Contributions to the Performance Analysis of Continuum Parallel Robots

Federico Zaccaria

MEMOIRE DE DOCTORAT DE

L'ECOLE CENTRALE DE NANTES
ET UNIVERSITA' DI BOLOGNA

ECOLE DOCTORALE N° 602
Sciences de l'Ingénierie et des Systèmes
Spécialité : *Robotique*

Par
Federico ZACCARIA

Contributions à l'analyse de la performance des robots parallèles continus.

Projet de recherche doctoral présenté et soutenu à l'École Centrale de Nantes le 20 décembre 2023
Unité de recherche : UMR 6004 Laboratoire des Sciences du Numérique de Nantes (LS2N)

Rapporteurs avant soutenance :

Jessica BURGNER-KAHRS, Associate Professor, University of Toronto, Canada
Kanty RABENOROSOA, Professeur des Universités, Université de Franche-Comté

Composition du Jury :

Président : Marc GOUTTEFARDE, Directeur de recherche CNRS, Université de Montpellier
Examinateurs : Quentin PEYRON, Chargé de recherche, INRIA Lille
Matteo RUSSO, Assistant Professor, Università degli Studi di roma 'Tor Vergata', Italie
Elwan HERY, Maitre de Conférences, École Centrale de Nantes

Directeur de recherches doctorales : Sébastien BRIOT, Directeur de recherche CNRS École Centrale de Nantes
Directeur de recherches doctorales: Marco CARRICATO, Full professor Università di Bologna 'Alma Mater Studiorum', Italie

Co-enc.de recherches doctorales: Edoardo IDA', Assistant professor, Università di Bologna 'Alma Mater Studiorum', Italie.

Contributions to the Performance Analysis of Continuum Parallel Robots

A dissertaton submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Mechanics and Advanced Engineering Sciences at the University of Bologna and in Engineering and Systems Sciences at the École Centrale de Nantes.

Presented by Federico Zaccaria

Nantes, France, December 20th, 2023

Abstract

Continuum parallel robots (*CPRs*) are manipulators employing multiple flexible beams arranged in parallel and connected to a rigid end-effector. *CPRs* promise higher payload and accuracy than serial *CRs* while keeping great flexibility. As the risk of injury during accidental contacts between a human and a *CPR* should be reduced, *CPRs* may be used in large-scale collaborative tasks or assisted robotic surgery. There exist various *CPR* designs, but the prototype conception is rarely based on performance considerations, and the *CPRs* realization in mainly based on intuitions or rigid-link parallel manipulators architectures. This thesis focuses on the performance analysis of *CPRs*, and the tools needed for such evaluation, such as workspace computation algorithms. In particular, workspace computation strategies for *CPRs* are essential for the performance assessment, since the *CPRs* workspace may be used as a performance index or it can serve for optimal-design tools. Two new workspace computation algorithms are proposed in this manuscript, the former focusing on the workspace volume computation and the certification of its numerical results, while the latter aims at computing the workspace boundary only. Due to the elastic nature of *CPRs*, a key performance indicator for these robots is the stability of their equilibrium configurations. This thesis proposes the experimental validation of the equilibrium stability assessment on a real prototype, demonstrating limitations of some commonly used assumptions. Additionally, a performance index measuring the distance to instability is originally proposed in this manuscript. Differently from the majority of the existing approaches, the clear advantage of the proposed index is a sound physical meaning; accordingly, the index can be used for a more straightforward performance quantification, and to derive robot specifications.

Résumé

Les robots parallèles continus (*RPCs*) sont des manipulateurs utilisant plusieurs tiges flexibles disposées en parallèle et connectées à une plateforme rigide. Les *RCPs* promettent des capacités de charge et une précision plus élevées que les robots sériels continus, tout en gardant une grande flexibilité. Puisque le risque de blessure lors d'un contact accidentel entre un humain et un *CPR* devrait être réduit, les *RCPs* peuvent être utilisés dans des tâches collaboratives à grande échelle ou dans de tâches de chirurgie robotique assistée. Différentes architectures de *RCP* existent, mais la conception du prototype est rarement basée sur des considérations de performance, et la réalisation de *RCPs* est principalement à partir d'intuitions en utilisant d'architectures de manipulateurs parallèles rigides. Les themes de recherche de cette thèse portent sur l'analyse des performances des *RCPs*, et sur les outils nécessaires à une telle évaluation, ainsi que sur les algorithmes de calcul de leur espace de travail. En particulier, les stratégies de calcul de l'espace de travail pour les *RCPs* sont essentielles pour l'évaluation du performances, car l'espace de travail peut être utilisé comment un indice de performance, par example pour des outils de conception optimale. Deux nouveaux algorithmes de calcul de l'espace de travail sont proposés dans ce manuscrit, le premier se concentrant sur le calcul du volume de l'espace de travail et la certification de ses résultats numériques, et le second sur le calcul des bords de l'espace de travail uniquement. En raison de la nature élastique des *RCPs*, un indicateur de performance essentiel pour ces robots est la stabilité de leurs configurations d'équilibre. Cette thèse propose la validation expérimentale de l'évaluation de la stabilité des équilibres sur un prototype réel, démontrant les limites de certaines hypothèses couramment utilisées. De plus, un indice de performance mesurant la distance à l'instabilité est proposé dans ce manuscrit. Contrairement à la majorité des approches existantes, l'avantage évident de l'indice proposé est une signification physique bien défini.

Contents

Acronyms and Symbols

Acronyms

DoFs ..... Degrees of freedom;

CRs ..... Continuum Robots;

CPRs ..... Continuum Parallel Robots;

PCPRs ..... Planar Continuum Parallel Robots;

CTRs ..... Concentric Tube Robots;

FGSP ..... Forward Geometrico-Static Problem;

IGSP ..... Inverse Geometrico-Static Problem;

MIGSP ..... Modified Inverse Geometrico-static Problem;

IA ..... Interval Analysis;

OC ..... Optimal Control;

WS ..... Workspace

JS ..... Jointspace;

DCLI ..... Directional Critical Load Index

ODEs ..... Ordinary Differential Equations

IVP ..... Initial values problem

Symbols

$\mathbf{G} \in SE(3)$ .... Pose matrix;

$\mathbf{R} \in SO(3)$ ..... Orientation matrix;

$\mathbf{p} \in \mathbb{R}^3$ ..... Position vector;

$\boldsymbol{\xi} \in \mathbb{R}^6$ ..... Strain vector;

$\boldsymbol{\Gamma} \in \mathbb{R}^6$ ..... Stress vector;

$\mathbf{u} \in \mathbb{R}^3$ ..... Angular rate of change;

$\mathbf{v} \in \mathbb{R}^3$ ..... Linear rate of change;

$\mathbf{K} \in \mathbb{R}^{6\times6}$ ..... Stiffness matrix;

$\mathbf{K}_{BT} \in \mathbb{R}^{3\times3}$ ... Bending-torsion stiffness matrix;

$\mathbf{K}_{SE} \in \mathbb{R}^{3\times3}$ .... Shear-extensibility stiffness matrix;

$\mathbf{w}_p \in \mathbb{R}^6$ ..... Platform wrench;

$\mathbf{w}_d \in \mathbb{R}^6$ ..... Distributed wrench;

$\mathbf{f}_p \in \mathbb{R}^3$ ..... Platform force;

$\mathbf{f}_d \in \mathbb{R}^3$ ..... Distributed force;

$\mathbf{f}_c \in \mathbb{R}^3$ ..... Concentrated force;

$n \in \mathbb{R}$ ..... number of actuated variables;

$n_b \in \mathbb{R}$ ..... number of beams;

$n_c \in \mathbb{R}$ ..... number of platform variables;

$n_\phi \in \mathbb{R}$ ..... number of constraint equations;

$m \in \mathbb{R}$ ..... number of discretization variables;

$V \in \mathbb{R}$ ..... Potential energy;

$\mathbf{\Phi} \in \mathbb{R}^{n_\phi}$ ..... Constraint vector;

$\mathcal{L} \in \mathbb{R}$ ..... Lagrangian function;

$\mathbf{q}_a \in \mathbb{R}^n$ ..... Actuated variables;

$\mathbf{q}_p \in \mathbb{R}^{n_c}$ ..... Platform pose variables;

$\mathbf{q}_e \in \mathbb{R}^m$ ..... Discretization variables;

$\mathbf{q}_c \in \mathbb{R}^n$ ..... Controlled variables;

$\mathbf{q}_u \in \mathbb{R}^{n_c-n}$ ..... Uncontrolled platform variables;

$\mathbf{q}_d \in \mathbb{R}^{n_c-n+m}$ .. Passive variables;

$\mathbf{x} \in \mathbb{R}^{n_c+m}$ ..... Non-actuated variables;

$\boldsymbol{\lambda} \in \mathbb{R}^{n_\phi}$ ..... Lagrange multipliers;

$\mathbf{y} \in \mathbb{R}^{n+n_c+m+n_\phi}$ Configuration variables;

Introduction

Continuum parallel robots (*CPRs* in short) are manipulators made by a parallel arrangement of several flexible beams connected to a rigid end-effector. These manipulators promise to alleviate the main limitation of serial-like continuum robots, that is, the reduced payload capacity. As parallel manipulators, *CPRs* promise higher payload and greater accuracy while keeping great flexibility. The latter characteristic and reduced moving mass due to the placement of the actuators at the robot's base may reduce the risk of injury during accidental contact between a human and a *CPR*. Considering the expected advances of *CPRs*, possible applications of these manipulators may include large-scale collaborative tasks or assisted robotic surgery.

CPRs are relatively new, and several research directions still need to be explored. The majority of the related works focus on the static modelling of these robots: as a cause of their intrinsic elasticity, the geometry of *CPRs* is not sufficient to describe the pose of the whole robot, the *CPRs* configuration is defined by the elastic deformation of the links, and the position problems of *CPRs* do not admit any analytical solution, in general. The complexity of the *CPRs* modelling approach further complicates other aspects, such as control, trajectory planning, and performance quantification. In particular, this thesis focuses on the latter aspect, the performance analysis of *CPRs*.

Performance analysis of *CPRs* is currently exploited by utilizing rigid-link robot performance indices readapted to the continuum robotics framework; the majority of the existing performance indices are based on analysis of the Jacobian matrix or stiffness matrix separately, and the intrinsic coupling between geometry and elasticity of *CRs* is frequently neglected. In particular, two major research directions are identified: the workspace evaluation and the equilibrium stability assessment. On the one hand, the workspace computation problem of *CPRs* plays a crucial role in the performance evaluation of *CPRs*. The *CPRs* workspace may be used to get useful metrics (e.g. workspace volume), and it could be used as an objective for optimal design tools. However, at the time of this thesis started, workspace computation techniques are based on time-consuming approaches, and no work is devoted explicitly to the workspace computation of *CPRs*. On the other hand, equilibrium stability analysis is important for the practical usability of *CPRs*. Most of the state of the art performs the equilibrium stability assessment by employing optimal control approaches, but the complexity of these techniques is relevant. However, alternative methods exist, such as the evaluation Hessian matrix to assess the equilibrium stability, but these techniques still need to be experimentally verified. Additionally, metrics to measure the distance to instability are essential, but the few existing works lack a physical meaning of the measure.

This thesis focuses on the performance analysis of *CPRs*, and the contributions

of this thesis concern two topics: the workspace computation strategies of *CPRs* and the equilibrium stability assessment. Two workspace computation strategies are proposed in this thesis, focusing on i) the workspace computation and the certification of the numerical results, and ii) the workspace boundary reconstruction. Regarding the equilibrium stability assessment, the first contribution is related to the experimental validation of the equilibrium stability of a newly-proposed prototype. On the same topic, a performance index is also proposed in this thesis to measure the distance to the instability. To better highlight the contribution of this thesis, the manuscript is structured in three parts. Each part comprises two chapters.

- **Part I**. This part is dedicated to state-of-the-art analysis. Chapter 1 starts with an illustration of the existing *CPRs* architectures. Then, the state of the art concerning performance evaluation of *CRs* is analysed, with a focus on workspace evaluation and equilibrium stability assessment. Then, Chapter 2 illustrates the state-of-the-art modelling framework employed in this thesis.

- **Part II**. This part illustrates the advancement in the workspace evaluation of *CPRs* proposed in this thesis. Chapter 3 proposes a new algorithm for the workspace evaluation of planar *CPRs*. Thanks to an energy-based modelling strategy, and derivative approximation by finite differences, the Kantorovich theorem is applied to certify the existence, uniqueness, and convergence of the solution of the inverse geometrico-static problem at each step of the workspace computation. In Chapter 4, a new algorithm for the computation of workspace boundaries of continuum parallel robots (*CPRs*) is proposed. The proposed algorithm is based on a free-space exploration strategy and a boundary reconstruction algorithm to reconstruct the workspace borders while also identifying voids and holes in the workspace, in reduced computational time with respect to full workspace reconstruction techniques.

- **Part III**. This part illustrates the contribution of this thesis on the equilibrium stability assessment. Chapter 5 proposes the experimental validation of *CPRs* equilibrium stability prediction based on discretised modelling techniques. Unstable configurations that limit the robot workspace are theoretically and experimentally investigated. A new *CPR* prototype for planar applications is proposed, designed, and tested for the scope. Experiments demonstrate that, even though the prototype is theoretically planar, a model neglecting out-of-the-plane phenomena is inadequate to assess equilibrium stability limits. Chapter 6 introduces a new performance indices for the distance-to-instability measurement. The proposed index, namely, the critical load index, estimates the magnitude of a force that perturbates the *CR* equilibrium and it is proposed as a measure of the distance to instability. The major advantages of this metric are the intrinsic physical meaning, the practical interpretation of the results and the well-defined unit.

At the end of Part III, Chapter 7 draws conclusions of this thesis, highlights the contributions and limitations of the proposed works, and it illustrates possible future-work directions.

The work presented in this thesis results from a co-tutorship agreement (co-tutelle) between the University of Bologna and the École Centrale de Nantes: the

work has been partially developed at the Department of Industrial Engineering in Bologna (Italy) and at the Laboratoire de Sciences du Numéque in Nantes (France).

