chapter 5

Conclusions

In this Thesis, the optimal control problem of single- and dual-arm serial robots was formulated to achieve the time-optimal handling of liquids and objects.
In Chapter 3, the planning of time-optimal anti-sloshing trajectories of an industrial serial robot carrying a cylindrical container filled with liquid was addressed considering 1-dimensional and 2-dimensional planar motions. The technique for the estimation of the *sloshing* height, firstly proposed in [29], was presented for 2-dimensional motions, and the model was extended to 3-dimensional motions. The quantitative and qualitative analysis of the experimental validation showed good reliability considering 1-dimensional and 2-dimensional motions with an acceleration of the container up to 9.5m/s^2. The performed 3-dimensional motions were obtained from the previous 2-dimensional ones by adding a vertical acceleration along the z-axis (up to 5m/s^2). The model extension was justified by achieving better accuracy indexes, representing the mean absolute error between the experimental results and the model predictions, compared to the adoption of the simpler 2-dimensional model.
For each of the 2-dimensional paths employed in the validation campaign, three optimizations were solved with different values of the sloshing-height thresholds. The comparison between the non-optimized and the optimized motions showed that the benefit in terms of sloshing-height reduction was obtained at the expense of a reasonable execution-time increase. In addition, in the optimized motions, the stringent limits imposed on the sloshing height were efficiently satisfied.
Future work will consider modeling the sloshing height for square-section containers and a further validation campaign with larger vertical accelerations. A sensitivity analysis will also be carried out to evaluate the reliability of the model w.r.t. the container dimensions. In the end, anti-sloshing trajectories will be planned for 3-dimensional motions, and different approaches for resolving the optimization problem will be considered.
The second topic of the Thesis, addressed in Chapter 4, regarded the time-optimal trajectory planning for dual-arm object handling. The main problem affecting the manipulation strategy called *cooperative grasping* is represented by the unilateral contact between the robots and the object, which may cause slipping during motion. To overcome this drawback, a virtual penetration was considered, aimed at generating an internal prestress in the contact-normal direction. The virtual penetration, together with the path parameter, was chosen as control input of the optimal control problem. This way, the resolution of the optimization provided, for each of the considered 3-dimensional translational motions, the minimum virtual penetration able to grant the

fulfillment of the friction-cone constraint, simultaneously executing the stable grasp of the object in minimal time.
Experiments were performed by employing two collaborative robots controlled in ROS, and an admittance control strategy was implemented. During the motions, the object was firmly handled by the cobots, even for fast motions, hence making progress w.r.t. previous work [59], where the value of the virtual penetration was found by trial and error, and slipping was detected.
Experimental results proved the correct exertion of the net wrench on the object, thus granting that the object faithfully followed the desired trajectory. An accuracy index expressing the error between the expected normal force on the object and its corresponding measured quantity showed the reliability of the control strategy. Furthermore, the force distribution among the two cobots, assumed during the model formulation, was revealed to be reasonable.
Additional experiments will foresee 6-dimensional motions of the object, hence considering an angular velocity other than zero. Regarding the optimization problem, point-to-point motions will be deemed to avoid constraining the object from following a prescribed path. This way, the algorithm will find the shortest path connecting the initial pose to the final one, respecting the criterion of minimum time, and fulfilling the friction-cone conditions, with the addition of collision-avoidance constraints.

Acknowledgement

The author would like to thank the Robotics Institute[1] of the Johannes Kepler University (Linz, Austria) for having hosted him during his 6-month period of research abroad and for the help in the sloshing validation campaign.

The author also would like to express his gratitude to the IMA[2] R&D Department for having provided him with the hardware and the setup needed to perform the dual-arm experiments.

[1] https://www.jku.at/en/institute-of-robotic
[2] https://ima.it

Bibliography

[1] J. T. Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*. Society for Industrial and Applied Mathematics, 2010.

[2] J. Nocedal and S. J. Wright, *Numerical Optimization*. Springer New York, 2006.

[3] J. Andersson, J. Gillis, G. Horn, J. Rawlings, and M. Diehl, "CasADi: A Software Framework for Nonlinear Optimization and Optimal Control," *Mathematical Programming Computation*, vol. 11, pp. 1–36, 2019.