Part I

State of the art

Chapter 1

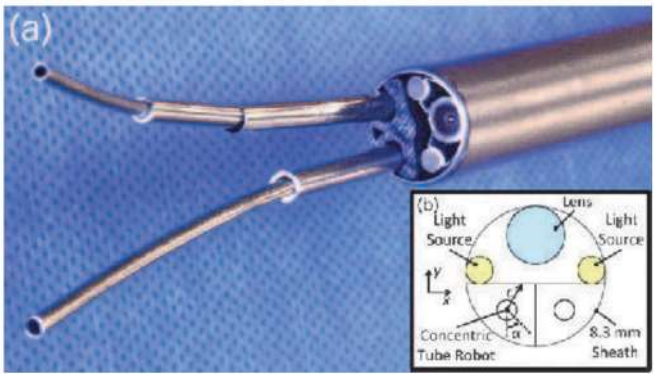
Architecture and Performance Analysis

Nowadays, most robots are mechanisms constructed from a series of rigid links connected by discrete single or multiple degrees of freedom (*DoFs* in short) joints, and a controlled movement is generated at some of these joints. The need for accuracy creates heavy mechanisms with stiff links and large passive sections supporting their own weight. Although this design might be essential for many practical operations where speed and accuracy play a key role, many practical applications exist where different attributes are required [1]. For instance, when dextrous manipulation in highly constrained and cluttered environments is required, hyper-redundant manipulators are well-suited [2] because they can conform to the environment shape (Fig. 1.1); this is achieved thanks to their peculiar structure, and thanks to their multiple actuated *DoFs* that exceed the minimal number required for the manipulator task. Another example is human-robot collaboration or cooperation [3], where the operative speed of the collaborative robots is usually reduced to comply with risk management rules, which are based on the limitation of the robot linear momentum.

As the complexity of the task or the environmental constraints increases, it is common to increase the degree of redundancy [2]. However, robot's joints possess a finite dimension, and there exists a minimum limit on their dimension, posing design challenges to create compact and dextrous hyper-redundant manipulators when the number of *DoFs* increases. Continuum robots are introduced to mitigate this issue. In the theoretical case, when the number of joints approaches infinity and the link length zero, the robot resembles a continuous structure where the joint dimension is negligible. In fact, continuum robots are manipulators that achieve the movement of their end-effector (*EE*) through the controlled displacement and deformation of slender elastic links, such as rods. Their design is inspired by biological structures like trunks, tentacles, snakes, and tendrils, and the morphology of their design has motivated researchers to model flexible robots [5]. Even if the term continuum robot is quite intuitive and it describes with simplicity the characteristics of these manipulators, some different definitions are available in the literature, such as the concept of continuously bendable structures [1], infinite degree of freedom structures [6], or systems that do not contain rigid links and identifiable joints [7]. Alternatively, it is possible to use the inclusive definition of [8], where continuum robots (*CRs* in short) are described as structures that can be deformed in a controllable way to perform a task.

Figure 1.1: An hyper-redundant manipulator working in a confined environment, courtesy of [4].

(a) The *CR* presented in [9]

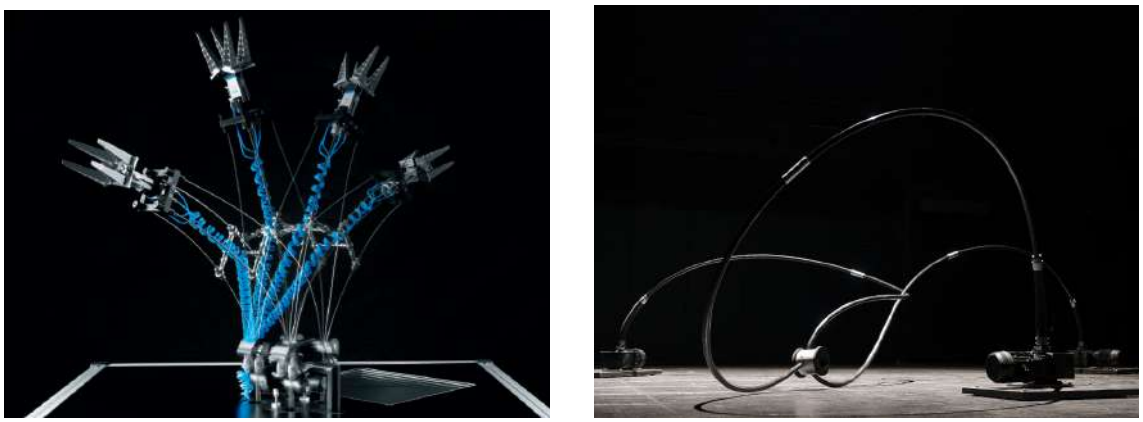


(b) The concentric tube robots of [10]

Figure 1.2: Two examples of continuum robots: a tendon-driven *CR* in a), and concentric tube robots in b).

There exist various *CRs* designs that suit different application scenarios. Tendon-driven continuum robots employ multiple tendons, or cables, to generate a controlled displacement of a highly flexible central backbone [9] (Fig. 1.2a). Concentric tube robots are made from several tubes that are nested within one another concentrically (Fig. 1.2b). These tubes are precurved and made of elastic material. When the tubes are grasped at their respective bases, and linear insertion/retraction and axial rotation motions are applied, tubes interact elastically and make one another bend and twist, thus determining the *EE* motion [10]. Also, continuum robots can be devised by installing several magnets on a flexible beam and governing the beam shape by applying a controlled magnetic field [11].

As in traditional robots, the classification between serial and parallel architectures also exists in continuum robots. *CRs* have historically been considered as serial devices [12] because they usually display long and slender shapes. Despite serial architectures being more famous for their applications in surgical tasks, con-



(a) (b)

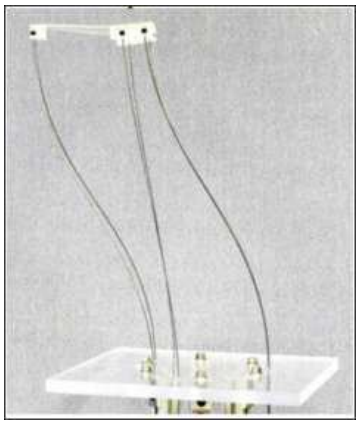
Figure 1.3: First *CPRs* prototypes: the Festo Bionic Tripod 3.0 in a), and the Nyloid sculpture in b).

tinuum parallel architectures are starting to be employed in research activities [13]. The interest in continuum parallel manipulators is due to the possibility of compactness, compliant motion, precision, and stability of parallel architectures. This thesis focuses on the performance analysis of continuum parallel robots (*CPRs*): this topic is of vital importance to enable performance-based design of *CPRs*. This Chapter, which aims to give an overview of the current state of the art in performance analysis, is structured as follows. Sec. 1.1 introduces *CPRs* design, and a summary of the existing prototypes is proposed, while Sec. 1.2 discusses the existing methodologies for the performance analysis of continuum robots. Then, the focus of the state-of-the-art analysis is restricted to two major areas: the workspace computation (Sec. 1.3), and the equilibrium stability assessment (Sec. 1.4).

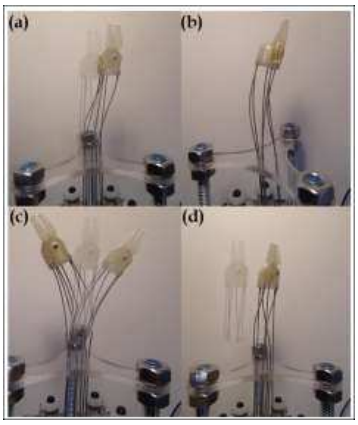
1.1 Continuum Parallel Robots

A *CPR* is a manipulator made by several flexible beams arranged in parallel and connected to a rigid *EE* [13]. Thanks to passive joints or rigid connections, the distal ends of the beams are connected to the *EE*, while each link is independently actuated by actuators placed at the base or by distributed actuation (e.g. tendons). Thus, the *CPR* motion is obtained by the controlled displacement and deformation of each link. *CPRs* promise a higher payload than serial *CRs* and greater accuracy [14]. Instead, compared with rigid-links parallel manipulators, *CPRs* display greater compliance and easier miniaturization to the scale of a few millimetres [15]. Additionally, *CPRs* intrinsic flexibility and reduced moving mass (due to the placement of the actuators at the base) can reduce the risk of human injury during accidental contacts for large-scale applications. Rigid-link parallel robots hardly ensure this important safety feature.

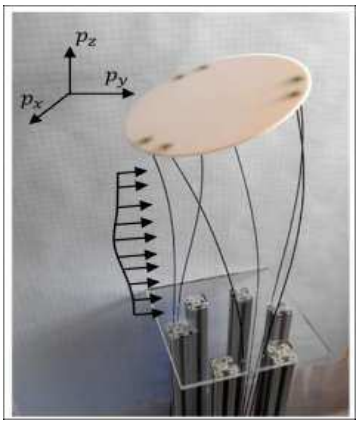
This flexible structure makes *CPRs* mechanically simple, lightweight, and compliant by design providing interesting features for robotic surgery [16] or large-scale collaborative tasks [17]. In this direction, *CPR* applications may include larger-scale industrial tasks traditionally performed by commercial cobots and may facilitate safer operations in human-shared environments. Thanks to their inherent lightweight and compliant design, collisions should result in reduced injury to the



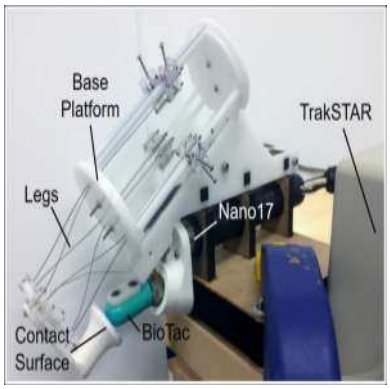
(a) Picture from [13]



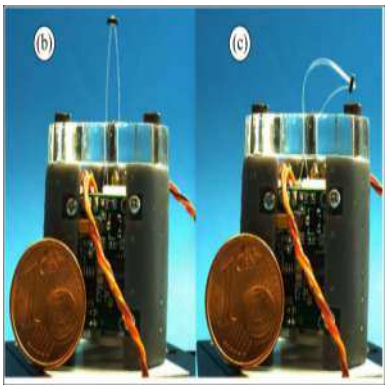
(b) Picture from [16]



(c) Picture from [23]



(d) Picture from [24]



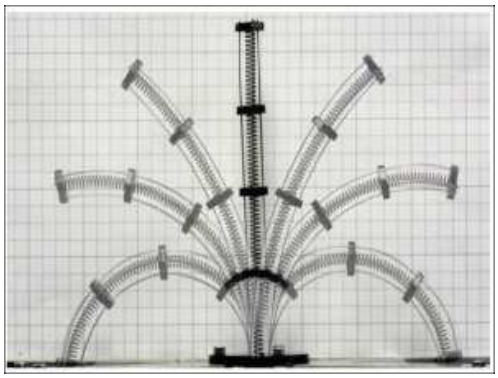
(e) Picture from [15]

Figure 1.4: Illustration of the existing extensible *CPRs*. These *CPRs* employ actuators that vary the length of the beam, employing a Gough-Stewart-like arrangement.

human operator.

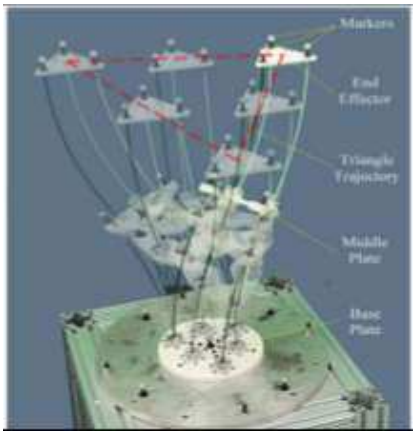
As *CPRs* are a relatively new kind of manipulators, numerous scientific questions are still open. In particular, the static modelling of *CPRs* received great attention from the scientific community [18]. In *CPRs*, this task is not trivial, since the geometry of the manipulator is not sufficient to describe the pose of the whole robot, and its configuration is defined by the elastic deformation of the links. This problem is called geometrico-static and, in *CPRs*, forward and inverse problems do not generally admit an analytical solution. Consequently, the geometrico-static model is usually simplified by introducing numerical approximations [19] with the approximate solution being computed by numerical schemes (chapter 2 is devoted to the description of *CRs* modelling strategies). Other open questions include the optimal design of *CPRs*, the feedback-control strategies [20], the trajectory planning [21], dynamic modelling [22], and the performance quantification [18]. In particular, the work of this thesis focuses on the performance analysis of *CPRs*.

To better understand the importance of performance analysis, let us first analyze the existing state of the art regarding *CPRs* design. Before the first academic *CPR* prototype of [13], proposed in 2014, the Festo Bionic Tripod [25] was proposed as a research prototype in 2010. The Bionic Tripod 3.0 (see Fig. 1.3a), resemble the current *CPR* paradigm in all its aspects: several flexible links are translated at the base and connected to a rigid platform to obtain a controlled *EE* displacement. In 2013, the Nyloid sculpture appeared as a large-scale *CPR* with 6-meter-long legs

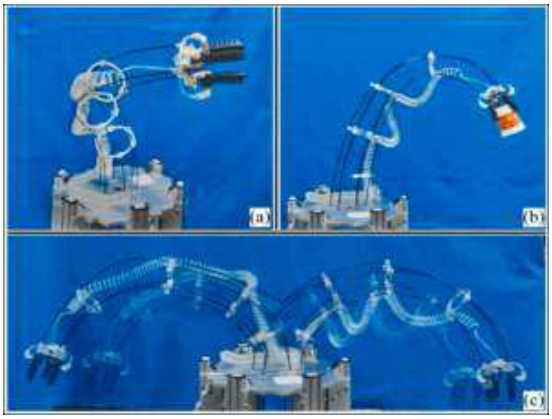


(a) Picture from [27]

(b) Picture from [28]



(c) Picture from [29]

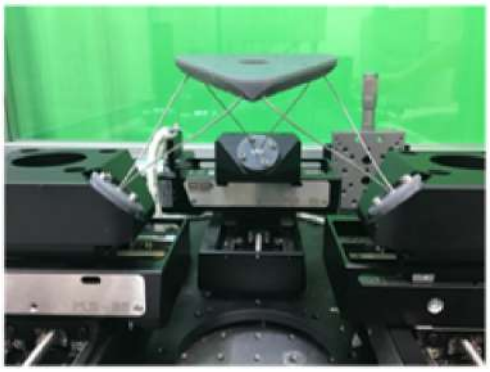


(d) Picture from [30]

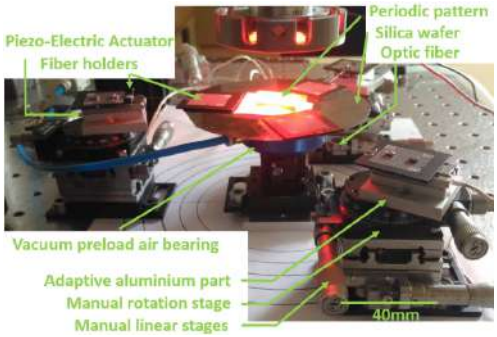
Figure 1.5: Illustration of the existing extensible *CPRs*. These *CPRs* employ intermediate disks.

[26] (Fig. 1.3b). Beams are actively rotated at the base, and *CPRs* instabilities are used to create artistic movements and sound effects. Then, the academic interest in *CPRs* increased after the work of Bryson and Rucker [13]. Various designs have been proposed, which differ by the actuation strategy, the beam arrangement, the prototype size, and the intended application. The following non-exhaustive list is dedicated to the description of the existing *CPR* prototypes.

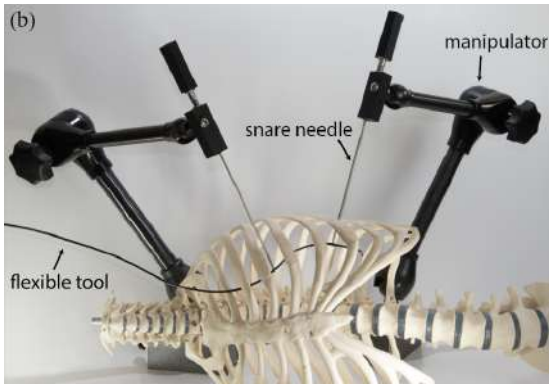
Extensible *CPRs*. The term extensible *CPRs* defines *CPRs* architectures that employ actuators varying the beam's active length, namely the portion of each leg that can undergo deformations inside the robot workspace, as the first *CPR* prototype of [13], illustrated in Fig. 1.4a. Extensible *CPRs* claim to attain a large workspace concerning both position and orientation abilities [16] (Fig. 1.4b), but the actuator size is relevant, and the required installation size is frequently comparable to the reachable *EE* workspace [16]. Moreover, the sliding nature of the beam's actuation makes installing sensors over the beams difficult, and frictional effects may increase the required actuation efforts [18]. Usually, the beam arrangement follows a Gough-Stewart-like topology [23] (Fig. 1.4c). Extensible *CPRs* have been proposed and tested for surgical tasks [16], haptics [24] (Fig. 1.4d), and for miniaturized-scale applications [15] as the glass-made prototype of Fig. 1.4e. Except for the prototype of [15], where a few design considerations are illustrated, the prototype design is never conducted following optimal design rules based on performance analysis.



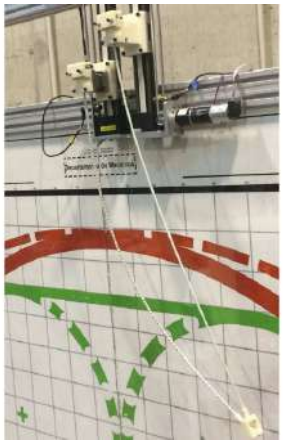
(a) Picture from [31]



(b) Picture from [32]



(c) Picture from [33]



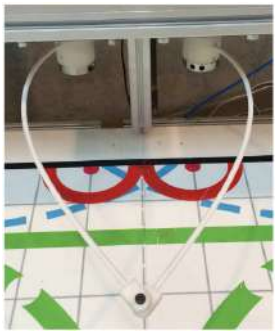
(d) Picture from [34]

Figure 1.6: *CPRs* with fixed beam length actuated at the robot's base, using prismatic motors

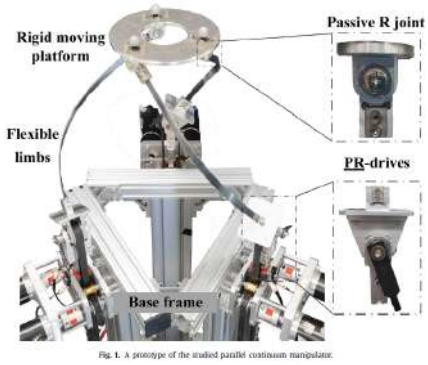
Intermediate constraints. As described in [27], using intermediate disks between the tip and base of the beam considerably increases the robot's orientation ability without compromising the translational capabilities in the case legs are extensible. Firstly proposed in [27] (Fig. 1.5a), the use of intermediate disk has been widely accepted as a design feature to increase the *CPR* performances at the cost of higher design complexity. Intermediate disks are used in [28] to constrain the *CPRs* passive beams (Fig. 1.5b). Moreover, the *CPRs* proposed in [29] and [30] employ an actively commanded intermediate disk to increase the robot's translational capabilities (Fig. 1.5c and 1.5d). Despite intuition led the researchers to this interesting and performance-enhancing feature, no design analysis or comparison with alternative solutions has been proposed.

Inextensible *CPRs* actuated at one beam's extremity. Extensible *CPRs*, which resemble traditional Gough-Stewart platforms, promise large workspaces. However, many *CPRs* prototypes are actuated differently using prismatic or rotative actuators at each beam's base, bringing complementary advantages, such as greater attainable velocities [17] and simpler mechanical design. The prototypes of [14], [31], [33] (illustrated in Figs. 1.6a, 1.6b, and 1.6c, respectively) use prismatic actuators to obtain the *CPRs* motion. Instead, rotative actuators are used in [34] and [17] for large-scale *CPRs* (Figs. 1.7b and 1.7a, respectively). Ultimately, prismatic-rotative drives (PR-drives) are tested in [35] for a spatial quasi-translational *CPRs*, and in

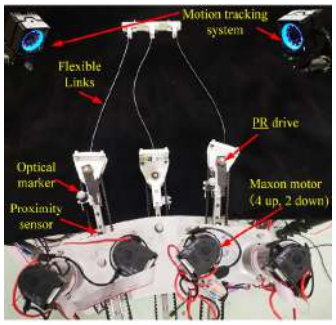
(a) Picture from [17]



(b) Picture from [34]



(c) Picture from [35]



(d) Picture from [36]

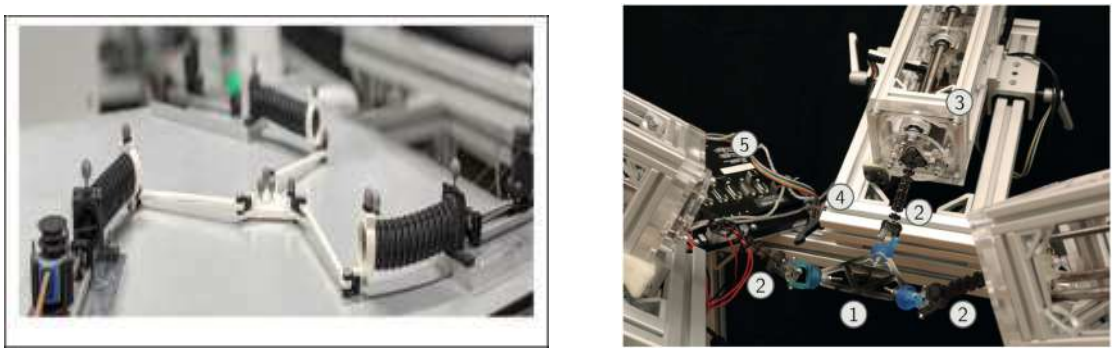
Figure 1.7: *CPRs* with fixed beam length actuated at the robot's base: a),b) use revolute actuations, while c),d) employ PR drives.

[36] for investigations on actuation redundancy (Figs. 1.7c and 1.7d respectively). As for the previous prototypes, also the inextensible *CPRs* design is never based on the optimization of some *CPR* performances.

Tendon-actuated *CPRs*. In the previously described *CPRs*, each beams deforms in a passive manner as a consequence of the actuation actions localized at the robots base. In contrast, tendon-actuated *CPRs* are made by assembling multiple tendon-driven *CRs* arranged in parallel and connected to a rigid *EE* [37]. Each link is actuated by cables coiled and uncoiled at the robots base. As the tendons are routed in disks placed over the flexible link, the actuation action is distributed all over the beams length, and each flexible link is actively deformed. A planar 3 *DoFs* tendon-driven *CPR* is presented in [38] with a focus on its modelling and calibration (Fig. 1.8a), and a reconfigurable tendon-driven spatial *CPR* is described in [39] (Fig. 1.8b). The work of [37] compares different designs in relation to the expected kinematic performances (manipulability).

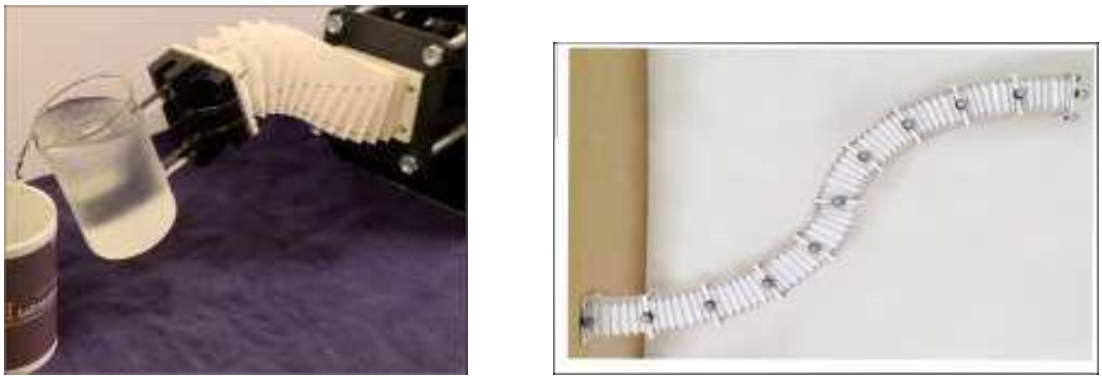
Origami structures. A recent trend in robotics proposes the use of origami structures as structural components to create robust and lightweight *CRs*. An origami structures connects the robots' base and the *EE*, while several beams are arranged in parallel inside the origami backbone. This design has been utilized in *CPRs* in the works of [40] and [41] (Figs. 1.9a, 1.9b, respectively), promising significant payload capabilities obtained without compromising the overall robot size.

CPRs received significant attention in the last decade: this is demonstrated by the significant number of different prototypes that emerged. However, by investigat-



(a) Picture from [38] (b) Picture from [39]

Figure 1.8: Tendon actuated *CPRs* designs



(a) Picture from [40] (b) Picture from [41]

Figure 1.9: Origami-based *CPRs* designs

ing the currently existing designs, few works (probably only [15], [37]) considered performance indices to drive the design selection. Although the experience and common sense rules may guide the design process, introducing performance-based analysis may critically improve *CPRs* design. However, as illustrated in the next section, the performance-analysis tools of *CPRs* are relatively immature and underdeveloped. This motivates further study in this direction. It is common to discuss the *CRs* modelling strategies after the design descriptions. However, Chapter 2 is dedicated to the model descriptions, and the focus of this Chapter is maintained on the performance quantification of *CRs*, which is the main topic of this thesis.

1.2 Performance Analysis

Performance indices are metrics designed to quantify the different performances of robot manipulators [42]. Performance indices help researchers and engineers to evaluate and compare different robot designs, their potential applications, and their limits. The literature on rigid-link robot's performance measurement is vast, and here the terminology of [42] is followed. Performance indices can be classified into local indices or global indices: the formers depend on the robot configuration, while the latters are associated with a specific robot design. The robot workspace volume or area is widely used as a global performance index, usually focusing on the reach-

able, translational or dexterous workspace[1]. Instead, the local performance analysis is usually associated with some metrics of the robot Jacobian matrix. In this Chapter, the term Jacobian matrix identifies the matrix relating the input velocities to the *EE* twist [44]. The dexterity analysis, that is, the capability to move the robot's end-effector in all directions with ease, is usually measured by evaluating the relative conditioning of the Jacobian matrix, or the value of its minimum singular value [45]. However, the dexterity measurement provides only directionality information, and does not consider the magnitude of the measured abilities. To overcome this issue, the manipulability index measures the determinant of the Jacobian matrix [46], quantifying the magnitude of the measured abilities. Still, also the determinant is inadequate to measure the closeness to the singularity of the Jacobian matrix [44][2]. Additional performances of interest are the force transmission ability, the robot's stiffness and accuracy [43].

The main difference between *CRs* and rigid-links robot is the intrinsic elastic nature of *CRs*. Moreover, *CRs* are intended for different tasks than rigid-link robots. Still, most of the literature related to the performance analysis of *CRs* is based on the readaptation of existing rigid-link indices. In the following, we classify existing *CRs* indices between local and global indices. Moreover, local indices between measurement based on the geometrico-static problem linearization, and indices that uses the geometrico-static problem solution only, are illustrated separately.

Local Indices based on the geometrico-static model linearization:

- **Dexterity.** In *CRs*, the dexterity is measured by considering the ratio between the minimum and maximum eigenvalue of the Jacobian matrix [47]. This dexterity measurement is used in [48] on a novel concentric tube robot, and the main disadvantages of using such a metric are illustrated in [47];

- **Manipulability.** as in rigid-links robots, a commonly used performance index in *CRs* is the manipulability, measured by the determinant of the Jacobian matrix. The manipulability ellipsoid was first formulated in [49] for *CRs* by adapting the existing approaches of rigid-links robot to the *CR* framework. Still, this approach is limited to the planar case and to the specific modelling assumptions presented in the paper. Manipulability is studied in [48] in relation to a novel concentric tube robot. *CPRs* manipulability is investigated in [37] on a 3-*DoFs* planar *CPRs* actuated by tendons, an in [15] on a spatial micro-scale *CPR* by computing the Jacobian matrix with a finite-differences approach.