[4] A. Wächter and L. T. Biegler, "On the Implementation of an Interior-point Filter Line-search Algorithm for Large-scale Nonlinear Programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, 2005.

[5] R. Fletcher and S. Leyffer, "Nonlinear programming without a penalty function," *Mathematical Programming*, vol. 91, no. 2, pp. 239–269, 2002.

[6] A. V. Fiacco and G. P. McCormick , *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*. John Wiley and Sons, NewYork, 1968.

[7] R. H. Byrd, G. Liu, and J. Nocedal, "On the local behavior of an interior point method for nonlinear programming," *Numerical Analysis*, 1997.

[8] A. Wächter and L. T. Biegler, "Line Search Filter Methods for Nonlinear Programming: Motivation and Global Convergence," *SIAM Journal on Optimization*, vol. 16, no. 1, pp. 1–31, 2005.

[9] R. Fletcher, *Practical Methods of Optimization*. John Wiley and Sons, New York, 1987.

[10] A. Reiter, *Optimal Path and Trajectory Planning for Serial Robots*. Springer Fachmedien Wiesbaden, 2020.

[11] R. Di Leva, M. Carricato, H. Gattringer, and A. Müller, "Sloshing Dynamics Estimation for Liquid-filled Containers under 2-Dimensional Excitation," in *2021 10th ECCOMAS Thematic Conference on Multibody Dynamics, Budapest, Hungary*, pp. 80–89, 2021.

[12] R. Di Leva, M. Carricato, H. Gattringer, and A. Müller, "Time-Optimal Trajectory Planning for Anti-Sloshing 2-Dimensional Motions of an Industrial Robot," in *2021 IEEE 20th International Conference on Advanced Robotics (ICAR), Ljubljana, Slovenia*, pp. 32–37, 2021.

[13] R. Di Leva, M. Carricato, H. Gattringer, and A. Müller, "Sloshing Dynamics Estimation for Liquid-filled Containers Performing 3-Dimensional Motions: Modeling and Experimental Validation," *Multibody System Dynamics*, vol. 56, no. 2, pp. 153–171, 2022.

[14] B. Moya, D. Gonzalez, I. Alfaro, F. Chinesta, and E. Cueto, "Learning Slosh Dynamics by Means of Data," *Computational Mechanics*, vol. 64, no. 2, pp. 511–523, 2019.

[15] B. Moya, I. Alfaro, D. Gonzalez, F. Chinesta, and E. Cueto, "Physically Sound, Self-learning Digital Twins for Sloshing Fluids," *PLOS ONE*, vol. 15, no. 6, pp. 1–16, 2020.

[16] S. S. Kolukula and P. Chellapandi, "Nonlinear Finite Element Analysis of Sloshing," *Advances in Numerical Analysis*, vol. 2013, pp. 1–10, 2013.

[17] C. Z. Wang and B. C. Khoo, "Finite Element Analysis of Two-dimensional Nonlinear Sloshing Problems in Random Excitations," *Ocean Engineering*, vol. 32, no. 2, pp. 107–133, 2005.

[18] M. Schörgenhumer and A. Eitzlmayr, "Modeling of Liquid Sloshing with Application in Robotics and Automation," *IFAC-PapersOnLine*, vol. 52, no. 15, pp. 253–258, 2019.

[19] M. D. Green and J. Peiró, "Long Duration SPH Simulations of Sloshing in Tanks with a Low Fill Ratio and High Stretching," *Computers & Fluids*, vol. 174, pp. 179–199, 2018.

[20] J. Huang and X. Zhao, "Control of Three-Dimensional Nonlinear Slosh in Moving Rectangular Containers," *Journal of Dynamic Systems, Measurement, and Control*, vol. 140, no. 8, 2018.

[21] O. M. Faltinsen, O. F. Rognebakke, and A. N. Timokha, "Resonant Three-dimensional Nonlinear Sloshing in a Square-base Basin," *Journal of Fluid Mechanics*, vol. 487, p. 1–42, 2003.

[22] R. A. Ibrahim, *Liquid Sloshing Dynamics: Theory and Applications*. Cambridge University Press, 2005.