- **Stiffness measurements.** As *CRs* are flexible by design, their stiffness plays a crucial role in their performance evaluation. The term stiffness matrix identifies the matrix relating the input actions (e.g. motor torques-forces) with the platform wrench. Since the *CRs* stiffness matrix is not equal to the transpose

[1] According to the terminology of [43], the reachable workspace is the set of all the positions that may be reached with at least one orientation of the platform. The translational workspace is the set of all possible locations that can be reached with a given orientation. The dextrous workspace is the set of all the locations for which all orientations are possible.

[2] As an example, consider the identity matrix of dimension three, whose determinant is unitary. However, multiplying the identify matrix by an arbitrarily small constant (e.g. 10^{-5}) drastically reduces the determinant, but still, the matrix is full rank.

of the Jacobian matrix [18], the *CRs* stiffness matrix is derived from the robot kinemato-static model in [35] or by ODEs integration in [50]. Passive elements compliance [51] or actuation redundancy [36] are considered in relation to the stiffness of the robot. Also, a global stiffness performance index is proposed in [52] to optimize the *CRs* stiffness in a defined direction.

- **Singularity.** Near singularities, robots lose some abilities [43]. Using the constant curvature assumption, singularities are investigated over the workspace of a 3-*DoFs* planar *CPR* actuated by tendons, obtaining similar results to the equivalent rigid-link parallel robot. Alternatively, using Plücker lines to investigate singularities was proposed for *CPRs* in [53], [54]. Instead, within the variable curvature framework, sufficient conditions for the forward singularities of serial *CRs* are formulated in [55], and a comprehensive singularity classification is proposed in [56] for *CPRs* by analyzing the second order derivatives of the robots' energy;

- **Equilibrium Stability.** The characterization of the equilibrium stability is a primary issue in *CRs*. The possible stable-to-unstable transitions that *CRs* may admit due to the intrinsic elasticity or load exchange with the environment complicates the use of *CRs* in tasks where safety and human-robot interactions are mandatory. Instability is a great concern in concentric tube robots [57] as well as in *CPRs* [58].

- **Other measurements.** Adapting the existing approaches of rigid-link robots to the *CR* framework creates intuitive performance indices. However, it should be considered that, due to the inherent compliance and elasticity of continuum robots, the common force-velocity duality does not govern their motion, and the optimal direction for exerting force is not necessarily the optimal direction for controlling velocity [59]. In this direction, the work of [60] proposes a unified force-velocity ellipsoid to consider simultaneously the effect of the Jacobian and the compliance matrix on the *CRs* power transmission. Alternatively, linearising the *CRs* geometrico-static modelling approaches provides useful matrices that correlate actuator and platform velocities, platform applied wrenches and actuation actions [18]. The resulting matrices are exploited to measure quantities specific to *CRs*, such as how a platform wrench is reflected on an actuation velocity. However, the robot performances in terms of velocity and force transmission are analyzed separately, and the instrinsic coupling between geometry and elasticty is ultimately neglected;

Local Indices based on the geometrico-static model solution

- **Orientability Analysis.** Orientability plays a crucial role, particularly in serial *CRs*, and the orientation ability is frequently measured by solving geometrico-static problems with different *EE* orientations until a limit of the orientation capability is reached (e.g. unsolvability of the geometrico-static problem, mechanical limits). An orientability index is computed in [61] as the ratio between the accessible and the full surface of a target sphere or as the ratio between the accessible and the full surface of a target anatomical volume [62]. Design optimization is also done in [63] by maximizing robot orientability. The main

difficulties in the quantification of the orientation abilities are due to the complexities of the workspace computation: how to explore the orientation space? How to efficiently compute the orientation ability boundaries?

Global indices:

- **Workspace**. As in rigid-link manipulators, the *CRs* workspace is also frequently used as a performance index. In particular, workspace measurements, such as the translational workspace size [64], have been widely used for design optimizations. Similarly, design optimization has been carried out by minimizing the difference between a target workspace and the *CR* workspace for tasks where the robot enter a confined space by a single location and large orientation abilities are required [65] or by considering the reachability of a target workspace by the robot [66]. Additionally, the workspace size is used as a performance metric by considering the reachability of a target volume in the presence of anatomical physical constraints [67]. Workspace-based indices are promising for *CPRs* performance analysis. However, the use of these indices is limited by the complexities of the workspace computation, that in *CPRs* is not trivial due to the geometrico-static nature of the *CPRs* analysis;

Most of the research effort has been directed toward the derivation of local indices based on the geometrico-static model linearization. Local indices based on the Jacobian matrix evaluation (dexterity, manipulability) or the stiffness matrix evaluation are straightforward. Indeed, these indices are simple to interpret because they are similar to rigid-link-robot indices. However, the elastic nature of *CRs* poses questions: is it adequate to measure the *CRs* performances by considering measurements of the Jacobian and the stiffness matrix independently and not to evaluate geometry and elasticity performances simoultaneously? Moreover, equilibrium stability assessment is fundamental for the use of *CPRs* in real tasks, but its modelling and quantification is still not deeply investigated. Finally, workspace computation algorithms are of great interest to the scientific community but are still problematic. These algorithms may play a crucial role in performance analysis.

Thus, the focus of this Chapter and, ultimately, the thesis contributions, is restricted from the general performance analysis to i) the workspace computation problem and ii) the equilibrium stability assessment. Accordingly, Section 1.3 details the current state of the art related to the workspace computation algorithms of *CRs*, while Sec. 1.4 illustrates the current approaches for the equilibrium stability assessment.

1.3 Workspace Computation: state of the art

Workspace evaluation, which can be briefly described as identifying all the configurations where the robot may rest in a stable equilibrium, is a well-known topic in robotics. However, the application of state-of-the-art techniques for rigid-link parallel robots [43] to *CRs* is precluded by the complexities related to their inherent flexible nature: their geometry is not sufficient to describe the pose of the whole robot since the deformation of several elastic links defines the configuration. For instance, rigid-link parallel robot approaches based on the geometry of the manipulators (known as geometric approaches [68]) cannot be used in *CPRs* since the robot

configuration also depends on the external forces that act on the system and the geometrico-static problems do not admit an analytical solution in general.

Workspace computation algorithms for *CRs* are commonly based on two fundamental tools: a workspace exploration strategy and the repeated solution of the geometrico-static problem. In the next Sections, some of the most popular approaches for the workspace evaluation of *CRs* are listed, discussing the employed exploration strategy, the modelling techniques employed, and the main advantages and drawbacks of each algorithm. Table 1.1 collects the state-of-the-art findings. The next section illustrated algorithms that may evaluate the full workspace of *CPRs* (Section 1.3.1) and algorithms designed to find the boundaries of the *CPR* workspace only (Section 1.3.2).

1.3.1 Full Workspace Computation Algorithms

Full workspace computation algorithms evaluate the *CPR* workspace by calculating each robot configuration that lies in the workspace. Two popular approaches can be found in the literature: i) actuation space sampling techniques and ii) task space sampling techniques.

Actuation space sampling techniques are generally based on the discretization of the *CPRs* actuation space and the iterative solution of the forward geometrico-static-problem (*FGSP*). These approaches bring general applicability and reduced complexity as advantages while their computational time can be significantly high since it increases with the sampling density and the number of actuators [69]. Actuation-space sampling approaches have been used in [70] for the workspace computation of a pneumatically actuated *CRs*, and then employed in [71] for the workspace computation of concentric tube *CRs* (*CTRs*), and also in [72] for a 6-*DoFs* *CPR*. Several sampling strategies are proposed (with uniform or random distributions), but the computational time is considerably high if good accuracy is required or the number of actuators increases. Moreover, strain limits are considered only after the workspace computation. This leads to a considerable increase in the overall time due to the computation of unreachable (and unnecessary) points. To mitigate these drawbacks, an approximation of the *CPRs* workspace was obtained by sampling the actuation space of each leg, computing its workspace, and then obtaining the robot workspace as the intersection of the leg's workspaces. However, the wrench exchanged between each leg and the end effector cannot be considered, and this may be a strong source of inaccuracy in the obtained results. This technique was applied for the workspace computation of a *CPR* made by three compact bionic manipulators [73] by uniformly sampling each leg actuator space and then intersecting the three resulting workspaces. A similar approach, with the same advantages and drawbacks, was proposed in [38] for the workspace computation of a *CPR* made by three independently-actuated tendon-driven *CRs*.

Reference	Group	Strategy	Model	Case Study	Comp. Cost
Trivedi et al. 2010 [70]	Full Workspace	Actuation sampling	[70]	OctArm VI *SCR*	5
Burgner et al. 2014 [71]	Full Workspace	Actuation sampling	[74]	concentric tube *SCR*	5
Orekhov et al. 2016 [72]	Full Workspace	Actuation sampling	[75]	6 -*DoFs CPR*	5
Singh et al. 2017 [73]	Full Workspace	Actuation sampling + intersection	[76]	3 CBHA connected in parallel	4
Nuelle et al. 2020 [38]	Full Workspace	Actuation sampling + intersection	[7]	3 tendon driven *SCRs* connected in parallel	4
Amehri et al. 2020 [69]	Full Workspace	Task-space sampling + small deformation neighborhood	[69]	tendon driven *SCR*	3
Mauze et al. 2020 [14]	Full Workspace	Task-space sampling of [77]	[7]	planar 3-*DoFs CPR*	3
Altuzarra et al. 2019 [78]	Full Workspace	Task-space sampling	[79]	planar *CPRs*	3
Lilge et al. [37]	Full Workspace	Task-space sampling	[7]	3-*DoFs CPR* and 3 tendon driven *SCRs* in parallel	3
Briot, Goldsztejn 2021 [56]	Full Workspace	Task-space sampling flooding	[80]	*CPRs*	3
Wu et al. 2022 [30]	Full Workspace	Task-space sampling trajectories	[27]	6-*DoFs CPR*	2
Wang et al. 2022 [81]	Boundaries	Closed-form	[81]	2 segment *SCR*	1
Amehri et al. 2021 [82]	Boundaries	Continuation	[83]	tendon-driven *SCR*	1
Amehri et al. 2021 [84]	Boundaries	Optimization	[85]	tendon-driven *SCR*	1
Amehri et al. 2022. [86]	Boundaries	Optimization	[69]	tendon-driven *SCR*	1

Table 1.1: Workspace computation approaches for *CRs*. The computational cost is expressed in a range between 1 (low) and 5 (high).

On the other side, **task space**[3] **sampling techniques** are based on the discretization of the *CPRs* task space and the iterative solution of the inverse geometrico-static problem (*IGSP*). This way, the computational time can be reduced with respect to (w.r.t.) actuation-sampling strategies since the overall time depends only on the sampling density but not on the number of actuators. This was shown in [69], where the proposed approach considerably reduced the computational time w.r.t. actuation-sampling techniques in the case of a soft manipulator driven by three cables. Preliminary results on the workspace evaluation of a 2-*DoFs CPR* were also shown in [78]. Still, the generality of their approach is reduced by the modelling strategy strictly limited to planar cases. Another task space sampling approach was employed in [14] for the workspace computation of a 3-*DoFs* planar *CPR*: this approach ensures generality and the inclusion of mechanical limits, but not singularity and equilibrium-stability analyses. Using a simplified mathematical modelling approach, singularity analysis was performed in [37] during workspace computation. Still, the employed modelling technique is simplified w.r.t. the real physics of the problem, and it may be inaccurate in terms of pose prediction. A flooding algorithm was proposed in [56] for the workspace computation of general planar *CPRs*, singularity identification, and equilibrium stability assessment. Flooding approaches are algorithms based on a fixed grid exploration: the direction on which the grid is explored is not fixed, and it depends on a neighborhood analysis performed at each step. Consequently, flooding algorithms provide general tools for the computation of complex workspace volumes. However, the computational time significantly increases when the desired accuracy is high. Finally, a task space exploration strategy has recently been proposed in [30], with the workspace computation algorithm being based on the generation of several trajectories on the task space and the iterative solution of the *IGSP* over these trajectories. This approach has shown to be computationally efficient, but the identification of singularity loci has not been considered.

In summary, full workspace computation algorithms are applicable for general *CPRs*. Computational time may be high (hours or even days), particularly if an actuation sampling strategy is employed. Even if most of the workspace exploration strategies are not dependent on the *CPR* modelling strategy, the latter determines the capability to evaluate several features, such as singularities and equilibrium stability. Moreover, the analysis of pose occurrences within the workspace is not trivial when employing task-space sampling techniques, while actuation sampling techniques do not suffer of this issue. The same problem is valid when analyzing possible robot redundancy.

1.3.2 Boundary Workspace Computation Algorithms

Because the computational time of full workspace computation algorithms is significant, workspace computation algorithms that identify only the boundaries are often used. In this Section, the analysis is limited to boundary-computation strategies based on i) the identification of closed-form analytical solutions, ii) continuation approaches, and iii) optimization approaches.

[3]With the term task space, we denote a subspace of the robot configuration space. This space is usually defined by some variables of the robot end-effector pose, such as its position or its orientation.

The closed-form equations of the workspace boundary may often be identified for rigid-link robots [87]. However, the complexity of the *CPR* modelling techniques makes such an identification particularly challenging, and exact solutions were found only in a few special cases. A relevant example in this direction is provided in [81], where the analytical condition for border detection is based on singularity conditions, unfortunately the method was limited to a specific serial *CR* design (a *CR* with two indipendently actuated segments).

Continuation algorithms [88],[82] provide common and computationally efficient tools. Still, they are generally limited to planar cases and three-dimensional workspaces are obtained only by superimposition of several planar slices. Additionally, inequality constraints (e.g. related to strain limits or stability assessment) are hard to include, and the identification of voids or holes in the workspace is tricky.