[23] W. Aribowo, T. Yamashita, and K. Terashima, "Integrated Trajectory Planning and Sloshing Suppression for Three-dimensional Motion of Liquid Container Transfer Robot Arm," *Journal of Robotics*, vol. 2015, 2015.

[24] J. Reinhold, M. Amersdorfer, and T. Meurer, "A Dynamic Optimization Approach for Sloshing Free Transport of Liquid Filled Containers using an Industrial Robot," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China*, pp. 2336–2341, 2019.

[25] L. Moriello, L. Biagiotti, C. Melchiorri, and A. Paoli, "Control of Liquid Handling Robotic Systems: A Feed-forward Approach to Suppress Sloshing," in *2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore*, pp. 4286–4291, 2017.

[26] L. Moriello, L. Biagiotti, C. Melchiorri, and A. Paoli, "Manipulating Liquids with Robots: A Sloshing-free Solution," *Control Engineering Practice*, vol. 78, pp. 129–141, 2018.

[27] L. Biagiotti, D. Chiaravalli, L. Moriello, and C. Melchiorri, "A Plug-In Feed-Forward Control for Sloshing Suppression in Robotic Teleoperation Tasks," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain*, pp. 5855–5860, 2018.

[28] H. Bauer, "Nonlinear Mechanical Model for the Description of Propellant Sloshing," *AIAA Journal*, vol. 4, pp. 1662–1668, 1966.

[29] L. Guagliumi, A. Berti, E. Monti, and M. Carricato, "A Simple Model-based Method for Sloshing Estimation in Liquid Transfer in Automatic Machines," *IEEE Access*, vol. 9, pp. 129347–129357, 2021.

[30] L. Guagliumi, A. Berti, E. Monti, and M. Carricato, "Anti-Sloshing Trajectories for High-Acceleration Motions in Automatic Machines," *Journal of Dynamic Systems, Measurement, and Control*, vol. 144, no. 7, 2022.

[31] W. Aribowo, T. Yamashita, K. Terashima, and H. Kitagawa, "Input Shaping Control to Suppress Sloshing on Liquid Container Transfer using Multi-joint Robot Arm," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Taipei, Taiwan*, pp. 3489–3494, 2010.

[32] L. Biagiotti and C. Melchiorri, *Trajectory Planning for Automatic Machines and Robots*. Springer Berlin, 2009.

[33] Z. Shiller and S. Dubowsky, "On the Optimal Control of Robotic Manipulators with Actuator and End-effector Constraints," in *1985 IEEE International Conference on Robotics and Automation (ICRA), St. Louis, MO, USA*, pp. 614–620, 1985.

[34] H. F. Bauer, "Tables of Zeros of Cross Product Bessel Functions," *Mathematics of Computation*, vol. 18, no. 85, pp. 128–135, 1964.

[35] L. Piegl and W. Tiller, *The NURBS Book*. Springer Berlin, 1995.

[36] M. Neubauer, H. Gattringer, and H. Bremer, "A Persistent Method for Parameter Identification of a Seven-axes Manipulator," *Robotica*, vol. 33, no. 5, p. 1099–1112, 2015.

[37] W. Khalil and E. Dombre, *Modeling, Identification and Control of Robots*. Elsevier, Butterworth-Heinemann, 2002.

[38] H. Gattringer, A. Müller, M. Oberherber, and D. Kaserer, "Time-Optimal Path Following for Robotic Manipulation of Loosely Placed Objects: Modeling and Experiment," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 8450–8455, 2020.

[39] B. Siciliano, L. Sciavicco, L. Villani and G. Oriolo, *Robotics: Modelling, Planning and Pontrol*. Springer London, 2009.

[40] R. Di Leva, M. Carricato, H. Gattringer, and A. Müller, "Time-Optimal Trajectory Planning for Dual-Arm Cooperative Grasping Based on Minimal Internal Force," 2023. Under Review.

[41] V. V. Unhelkar, S. Dorr, A. Bubeck, P. A. Lasota, J. Perez, H. C. Siu, J. C. Boerkoel, Q. Tyroller, J. Bix, S. Bartscher, and J. A. Shah, "Mobile Robots for Moving-Floor Assembly Lines: Design, Evaluation, and Deployment," *IEEE Robotics and Automation Magazine*, vol. 25, no. 2, pp. 72–81, 2018.