Finally, **optimization approaches** are a class of powerful tools for the computation of workspace boundaries, firstly introduced in [89] for rigid-link robots. These approaches find boundary location by solving an optimization problem, where the optimization target is the distance between the *EE* position and a given point to be reached: inequality and equality constraints can be included, leading to general formulations. However, if the robot admits multiple solution for the same geometrico-static problem considered in the optimization process, the numerical optimization may be directed toward a different working mode, leading to incorrect predictions of the workspace; the same as for continuation methods, where voids are challenging to individuate. Optimization algorithms for *CRs* were proposed with different modelling strategies in [86], [84]. In summary, boundary workspace computation algorithms offer better computational performances than full workspace algorithms. As a drawback, these algorithms are strongly influenced by the selected modelling strategy and computational methodology, which affects not only the performance of the algorithm but also their applicability.

In conclusion, in the author's opinion there is a significant margin for improvement concerning the state of the art. Concerning the full workspace computation, the state of the art is mainly directed toward actuation sampling techniques, which are generally time-consuming if a large number of workspace candidates are to be tested, the latter increasing with the number of actuators employed. The few works related to task-space sampling are preliminary and frequently lack generality. The workspace boundary computation is even more challenging since no work has been directed in this direction for *CPRs* at the current date. Few works have been directed in this direction for *CRs*, with each approach's relative advantages and disadvantages, such as the inability to detect holes in the workspace, the difficult application to three-dimensional volumes, or inability to generalize to *CPRs* models.

1.4 Equilibrium Stability Assessment: state of the art

A fundamental problem in both serial and parallel *CR* design and analysis is the assesment of equilibrium stability: stable-to-unstable configuration transitions are possible due to the elastic structure of *CRs*. For instance, *CTRs* exhibit instabilities with significant impact on the usability and controllability of the robot [57]. As the tubes rotate and translate w.r.t. each other, elastic potential energy accumulates

until an unstable configuration is met, and the energy is released with a dangerous snapping [90]. To avoid instability, the tube curvature can be optimized [91], but design anisotropies can also be introduced [92]. Alternatively, [93] discusses design rules which aim at ensuring the absence of instabilities. In the same fashion, *CPR* designs displayed stable-to-unstable transition [58], which limits the *CPR* motion abilities and their potential intrinsic safety. It should be mentioned that stability assessment is a relevant problem not only in robotics: the Euler's buckling load [94] established the foundations for the elastic stability assessment of rigid beam structures. Successive works focused on different aspects of equilibrium stability, such as secondary bifurcations [95], post-buckling instabilities [96], and investigation of stability bifurcations [97].

1.4.1 Equilibrium Stability Characterization

Energetic considerations are necessary to characterise equilibrium stability because stable *CR* configurations are associated with a minimum of the total *CR* potential energy [74], [56]. Two prevalent approaches have been identified in the literature: optimal control approaches and the analysis of the Hessian matrix of the *CR* energy.

When continuous (not discrete) *CRs* modelling approaches are used (e.g. [18], [78]), equilibrium stability is frequently studied using **optimal control approaches** (*OC*) [57]. These approaches derive stability conditions through non-discretized *CR* equations, and the resulting numerical test, based on the integration of differential equations, determines the robot's stability. *OC* approaches provide a rigorous approach minimally affected by discretization issues at the cost of increased mathematical complexity. State-of-the-art methods on *CTRs* determine the equilibrium stability by the use of *OC* approaches [98], [57], and *OC* is also used in [58] to show that *CPRs* admits stable-to-unstable transitions. Equilibrium stability of planar *CPRs* also received significant attention, and *OC* theory is used preliminary in [99], and on a family of three-actuated-*DoFs* planar *CPRs* in [100]. Indeed, other robotic systems benefit from *OC* approaches for the equilibrium stability assessment. Sagging cables of cable-driven parallel robots are analogous to long and slender flexible beams frequently modelled with Irvine's model (a particular subcase of the Cosserat beam's model) and *OC* can be used to assess the equilibrium stability of sagging cable-driven parallel robot [101].

Alternatively, when the *CRs* configuration is described by a finite number of variables (e.g. when using discretized *CRs* equations), equilibrium stability can be characterized by evaluating the positive definiteness of the **Hessian matrix** of the potential energy [56]. In contrast to *OC* approaches, stability assessment based on the Hessian matrix provides intuitive mathematical derivation and simplicity, but the accuracy depends on the number of discretization coordinates. The approach proposed in [102] established a numerical method (based on the reduced Hessian matrix evaluation) to assess the *CTRs* stability, follow stable paths by continuation, and determine bifurcations where instabilities occur. Equilibrium stability assessment plays a crucial role in the workspace evaluation of *CRs*, and unstable configurations define the attainable workspace boundaries. During the workspace computation, the equilibrium stability of each configuration is to be verified, and the positive definiteness of the reduced Hessian matrix is a straightforward and effective strategy that avoids differential equation integrations proper of *OC* approaches.

The same approach is used for different problems, such as equilibrium stability assessment of underconstrained cable-driven robots [103] (assuming cables as straight) or stable equilibrium continuation of elastic beams [97].

1.4.2 Stability metrics

Stability metrics defining a distance to instability are essential in many cases, such as when planning stable *CRs* trajectories [21], or when an external load is to be applied to the *CR* [57]. However, stability metrics were rarely investigated, and only a few works were proposed in this direction. Based on the *OC* framework, an index for the equilibrium stability measurement of *CTRs* is given in [98]. Except for the two-tubes case, the proposed index is based on the use of a matrix determinant which cannot meaningfully indicate the closeness to non-optimality in general [44]. One may use the condition number as an alternative to the matrix determinant to solve this issue. However, when the matrix is made of non-homogeneous units, the condition number may fail in indicating the closeness to rank deficiency. The work of [104] proposed to measure the equilibrium stability of *CTRs* from a different perspective. As *S-curves* describes the input-output relation of a two-tubes *CTR*, the observation that stable *CTRs S-curve* do not exhibit a negative slope suggests using the *S-curve* slope as stability metric. However, this approach is limited to the two-tubes case. In the case of *CPRs*, a heuristic index is proposed in [75] within the *OC* framework. The proposed index is based on an equivalent integration length to make appear the so-called conjugate points where instability should occur. The proposed index is heuristic, and its use is limited by the fact that there is no straghtforward proof that the mechanism reaches the limit of stability when the conjugate points appears at the abscissa zero. Additionally, the sensitivity of this metric to small changes in other model parameters could be high, and the index should be used with caution [58].

The equilibrium-stability analysis of *CPRs* is a crucial topic for their real application since stable-to-unstable transitions need to be avoided for practical tasks. The work of [58] was the first to investigate the equilibrium stability assessment of *CPRs*, and rigorous results are obtained with the *OC* approach also in serial *CRs* [98]. However, the alternative approach of the Hessian matrix evaluation is competitive for its more straightforward and effective applicability; a deep experimental investigation of its accuracy is currently missing. Moreover, investigating the causes of the instability of *CPRs*, and the underlying physical phenomena happening when the *CPR* equilibrium becomes unstable is of interest. Additionally, a few works proposed distance to instability metrics, but the existing indices frequently lack of physical meaning.

1.5 Summary of the state of the art and associated perspectives

CPRs are a recently proposed class of manipulators with many open research directions. *CPRs* modelling [18], design [105] and control [20] requires further investigations, and this thesis focuses on the performance analysis of *CPRs*.

Performance analysis of *CRs* (and of *CPRs*) is frequently carried out by re-adapting existing indices proper of rigid-links robots. Most of the performance investigation focuses on the *CRs* Jacobian matrix or stiffness matrix separately, and the intrinsic coupling between geometry and elasticity of *CRs* is frequently neglected. In particular, two major limitations in the state of the art are identified, that is, i) efficient workspace computation strategies for *CRs* and ii) techniques for the assessment and measurement of the equilibrium stability.

The workspace computation problem of *CPRs* plays a crucial role in the performance evaluation of *CPRs*, since the *CPRs* workspace may be used to get useful metrics such as the workspace volume, and it could be used as an objective for optimal design tools. Full workspace computation is mainly performed by using actuation sampling techniques at the cost of high computational time. On the other hand, task-space sampling is still at a preliminary level of development. Workspace boundary computation of *CPRs* is even more challenging. Despite some technique for serial *CRs* exists, no work is specifically devoted to the boundary workspace computation of *CPRs*. As shown in Sec. 1.3.2, few works have been directed in this direction for *CRs*, with relative advantages and disadvantages that each approach brings, such as the inability to detect holes in the workspace and the difficult application to three-dimensional volumes.

The second topic of interest is the equilibrium stability analysis. *OC* approaches are mostly used in the state of the art. In the author's opinion, this is mainly due to the popularity of continuous modelling approaches (discussed in the next chapter) that fit the *OC* formalism. However, the alternative approach of the Hessian matrix evaluation is competitive for its effectiveness and simplicity of derivation once the discretized robot model is derived. The major shortcoming of the state of the art is the almost complete absence of equilibrium stability indices. Metrics to measure the distance to instability are essential, but the few existing works lack a physical meaning of the measure.

In this thesis, the following topics are explored and contributions are proposed in these directions:

- The workspace full exploration. New full workspace computation algorithms that bring generality in terms of algorithm applicability to different *CRs* architecture are necessary, and an improvement of the computational performances w.r.t. the state-of-the-art approaches is desired. Additionally, it is investigated how to efficiently certify the numerical results in terms of the existence and uniqueness, and continuity of the numerical solutions at each workspace point, in contrast to state-of-the-art time-consuming approaches. Chapter 3 focuses on these aspects;

- The workspace boundaries computation. The goal is to develop a new boundary computation algorithm for *CPRs*. The focus is directed on the general applicability of the algorithm and on the possibility of detecting holes in the *CPR* workspace, in constrast to existing approaches that may fail in these directions. Chapter 4 describes these contributions;

- The experimental equilibrium assessment on a new *CPR* prototype (chapter 5). It is of interest to verify the accuracy of the equilibrium stability assessment based on the Hessian matrix evaluation and to better understand the physical phenomena of the stable-to-unstable transitions;

- The distance to instability measurement. The scope is to propose a new index to measure the distance of a *CR* configuration from the instability (Chapter 6). The scope is to obtain an index with a the physical meaning of distance measurement, rather than existing metrics lacking in physical meaning of the results.

The next chapter concludes the state-of-the-art analysis, as it will discuss the modelling techniques for *CRs*. Despite the thesis focuses on performance analysis, it is fundamental to describe the modelling state of the art and the employed modelling strategies to better understand the contributions of this thesis.

Chapter 2

Geometrico-Static Modelling of Continuum Parallel Robots

This Chapter focuses on the geometrico-static modelling of *CPRs*. Although the topic of this thesis is the performance analysis of *CPRs*, the choice of the *CPRs* modelling techniques plays a crucial role in robot analysis. Selecting the appropriate *CPRs* modelling approach for the robot analysis among the vast literature [106] is a complex task. Section 2.1 presents a concise state-of-the-art analysis of *CRs* modelling strategies to illustrate the existing possibilities. Then, Section 2.2 describes the energy-based discretized modelling technique used in this thesis.

2.1 Brief State of the art on Continuum Robot Modelling

The geometrico-static modelling of *CRs* received great attention from the scientific community, and the literature is vast [106]. Most of the currently employed *CRs* modelling approaches focus on largely deformable components, and the *CRs* flexible components are frequently represented as slender beams. Beam-like structures historically received great interest from the community: the early works of Bernoulli and Euler, Kirchhoff [107], Love [108], Cosserat [109], and Reissner [110] can be considered as the origin of beam's modelling strategies that undergo large displacements and rotations. However, the application of three-dimensional elasticity results in complex beam equations. The resulting mathematical models are computationally expensive, and, usually, assumptions are introduced to obtain a simplified modelling approach that can be adequate for practical applications [111]. In the following Sections, the frequently introduced assumptions that transform the three-dimensional beam problem into a simplified numerical problem are illustrated.

2.1.1 Assumptions

Classical small-deformation theories assume that the deformed beam configuration is close enough to the undeformed one. However, *CRs* achieve motion through the controlled large displacement of some elastic beams, and it is not possible to rely on the small-deflection assumption, in general. Several simplifications need to be introduced to transform the three-dimensional elasticity beams problem into a math-

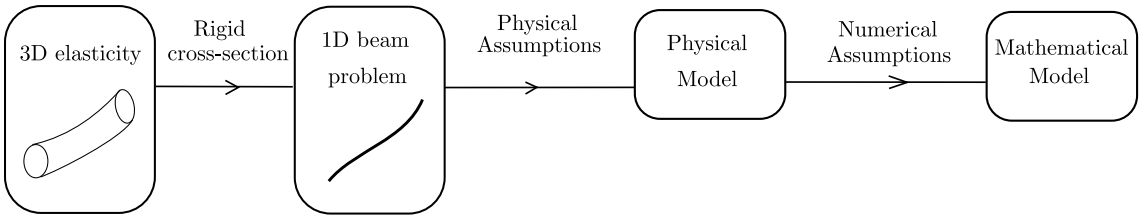


Figure 2.1: The different levels of assumptions that transforms the three-dimensional beams' elasticity into a simplified mathematical problem.

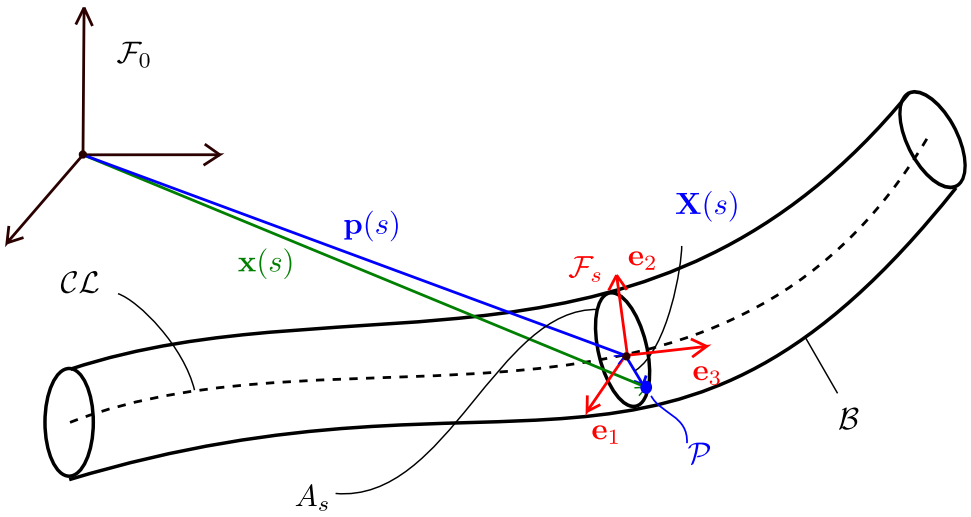


Figure 2.2: A slender beam undergoing large deformations.

ematically solvable problem. Figure 2.1 resumes the different levels of assumptions [19]:

- the three-dimensional beams elasticity problem is converted into a mono-dimensional beam problem introducing the rigid cross-section assumption;
- the mono-dimensional beams elasticity is eventually simplified by introducing physical assumptions that restrict the possible strains acting on the beam. The resulting model is named physical model;
- the physical model is further simplified by using numerical approaches to obtain a mathematically solvable model.

Each assumption level is detailed in the following.

First, let us focus on the beam representation of Fig. 2.2. The beam is considered as a three-dimensional continuum body $\mathcal{B}$, and a one-dimensional domain $\mathcal{CL}$ represents the beam's centerline. $\mathcal{CL}$ is parametrized with $s \in [0; L]$, and L is the length of the beam at the reference configurations. For each value of s, a two-dimensional space $\mathcal{A}_s$ called cross-section is considered, which is a subset of $\mathcal{B}$, and which intersects $\mathcal{CL}$ in the cross-sections centre of gravity. In particular, $\mathcal{B}$ is assumed to be a the disjoint union of all the cross-sections. At each s a floating frame $\mathcal{F}_s$, called local frame, is attached with the origin placed at the centroid of the cross-section. A frame $\mathcal{F}_0$ is considered an inertial fixed frame. In this framework, the rigid cross-section assumptions is introduced [111]: each cross-section is assumed to remain rigid, and thus it can only rotate and translate as a rigid body.

This assumption implies that no change in the shape or size of the cross-section is allowed. Moreover, let us consider:

- A set of unitary vectors $\mathbf{e}_i,\ i = 1,2,3$ associated to $\mathcal{F}_s$, whose values are expressed w.r.t. the local frame, namely $\mathbf{e}_1 = [1,0,0], \mathbf{e}_2 = [0,1,0], \mathbf{e}_3 = [0,0,1]$. In particular, $\mathbf{e}_3$ is assumed to be orthogonal to $\mathcal{A}_s$;
- The vector $\mathbf{X}_s(t_1,t_2) = t_1\mathbf{e}_1 + t_2\mathbf{e}_2 \in \mathbb{R}^3$ that defines the position of a point $\mathcal{P}$ over $\mathcal{A}_s$ w.r.t. the local frame.
- $\mathbf{p}(s) \in \mathbb{R}^3$, that is the position of $\mathcal{F}_s$ w.r.t. $\mathcal{F}_0$;
- $\mathbf{R}(s) \in SO(3)$, that is the matrix expressing the orientation of $\mathcal{F}_s$ w.r.t. $\mathcal{F}_0$.

Under these assumptions, the position of any point $\mathcal{P}$ w.r.t. $\mathcal{F}_0$ can be obtained as (see Fig. 2.2):

$$\mathbf{x}(s,t_1,t_2) = \mathbf{p}(s) + \mathbf{R}(s)\mathbf{X}_s(t_1,t_2) \tag{2.1}$$

Thus, the set of the cross-section poses $\mathcal{C}$ can be defined as:

$$\mathcal{C} = \left\{ \mathbf{G}(s) = \begin{bmatrix} \mathbf{R}(s) & \mathbf{p}(s) \\ \mathbf{0} & 1 \end{bmatrix} \mid \mathbf{G} : [0;L] \to SE(3) \right\} \tag{2.2}$$

which is a non-linear differentiable manifold [112]. Equation (2.1) shows that, in order to recover the position of each point of the beam, it is sufficient to know the manifold $\mathcal{C}$: the three-dimensional beams elasticity problem is simplified into a mono-dimensional problem in the variable s. Since the configuration of the beam is defined by the set of all the matrices $\mathbf{G} \in SE(3)$ from 0 to L, the space variation of $\mathbf{G}$ is governed by a twist field $\boldsymbol{\xi} \in \mathbb{R}^6$, defined by:

$$\widehat{\boldsymbol{\xi}}(s) = \mathbf{G}^{-1}(s)\mathbf{G}'(s) \tag{2.3}$$

with $(.)' = \frac{d}{ds}$, the vector $\boldsymbol{\xi}$, whose components are expressed in $\mathcal{F}_s$, is structured as $\boldsymbol{\xi} = [\mathbf{u}, \mathbf{v}]$, and $\widehat{\boldsymbol{\xi}} \in se(3)$ is defined as:

$$\widehat{\boldsymbol{\xi}}(s) = \begin{bmatrix} \widehat{\mathbf{u}}(s) & \mathbf{v}(s) \\ \mathbf{0}_{1\times 3} & 0 \end{bmatrix} \tag{2.4}$$

where $\widehat{\mathbf{u}} \in so(3)$ is the skew-symmetric representation of $\mathbf{u}$. Moreover, $\boldsymbol{\xi}$ is a strain measure of the beam [19], $\mathbf{u} \in \mathbb{R}^3$ represents bending and torsion of the beam, and $\mathbf{v} \in \mathbb{R}^3$ describes shear and extensibility. Since the director $\mathbf{e}_3$ is considered as orthogonal to the cross-section, v_1, v_2 represent the shear, and v_3 the extensibility, u_1, u_2 represents the bending, and u_3 the torsion. Assuming material properties as elastic, linear, isotropic, and constant over the length of the beam, the beam constitutive law, namely the relation between the beams stresses and strains, is written as:

$$\boldsymbol{\Gamma}(s) = \mathbf{K}(\boldsymbol{\xi}(s) - \boldsymbol{\xi}^*(s)) \tag{2.5}$$

where $\boldsymbol{\Gamma} \in \mathbb{R}^6$ is the beam internal stress whose components are expressed in the local frame $\mathcal{F}_s$, $\boldsymbol{\xi}^* \in \mathbb{R}^6$ denotes the undeformed strain configuration and the matrix $\mathbf{K} \in \mathbb{R}^{6\times 6}$ is structured as:

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_{BT} & \mathbf{0}_{3\times 3} \\ \mathbf{0}_{3\times 3} & \mathbf{K}_{SE} \end{bmatrix} \tag{2.6}$$

Name	Cosserat	Kirchoff	Kirchhoff Inextensible	Untorsionable Inextensible Kirchhoff
Strain	$\boldsymbol{\xi} = \begin{bmatrix} u_x \\ u_y \\ u_z \\ v_x \\ v_y \\ v_z \end{bmatrix}$	$\boldsymbol{\xi} = \begin{bmatrix} u_x \\ u_y \\ u_z \\ 0 \\ 0 \\ v_z \end{bmatrix}$	$\boldsymbol{\xi} = \begin{bmatrix} u_x \\ u_y \\ u_z \\ 0 \\ 0 \\ 1 \end{bmatrix}$	$\boldsymbol{\xi} = \begin{bmatrix} u_x \\ u_y \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$

Table 2.1: Summary of the most frequent allowed-strain models. For each case, $\boldsymbol{\xi}$ is written by imposing to zero the non-allowed components.

Matrices $\mathbf{K}_{BT}, \mathbf{K}_{SE} \in \mathbb{R}^{3\times3}$ are diagonal and frequently $\mathbf{K}_{BT} = \text{diag}(EI_x, EI_y, GI_z)$, $\mathbf{K}_{SE} = \text{diag}(EA, EA, GA)$, E is the Young modulus, G is the shear modulus, and I_x, I_y, I_z are the principal inertia moments of the cross-section, and A is the cross-section area. Then, let us consider the beam equilibrium. A distributed wrench $\mathbf{w}_d \in \mathbb{R}^6$, whose components are expressed in the **local** frame $\mathcal{F}_s$, acts on the beam, and the beams equilibrium in $\mathcal{F}_s$ is stated as [22]:

$$\boldsymbol{\Gamma}'(s) = \mathbf{ad}_{\boldsymbol{\xi}}^T(s)\boldsymbol{\Gamma}(s) - \mathbf{w}_d(s) \tag{2.7}$$

where the matrix $\mathbf{ad}_{\boldsymbol{\xi}} \in \mathbb{R}^{6\times6}$ is structured as:

$$\mathbf{ad}_{\boldsymbol{\xi}}(s) = \begin{bmatrix} \widehat{\mathbf{u}}(s) & \widehat{\mathbf{v}}(s) \\ \mathbf{0}(s)(s) & \widehat{\mathbf{u}}(s) \end{bmatrix} \tag{2.8}$$

By collecting Eq. (2.7), Eq. (2.6), and by rearranging Eq. (2.3) the algebraic-differential equations that govern the beams pose are obtained as follows[1]:

$$\begin{cases} \mathbf{G}'(s) = \mathbf{G}(s)\widehat{\boldsymbol{\xi}}(s) \\ \boldsymbol{\Gamma}'(s) = \mathbf{ad}_{\boldsymbol{\xi}}^T(s)\boldsymbol{\Gamma}(s) - \mathbf{w}_d \\ \boldsymbol{\Gamma}(s) = \mathbf{K}(\boldsymbol{\xi}(s) - \boldsymbol{\xi}^*(s)) \end{cases} \tag{2.9}$$

Thus, to find a beam configuration, it is necessary to find a set $\mathcal{C}$ that satisfies Eq. (2.9) and given boundary values of $\mathbf{G}, \boldsymbol{\Gamma}$.

Depending on the beam's properties, it is legitimate to neglect some components of $\boldsymbol{\xi}$. In this thesis, **physical assumptions** are defined as the simplifications introduced to approximate $\boldsymbol{\xi}$, and Table 2.1 collects the most frequent simplifications:

- **Full Cosserat**. No simplification is introduced in $\boldsymbol{\xi}$. This model is required for medium to large cross-sections, where the shear is important (e.g. silicon made *CRs*);

- **Kirchoff**. Neglecting the shear components of $\boldsymbol{\xi}$ usually holds when the cross-section dimension is small, but still extensibility is present (e.g. cables);

[1] These equations are also known as the strong form of the Cosserat beams equilibrium conditions [111]

- **Kirchhoff inextensible**. A Kirchhoff beam model neglects shear and extensibility of $\boldsymbol{\xi}$. This assumption holds for long and slender beams that are inextensible (e.g. NiTinol, fibreglass beams);

- **Untorsionable inextensible Kirchhoff**. In addition to the shear and extensibility, also the torsion may be neglected in some cases (e.g. when the torsional stiffness is considerably greater than the flexural stiffness).

Physical assumptions may reduce the model's complexity, and the resulting model is named **physical model** in this thesis. In particular, only inextensible Kirchhoff beams are considered in this manuscript, and thus shear and extensibility are considered to be negligible. Since the goal is to model *CPRs* with small cross-sections where extensibility is not important, it seems to be adequate to neglect shear and extensibility. Still, the integration of Eq. (2.9) in the $SE(3)$ group is not trivial, and numerical approaches are further introduced to find beam solutions. Some of the possible approaches are illustrated in the next Section, and the interested reader is addressed to specialized review papers for a more comprehensive overview (e.g. [106])

2.1.2 Literature

In this Section, some of the currently employed **numerical approaches** used to approximate the *CRs* physical model of Eq. (2.9) are outlined. In this thesis, three groups of approaches are described: continuous approaches, physically-approximated approaches, and discretized approaches

Continous Approaches: these techniques do not simplify Eq. (2.9) to find its solutions.

- **The shooting method** is widely used to obtain accurate solutions in *CRs* applications. This approach does not introduce further assumptions than physical models, and it provides significant accuracy and computational efficiency [74]. *CPRs* massively employ the shooting method to find solutions of Eq. (2.9), perhaps because of their variable curvature by design, and [13],[18] introduce the use of shooting methods for *CPRs*. A multiple shooting is proposed in [27] to simulate *CPRs* with intermediate disks and intermediate loads. However, when using multiple shooting, the model's complexity considerably increases. Computational efficiency is improved with an ad-hoc procedure for the computation of the shooting Jacobian matrix [75]. The accuracy of the shooting method is tested on a planar *CPR* in [14], and in tendon-driven parallel continuum robots in [113]. Continuous approaches based on the shooting method are accurate and computationally efficient, but the robot performance analysis is not immediate when using these approaches (as discussed in Sec. 2.1.3);

- **Elliptic Integral** solutions of the beam's physical models are available in a few cases. Assuming planar displacements only, shear, extensibility to be negligible, no external distributed loads, and only tip loads, the elliptic integral solutions are available to describe the configuration of the beam without any further computational effort [114]. The elliptic integral solutions have been

identified for the kinematic analysis of planar *CPRs* [115], for the solution of the forward and inverse kinematostatic problem of *CPRs* [116],[78], for preliminary investigation on stability and singularity analysis [117],[99], shape modelling of panel-made *CPRs* [118], and for a certified solution of the *CPRs* static problem in combination with interval analysis [79]. Elliptic integral solutions provide accuracy and computational efficiency as an advantage, but their application is extremely limited due to the considerable amount of assumptions to be satisfied (planar motions only and loads applied only at the rod's tip);

Despite the fact that continuous approaches use Eq. (2.9) with no additional approximation, there exists no analytical solution in the general case and numerical integration approaches must be employed to obtain an approximated solution to the exact one. Alternatively, additional simplifications can be introduced to further simplify the modelling approach.

Physically-approximated Approaches: these techniques introduces further assumptions to get simplified *CRs* models. However, physically-approximated approaches simplifies the *CRs* physical representation, introducing assumptions that limit the possible cross-section motion and/or the allowed motions of the *CR*.