[42] N. Pedrocchi, F. Vicentini, M. Matteo, and L. M. Tosatti, "Safe Human-Robot Cooperation in an Industrial Environment," *International Journal of Advanced Robotic Systems*, vol. 10, no. 1, 2013.

[43] S. Comari, R. Di Leva, M. Carricato, S. Badini, A. Carapia, G. Collepalumbo, A. Gentili, C. Mazzotti, K. Staglianò, and D. Rea, "Mobile Cobots for Autonomous Raw-material Feeding of Automatic Packaging Machines," *Journal of Manufacturing Systems*, vol. 64, pp. 211–224, 2022.

[44] S. Comari and M. Carricato, "Vision-Based Robotic Grasping of Reels for Automatic Packaging Machines," *Applied Sciences*, vol. 12, no. 15, p. 7835, 2022.

[45] K. Berntorp, K.-E. Arzen, and A. Robertsson, "Mobile Manipulation with a Kinematically Redundant Manipulator for a Pick-and-place Scenario," in *2012 IEEE International Conference on Control Applications, Dubrovnik, Croatia*, pp. 1596–1602, IEEE, 2012.

[46] M. Nieuwenhuisen, D. Droeschel, D. Holz, J. Stuckler, A. Berner, J. Li, R. Klein, and S. Behnke, "Mobile Bin Picking with an Anthropomorphic Service Robot," in *2013 IEEE International Conference on Robotics and Automation (ICRA), Karlsruhe, Germany*, pp. 2327–2334, 2013.

[47] A. Baldassarri, G. Innero, R. Di Leva, G. Palli, and M. Carricato, "Development of a Mobile Robotized System for Palletizing Applications," in *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Vienna, Austria*, pp. 395–401, 2020.

[48] J. Aleotti, A. Baldassarri, M. Bonfè, M. Carricato, D. Chiaravalli, R. Di Leva, C. Fantuzzi, S. Farsoni, G. Innero, D. L. Rizzini, C. Melchiorri, R. Monica, G. Palli, J. Rizzi, L. Sabattini, G. Sampietro, and F. Zaccaria, "Toward Future Automatic Warehouses: An Autonomous Depalletizing System Based on Mobile Manipulation and 3D Perception," *Applied Sciences*, vol. 11, no. 13, p. 5959, 2021.

[49] F. Zaccaria, A. Baldassarri, G. Palli, and M. Carricato, "A Mobile Robotized System for Depalletizing Applications: Design and Experimentation," *IEEE Access*, vol. 9, pp. 96682–96691, 2021.

[50] R. Krug, T. Stoyanov, V. Tincani, H. Andreasson, R. Mosberger, G. Fantoni, and A. J. Lilienthal, "The Next Step in Robot Commissioning: Autonomous Picking and Palletizing," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 546–553, 2016.

[51] G. Michalos, S. Makris, J. Spiliotopoulos, I. Misios, P. Tsarouchi, and G. Chryssolouris, "ROBO-PARTNER: Seamless Human-Robot Cooperation for Intelligent, Flexible and Safe Operations in the Assembly Factories of the Future," *Procedia CIRP*, vol. 23, pp. 71–76, 2014.

[52] J. Saenz, F. Penzlin, C. Vogel, and M. Fritzsche, "Valeri — A Collaborative Mobile Manipulator for Aerospace Production," in *Advances in Cooperative Robotics*, pp. 186–195, 2016.

[53] C. Smith, Y. Karayiannidis, L. Nalpantidis, X. Gratal, P. Qi, D. V. Dimarogonas, and D. Kragic, "Dual arm manipulation—A survey," *Robotics and Autonomous Systems*, vol. 60, no. 10, pp. 1340 – 1353, 2012.

[54] F. Caccavale, P. Chiacchio, A. Marino, and L. Villani, "Six-DOF Impedance Control of Dual-Arm Cooperative Manipulators," *IEEE/ASME Transactions on Mechatronics*, vol. 13, no. 5, pp. 576–586, 2008.

[55] H. Sadeghian, F. Ficuciello, L. Villani, and M. Keshmiri, "Global Impedance Control of Dual-Arm Manipulation for Safe Interaction," *IFAC Proceedings Volumes*, vol. 45, no. 22, pp. 767 – 772, 2012.