- **The constant curvature** assumption has been successfully applied to many *CRs* applications: the overall beam curvature is assumed to be constant (or made by several constant-curvature segments), and thus the beam cross-sections are allowed to move accordingly. In the review work of [7], the main features of constant curvature models are recalled, and the simplified representation is the paramount advantage [119]. Constant curvature approximation achieve good results when a constant moment is applied to the *EE* [120], and this perfectly fits tendon-driven *CRs* [121]. However, if the *CR* does not fit the constant curvature assumption, the accuracy of this approach may be limited. Constant curvature approximation has been successfully employed in *CPRs* for modelling and workspace enhancement [38], and for the comparison of various designs [37]. In conclusion, constant curvature approximations provide simplified models as an advantage at the price of a decreased accuracy;

- **Lumped parameter models** for *CRs* represent the natural transition of traditional rigid-link robot modelling approaches [122]. These approaches further simplify the physical representation of the system by assuming the *CR* to be represetable as a set of springs, masses and dampers connected in series. When used for a relatively few rigid segments, lumped models avoid the complex expressions intrinsic in continuum descriptions, leading to efficient results [123]. Lumped parameter models have been used in [35] for the *CPRs* kinetostatic modelling. As for constant curvature approaches, lumped parameter approaches requires additional computational effort to accurately represent the complex dynamics of *CRs* [124], but the resulting mathematical models resemble traditional rigid-link robot models;

Discretized Approaches: these techniques are based on the use of additional numerical simplifications added on the *CRs* physical model, but the allowed *CR*

motion is in accordance with Eq. (2.9). The following non-exhaustive list proposes some of the most popular discretized approaches.

- **Finite differences** approach introduces further assumptions on the top of physical models to get simplified modelling representation. As finite differences approximate derivatives [125], the resulting beams equilibrium equations do not require any ODE integration and it can be written explicitly [80], thus simplifying mathematical manipulation of these equations (e.g. for derivative computations). However, a large number of elastic coordinates is frequently required to achieve *CRs* accuracy. Finite differences approximation is successfully employed for *CRs* in concentric tube modelling [126], for their stability analysis, and for the quantification of the number of stable solutions [102]. Also, beams computer graphics simulations benefit from the use of finite-differences [127]. This approach has been applied for the forward and inverse position problem of planar *CPRs* [80]. As an advantage, finite differences provide analytical formulations of the *CRs* geometrico-static problem, but other discretization techniques offer a better trade-off between accuracy and computational cost [128];

- **Collocation methods** are a powerful class of techniques for the solution of differential problems [125]. Starting with the discretization of the strong form and the approximation of the configuration variables with basis functions, several interpolatory functions are used in the context of beam modelling to simplify the geometry of the beam. Basis functions include Chebyshev polynomials [129], B-Splines [130], magnus expansion [131], and NURBS [132] (also known as isogeometric collocation). To the author's knowledge, collocation methods are not currently used in *CPRs*;

- **The finite element method** of slender elastic rods received significant attention since the early development of the beams model's [111]. Finite-element approaches discretize the continuum beam into a finite set of deformable segments. Thanks to the use of powerful numerical tools [133], promising computation performances and accuracy can be achieved [134]. Finite-element methods provide useful tools for model-based closed-loop control of continuum robots [135], and for the dynamic modelling of *CPRs* [136]. The main advantage of the finite-element approach is the possibility of modelling various geometries, but the implementation of finite-element methods is not trivial;

- **Piecewise constant strain** approaches provide a good tradeoff between accuracy and computational cost [137]. Piecewise constant strain approaches approximate the beam's strain field into a finite set of constant strain segments [138]. However, in contrast to the constant curvature assumptions, also constant extensibility, shear and torsion can be simulated [85]. Piecewise constant strain provides advantages for the dynamic simulations of *CRs* [85] and for their static closed-loop control [139]. To the author's knowledge, piecewise constant strains have not been used in *CPRs*;

- **Assumed strain modes** approaches introduce discretizations at the beams strain level, providing accurate results with a reduced number of elastic coordinates [22]. However, the model's mathematical complexity is high. Assumed strain modes approaches have been tested for the simulation of soft

and continuum robots [140], for the dynamics of variable length beams [141], for the simulation of *CPRs* [128]. Assumed strain mode approaches promise an excelent accuracy-computational cost ratio: the expected model accuracy is comparable to shooting-based approaches, while the computational-cost is drastically lower than other discretization approaches (e.g. finite-differences). However, the complexity of the approach is significant;

In this thesis discretized approaches are preferred to physically-approximated approaches. This decision is mainly driven by the need to get high accuracy of the *CPRs* models for simulation and analysis purposes. In particular, i) the finite-differences approach and ii) the assumed-strain mode approach are selected. Before going into the model description, it is necessary to better justify the decision of using discretized model instead of continuous approaches.

2.1.3 Why Using Discretized Models?

The current state of the art in *CPRs* modelling is mainly focused on the use of shooting approaches [13]. Shooting approaches provide the following paramount advantages in the case of quasi-static simulations[2]:

- Accuracy. Since no additional assumptions are introduced, shooting approaches provides accurate results [18];
- Computational cost. It has been demonstrated that shooting approaches perform well in relation to the computational cost [75];

Instead, the major drawbacks of shooting approaches are here listed:

- Complex inclusion of intermediate constraints and intermediate loads. As previosly mentioned, multiple shooting approaches are used to include intermediate constraints or intermediate loads, and the complexity of these approaches is significant.
- Differential equation integration. When dealing with shooting approaches, differential equations should be integrated considering the $SO(3)$ (or $SE(3)$) group structure as shown in Eq. (2.3), with further complications on the numerical integration process.
- Equilibrium stability assessment. In the case shooting approaches being employed, complex *OC* techniques are required to characterize the equilibrium stability.

In relation to the same mentioned points, discretized techniques perform differently depending on the selected discretization approach. The inclusion of intermediate loads or constraints requires complex multiple-shooting approaches if shooting techniques are used while, to the author's experience, discretized techniques includes these load in a simpler manner. Additionally, the equations of different discretization techniques may be written with the same formalism, and the discretization

[2]In the case of dynamic simulations, these advantages are not valid anymore: the shooting approach for dynamics simulation as described in [142] may lack in accuracy, and the stability of the numerical approach is complex to be guaranteed [140]

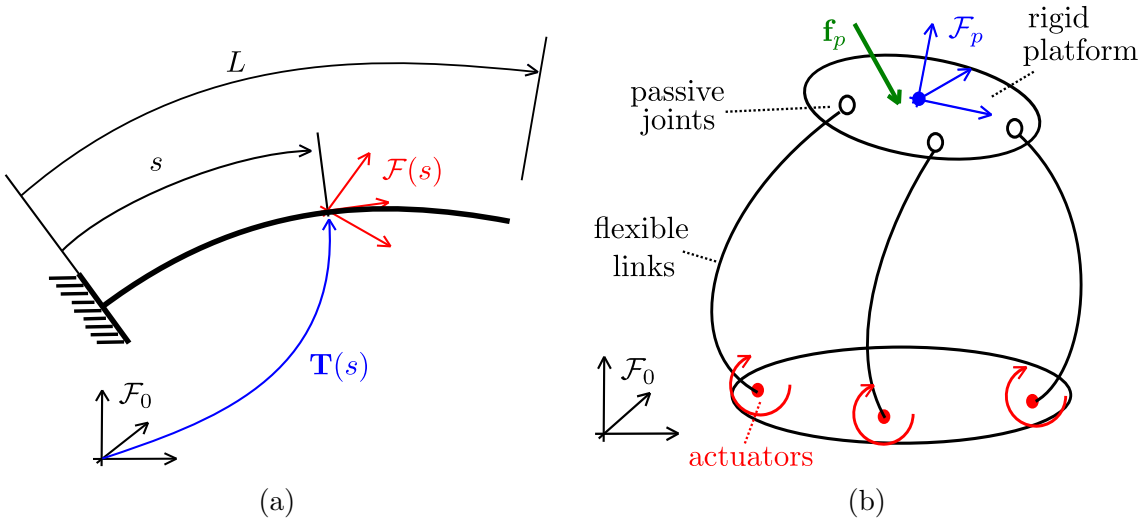


Figure 2.3: (a) Continuous parametrization of a slender beam, and (b) schematics of a *CPR*.

technique is decided only in relation to the predefined model's scope (e.g. computational performances [128], analytical formulation of the model equations [80]). Finally, no matter the selected discretization approach, discretized techniques enable simpler but effective **equilibrium stability assessment**, and this is a fundamental aspect in performance evaluation *CRs*. While the equilibrium stability assessment requires complex *OC* techniques in the case shooting approaches are employed, discretized *CRs* equation provides straightforward equilibrium stability conditions by the evaluation of the Hessian matrix of the *CRs* potential energy. These are the reasons that motivated our selection.

2.2 Energy-Based Discretized Modelling Approach

In this Section, the *CPRs* modelling framework employed in this thesis is described. The energy-based discretized modelling of *CPRs*[3] can be found in the state of the art, but it is described to clarify the notations and the concept later used during *CPRs* analysis. The description starts by deriving the potential energy of an isolated flexible beam in Sec. 2.2.1. Then, the *CPR* model is introduced as an assembly of several rigid and flexible components: configuration variables, the *CPR* energy, and geometric constraints are described in Sec. 2.2.2. The discretization process and discretized robot equations are introduced in Sec. 2.2.3, while the configuration analysis (in terms of equilibrium stability and singularity analysis) are discussed in Sec. 2.2.4. Finally, Sec. 2.3 discusses two discretization techniques employed in this thesis and highlights how to formulate geometrico-static problems with these approaches.

2.2.1 Beams potential energy

Instead of imposing the beam's equilibrium by considering the differential equations Eq. (2.7), an equivalent and alternative approach is based on the use of energetic

[3]Energy-based approaches are the starting point for the so called weak-form approaches [111].

considerations. For the scope of this thesis, let us consider a clamped slender beam as illustrated in Fig. 2.3a. The index i-th $i = 1, \cdots, n_b$ labels variables of the i-th beam, with n_b the total beams number composing the *CPR*. As previously mentioned, a variable-curvature Kirchhoff beam model is employed. The deformation energy of the i-th beam is given by [143]:

$$V_{ei} = \frac{1}{2} \int_0^{L_i} \mathbf{\Gamma}_i^T(s) \boldsymbol{\xi}_i(s) ds \tag{2.10}$$

Assuming material properties as elastic, linear, isotropic, and constant over the length of the beam, the expression of $\mathbf{\Gamma}_i$ of Eq. (2.6), and Eq. (2.10) simplifies as follows:

$$V_{ei} = \frac{1}{2} \int_0^{L_i} \left(\boldsymbol{\xi}_i(s) - \boldsymbol{\xi}_i^*(s)\right)^T \mathbf{K} \left(\boldsymbol{\xi}_i(s) - \boldsymbol{\xi}_i^*(s)\right) ds \tag{2.11}$$

where, according to Eq. (2.3), $\boldsymbol{\xi}_i \in \mathbb{R}^6$ is obtained from $\widehat{\boldsymbol{\xi}}_i(s) = \mathbf{G}_i^{-1}(s)\mathbf{G}_i(s)$. In the case a Kirchhoff inextensible strain model is used, $\mathbf{v}_i = \mathbf{e}_3 = [0; 0; 1]$, and V_e simplifies as:

$$V_{ei} = \frac{1}{2} \int_0^{L_i} \left(\mathbf{u}_i(s) - \mathbf{u}_i^*(s)\right)^T \mathbf{K}_{BT} \left(\mathbf{u}_i(s) - \mathbf{u}_i^*(s)\right) ds \tag{2.12}$$

This assumption usually holds for long and slender beams. Moreover, in the case of an initially straight beam, $\mathbf{u}_i^* = \mathbf{0}$.

Then, let us consider external loads. Assuming that only a distributed conservative force $\mathbf{f}_d \in \mathbb{R}^3$ (expressed in global coordinates) acts on the beam, the contribution of distributed loads V_{di} can be obtained as:

$$V_{di} = -\int_0^{L_i} \mathbf{f}_d^T \mathbf{p}_i(s) ds \tag{2.13}$$

Instead, the contribution of a concentrated conservative force $\mathbf{f}_c \in \mathbb{R}^3$ applied at $\mathbf{p}_i(s_n)$, whose components are represented in $\mathcal{F}_0$, is formulated as:

$$V_{ci} = -\mathbf{f}_c^T \mathbf{p}_i(s_n) \tag{2.14}$$

Three-dimensional distributed or concentrated moments that are non-conservative by essence are considered to not appear [144][4]. The total potential energy of the beam is obtained as the sum of deformation energy and external loads energy, that is:

$$V_{beam_i} = V_{ei} + V_{di} + V_{ci} \tag{2.15}$$

2.2.2 *CPRs* potential energy, constraints

In this Section, the *CPRs* energy expression is derived. Let us consider a *CPR* composed of n_b flexible beams (Fig. 2.3b). Each beam is connected at one extremity to a motor and, at the opposite end, to a rigid platform with a passive joint. The beam is considered as clamped at the actuator sections: these assumptions do not limit the generality of the approach, and prismatic, rotative or actuators that varies

[4]Instead, one-dimensional moments normal to the displacement plane are conservative and have an associated potential energy function. This frequently happens in planar models.

the beam length can be considered [13]. The robot base frame is denoted with $\mathcal{F}_0$, and a frame $\mathcal{F}_p$ is attached to the rigid platform. Under these assumptions, the following *CPRs* variables are considered:

- **Actuated variables**. The i-th actuated variable is called q_{ai}, and the vector $\mathbf{q}_a = [q_{a1}, \cdots, q_{an}] \in \mathbb{R}^n$ collects the actuated variables. In the general case $n \neq n_b$ since beams can be passive or actuated by the same motor;
- **Platform pose variables**. The vector $\mathbf{q}_p$ collects the pose variables, and in general $\mathbf{q}_p = [\mathbf{p}_p, \boldsymbol{\varphi}] \in \mathbb{R}^{n_c}$, where $n_c = 3$ for the planar case, $n_c \geqslant 6$ for the spatial case, and $\mathbf{p}_p, \boldsymbol{\varphi}$ represent the position and the orientation parameters of the platform w.r.t. $\mathcal{F}_0$, respectively[5]. The platform orientation matrix $\mathbf{R}_p \in SO(3)$ is obtained from $\boldsymbol{\varphi}$ in relation to the chosen orientation parametrization;
- **Controlled variables**. Assuming the same number of controlled and actuated variables, $\mathbf{q}_c \in \mathbb{R}^n$ collects the controlled variables. Usually, $\mathbf{q}_c$ is a subset of $\mathbf{q}_p$;
- **Uncontrolled platform variables**. In the case $n \leqslant n_c$, some variables of $\mathbf{q}_p$ are not controlled. These variables are grouped into $\mathbf{q}_u \in \mathbb{R}^{n_c - n}$ for later convenience.