[56] W. Gueaieb, F. Karray, and S. Al-Sharhan, "A Robust Hybrid Intelligent Position/Force Control Scheme for Cooperative Manipulators," *IEEE/ASME Transactions on Mechatronics*, vol. 12, no. 2, pp. 109–125, 2007.

[57] B. Chen, Y. Wang, and P. Lin, "A Hybrid Control Strategy for Dual-arm Object Manipulation Using Fused Force/Position Errors and Iterative Learning," in *2018 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), Auckland, New Zealand*, pp. 39–44, 2018.

[58] M. Bjerkeng, J. Schrimpf, T. Myhre, and K. Y. Pettersen, "Fast Dual-arm Manipulation using Variable Admittance Control: Implementation and Experimental Results," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Chicago, IL, USA*, pp. 4728–4734, 2014.

[59] D. Kaserer, H. Gattringer, and A. Müller, "Time Optimal Motion Planning and Admittance Control for Cooperative Grasping," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2216–2223, 2020.

[60] G. Zeng and A. Hemami, "An Overview of Robot Force Control," *Robotica*, vol. 15, no. 5, p. 473–482, 1997.

[61] M. Schumacher, J. Wojtusch, P. Beckerle, and O. von Stryk, "An Introductory Review of Active Compliant Control," *Robotics and Autonomous Systems*, vol. 119, pp. 185–200, 2019.

[62] M. T. Mason, "Compliance and Force Control for Computer Controlled Manipulators," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 11, no. 6, pp. 418–432, 1981.

[63] S. Chiaverini and L. Sciavicco, "The Parallel Approach to Force/position Control of Robotic Manipulators," *IEEE Transactions on Robotics and Automation*, vol. 9, no. 4, pp. 361–373, 1993.

[64] B. Siciliano and L. Villani, *Robot Force Control*. Springer US, 1999.

[65] N. Hogan, "Impedance Control: An Approach to Manipulation: Part I—Theory," *Journal of Dynamic Systems, Measurement, and Control*, vol. 107, no. 1, pp. 1–7, 1985.

[66] A. Müller, "Screw and Lie Group Theory in Multibody Kinematics," *Multibody System Dynamics*, vol. 43, no. 1, pp. 37–70, 2018.

[67] F. Caccavale, C. Natale, B. Siciliano, and L. Villani, "Six-DOF Impedance Control Based on Angle/Axis Representations," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 2, pp. 289–300, 1999.

[68] D. Kaserer, H. Gattringer, and A. Müller, "Admittance Control of a Redundant Industrial Manipulator without Using Force/Torque Sensors," in *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society, Florence, Italy*, pp. 5310–5315, 2016.

[69] I. D. Walker, R. A. Freeman, and S. I. Marcus, "Analysis of Motion and Internal Loading of Objects Grasped by Multiple Cooperating Manipulators," *The International Journal of Robotics Research*, vol. 10, no. 4, pp. 396–409, 1991.

[70] L.-T. Schreiber and C. Gosselin, "Determination of the Inverse Kinematics Branches of Solution Based on Joint Coordinates for Universal Robots-Like Serial Robot Architecture," *ASME Journal of Mechanisms and Robotics*, vol. 14, no. 3, 2021.

[71] T. Watanabe and T. Yoshikawa, "Grasping Optimization Using a Required External Force Set," *IEEE Transactions on Automation Science and Engineering*, vol. 4, no. 1, pp. 52–66, 2007.

[72] S. Erhart and S. Hirche, "Internal Force Analysis and Load Distribution for Cooperative Multi-Robot Manipulation," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1238–1243, 2015.

[73] A. Völz and K. Graichen, "An Optimization-Based Approach to Dual-Arm Motion Planning with Closed Kinematics," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain*, pp. 8346–8351, 2018.

[74] D. Kaserer, H. Gattringer, A. Müller, and M. Neubauer, "Optimal Point-To-Point Trajectory Planning with Collision Avoidance for Dual Arm Setup," in *2017 8th ECCOMAS Thematic Conference on Multibody Dynamics, Prague, Czech Republic*, pp. 645–654, 2017.