After this variables classification, it is possible to compute the total *CPR* energy. In particular, considering only a concentrated force $\mathbf{f}_p \in \mathbb{R}^3$ applied at the *CPRs* platform, the contribution of $\mathbf{f}_p$ to the *CPR* energy is given by:

$$V_p = -\mathbf{f}_p^T \mathbf{p}_p \tag{2.16}$$

Three-dimensional platform moments, which are non-conservative, are considered to not appear. The total potential energy of the *CPR* is finally obtained as:

$$V_{tot} = \sum_{i=1}^{n_b} V_{beam_i} + V_p \tag{2.17}$$

Due to the closed-loop architecture of *CPRs*, position and orientation geometric constraints have to be enforced. *CPRs* passive joints, which connect the rigid platform to the passive beams, serve as a connection to create the parallel robot architecture. For instance, revolute joints [34], spherical joints [39], cylindrical joints [27] and fixed joints [13] can be modelled. Without loss of generality, the constraint of the i-th beam $\boldsymbol{\Phi}_i \in \mathbb{R}^{n_{\phi_i}}$ can be represented by:

$$\boldsymbol{\Phi}_i = \mathbf{C}_i \begin{bmatrix} \widetilde{\left(\mathbf{R}_p^T \mathbf{R}_i(L_i) - \mathbf{R}_i^T(L_i)\mathbf{R}_p\right)} \\ \mathbf{p}_i(L_i) - (\mathbf{p}_p + \mathbf{R}_p \mathbf{p}_{fi}) \end{bmatrix} \tag{2.18}$$

where $\mathbf{p}_{fi} \in \mathbb{R}^3$ is a vector pointing from the i-th joint position to the platform centre, and constant w.r.t. $\mathcal{F}_p$, $\mathbf{p}_p, \mathbf{R}_p$ are the position and the orientation matrix of $\mathcal{F}_p$ w.r.t. $\mathcal{F}_0$, respectively. The vector $\mathbf{p}_i(L_i)$ is the position of the i-th beams at the coordinates L_i and $\mathbf{R}_i(L_i)$ is the orientation matrix at L_i of the same beam. The superscript $\widetilde{(.)}$ indicates the extraction of the three independent components of its

[5] In the case the platform orientation is parametrized by using a minimal representation (e.g. Euler's angles), $n_c = 6$ while, if quaternions are used, $n_c = 7$.

Name	Fixed	Spherical	Revolute Local z	Cylindrical Local z
n° *DoFs*	0	3	1	2
$\mathbf{C}$	$\mathbf{I}_6$	$\begin{bmatrix}\mathbf{0}_{3\times3} & \mathbf{I}_3\end{bmatrix}$	$\begin{bmatrix}\mathbf{I}_2 & \mathbf{0}_{2\times4} \\ \mathbf{0}_{3\times3} & \mathbf{I}_3\end{bmatrix}$	$\begin{bmatrix}\mathbf{0}_2 & \mathbf{0}_{2\times1} & \mathbf{I}_{2\times2} & \mathbf{0}_{2\times1} \\ \mathbf{I}_2 & \mathbf{0}_{2\times1} & \mathbf{0}_2 & \mathbf{0}_{2\times1}\end{bmatrix}$

Table 2.2: Values of $\mathbf{C}$ for different platform constraints. The number of *DoFs* is indicated with n° *DoFs*, $\mathbf{I}_k$ is the identify matrix of dimension k, and $\mathbf{0}_{k\times h}$ denotes a matrix of dimension $k \times h$ full of zeros. Revolute and Cylindrical pairs are assumed to be aligned with the z local axis.

argument, assumed to be a skew-symmetric matrix. Matrix $\mathbf{C}_i \in \mathbb{R}^{n_{\phi_i}\times6}$ is named joint matrix, and Table 2.2 summarizes how $\mathbf{C}_i$ is structured for different kinds of passive platform joints. Finally, the vector $\mathbf{\Phi} \in \mathbb{R}^{n_\phi}$, $n_\phi = \sum_{i=1}^{n_\psi} n_{\phi_i}$ is introduced to stack all the geometric constraints $\mathbf{\Phi}_i$ in a single vector, and n_ϕ is the number of constraints to be considered.

2.2.3 Discretized *CPR* equations and equilibrium conditions

CPRs equilibrium configurations are associated with critical points of the robot potential energy V_{tot} of Eq. (2.17) [56]. In particular, by inspecting V_{tot} and its terms, it is possible to note that V_{tot} depends on, the actuated variables $\mathbf{q}_a$, the platform variables $\mathbf{q}_p$, a set of continuous functions $\boldsymbol{\xi}_i$ and $\mathbf{p}_i, \mathbf{R}_i$, related together by the differential equations (2.3). Finding the values of $\mathbf{q}_a, \mathbf{q}_p$ and the exact functions $\boldsymbol{\xi}_i$ that lead to equilibrium conditions of the *CPR* is not trivial. A practical way to solve this problem is to employ discretization strategies. To obtain a discretized form of V_{tot}, the following variables are introduced:

- **Discretization variables.** These variables are used to represent V_{tot} and $\mathbf{\Phi}$ with a finite set of coordinates (instead of using the continuous functions $\boldsymbol{\xi}_i$ and $\mathbf{p}_i, \mathbf{R}_i$). To keep the modelling description general, the details on how the discretization coordinates are chosen are addressed in Sec. 2.3. The discretization coordinates of each beam are collected into $\mathbf{q}_{ei} \in \mathbb{R}^{m_i}$, with m_i the number of discretization coordinates of the i-th beam, and the vector $\mathbf{q}_e \in \mathbb{R}^m$, $m = \sum_{i=1}^{n_b} m_i$ stacks all the discretization variables.

- **Passive variables**. The vector $\mathbf{q}_d = [\mathbf{q}_u, \mathbf{q}_e] \in \mathbb{R}^{m+n_c-n}$ groups the variables that are not actuated and not controlled.

- **Non-actuated variables.** The vector $\mathbf{x}$ is introduced to collect all the variables that are non-actuated. Thus, $\mathbf{x} = [\mathbf{q}_d, \mathbf{q}_c] \in \mathbb{R}^{m+n_c}$.

After introducing the discretization variables, V_{tot}, that was a functional, becomes a discrete function of the previously defined variables, that is $V_{tot} = V_{tot}(\mathbf{q}_a, \mathbf{x})$. A robot configuration is an equilibrium configuration if, for fixed values of $\mathbf{q}_a$, $\mathbf{x}$ is a critical point of V_{tot} [145]. However, critical points of V_{tot} are subjected to the verification of geometric constraints $\mathbf{\Phi}$, and $\mathbf{x}$ can be identified by the solution of a constrained minimization problem. A pratical way to identify critical points of V_{tot} is to consider Lagrange conditions [145], which defines sufficient conditions for $\mathbf{x}$ to be a critical point of V_{tot} subjected to $\mathbf{\Phi}$. Let us first define the Lagrangian function

$\mathcal{L}$ as:

$$\mathcal{L} = V_{tot} + \mathbf{\Phi}^T \boldsymbol{\lambda} \tag{2.19}$$

with $\boldsymbol{\lambda} \in \mathbb{R}^{n_\Phi}$ a vector of Lagrange multipliers. Assuming that $\nabla_{\mathbf{x}} \mathbf{\Phi}$ is full rank, $\mathbf{x}$ is a critical point if there exist $\boldsymbol{\lambda} \in \mathbb{R}^{n_\phi}$ such as [145]:

$$\begin{cases} \nabla_{\mathbf{x}} \mathcal{L} = \nabla_{\mathbf{x}} V_{tot} + \nabla_{\mathbf{x}} \mathbf{\Phi}^T \boldsymbol{\lambda} = \mathbf{0} \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L} = \mathbf{\Phi} = \mathbf{0} \end{cases} \tag{2.20}$$

Equations (2.20) represent the implicit geometrico-static model of a *CPR*, and it is a set of $m+n_c+n_\phi$ equations in $m+n_c+n_\phi+n$ unknowns. Thus, by fixing n variables, a square problem is obtained to be solved numerically . The **configuration variables** of the robot are defined by the vector $\mathbf{y} = [\mathbf{q}_a, \mathbf{x}, \boldsymbol{\lambda}] \in \mathbb{R}^{n+m+n_c+n_\phi}$. Typically, two geometrico-static problems are of interest. On the one hand, the forward geometrico-static problem (*FGSP*) consists in the evaluation of $\mathbf{q}_d, \mathbf{q}_c, \boldsymbol{\lambda}$ for given external loads and assigned $\mathbf{q}_a$, that is

$$\mathbf{F}(\mathbf{y}) = \begin{cases} \nabla_{\mathbf{x}} V_{tot} + \nabla_{\mathbf{x}} \mathbf{\Phi}^T \boldsymbol{\lambda} = \mathbf{0} \\ \mathbf{\Phi} = \mathbf{0} \\ \mathbf{q}_a - \mathbf{q}_a^d = \mathbf{0} \end{cases} \tag{2.21}$$

where $\mathbf{q}_a^d \in \mathbb{R}^n$ is the assigned motor values. On the other hand the inverse geometrico-static problem (*IGSP*) consists in the evaluation of $\mathbf{q}_d, \mathbf{q}_a, \boldsymbol{\lambda}$ for given external loads and assigned $\mathbf{q}_c$, that is:

$$\mathbf{F}(\mathbf{y}) = \begin{cases} \nabla_{\mathbf{x}} V_{tot} + \nabla_{\mathbf{x}} \mathbf{\Phi}^T \boldsymbol{\lambda} = \mathbf{0} \\ \mathbf{\Phi} = \mathbf{0} \\ \mathbf{q}_c - \mathbf{q}_c^d = \mathbf{0} \end{cases} \tag{2.22}$$

and $\mathbf{q}_c^d \in \mathbb{R}^n$ is the assigned controlled variables value. In the general case, geometrico-static problems can be formulated in a unified way:

$$\mathbf{F}(\mathbf{y}) = \begin{cases} \nabla_{\mathbf{x}} V_{tot} + \nabla_{\mathbf{x}} \mathbf{\Phi}^T \boldsymbol{\lambda} = \mathbf{0} \\ \mathbf{\Phi} = \mathbf{0} \\ \mathbf{e} = \mathbf{0} \end{cases} \tag{2.23}$$

with $\mathbf{e} \in \mathbb{R}^n$ the equations that fix n variables of $\mathbf{y}$ (or a combination of its variables) to a defined value. Equations (2.23) form a square system of equations of $n + m + n_c + n_\phi$ equations in $\mathbf{y}$ and, since Eq. (2.23) is nonlinear, and root-finding techniques are employed to identify numerical solutions. In particular, the Jacobian matrix $\mathbf{J}$ of the forward/inverse geometrico-static problems can be supplied to the solver to accelerate the computation. This matrix is structured as follows [56]:

$$\mathbf{J} = \frac{\partial \mathbf{F}}{\partial \mathbf{y}} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{U}_1 & \mathbf{P}_1 & \mathbf{\Lambda}^T \\ \mathbf{A}_2 & \mathbf{U}_2 & \mathbf{P}_2 & \mathbf{0}_{n_\phi \times n_\phi} \\ \mathbf{E} & \mathbf{0}_{n \times (m+n_c-n)} & \mathbf{I}_n - \mathbf{E} & \mathbf{0}_{n \times n_\phi} \end{bmatrix} \tag{2.24}$$

where $\mathbf{J}$ is square of dimension $m + n_c + n_\phi + n$, matrix $\mathbf{I}_n$ is the identity matrix of dimension n. The matrix $\mathbf{E}$ is equal to $\mathbf{I}_n$ if the forward problem is considered (Eq. (2.21)) or equal to $\mathbf{0}_{(n \times n)}$ if the inverse problem of Eq. (2.22) is solved. The other matrices are defined as:

1. $\mathbf{A}_1 = \nabla_{\mathbf{q}_a}(\nabla_{\mathbf{x}}\mathcal{L}) \in \mathbb{R}^{(m+n_c)\times n}$, $\mathbf{U}_1 = \nabla_{\mathbf{q_d}}(\nabla_{\mathbf{x}}\mathcal{L}) \in \mathbb{R}^{(m+n_c)\times(m+n_c-n)}$
2. $\mathbf{P}_1 = \nabla_{\mathbf{q}_c}(\nabla_{\mathbf{x}}\mathcal{L}) \in \mathbb{R}^{(m+n_c)\times n}$, $\mathbf{\Lambda} = \nabla_{\mathbf{x}}(\nabla_{\boldsymbol{\lambda}}\mathcal{L}) \in \mathbb{R}^{(m+n_c)\times n_\phi}$
3. $\mathbf{A}_2 = \nabla_{\mathbf{q}_a}\mathbf{\Phi} \in \mathbb{R}^{n_\phi \times n}$, $\mathbf{U}_2 = \nabla_{\mathbf{q}_d}\mathbf{\Phi} \in \mathbb{R}^{n_\phi\times(m+n_c-n)}$, $\mathbf{P}_2 = \nabla_{\mathbf{q}_c}\mathbf{\Phi} \in \mathbb{R}^{n_\phi\times n}$

It should be also noted that $[\mathbf{U}_2, \mathbf{P}_2] = \mathbf{\Lambda}$, since derivatives w.r.t. $\boldsymbol{\lambda}, \mathbf{x}$ are commutative.

2.2.4 Equilibrium Configuration Analysis

If a solution $\mathbf{y}^\star$ of Eq. (2.22) is found, the practical feasibility of this configuration must be checked: equilibrium stability and singularity conditions should be considered. Moreover, strains over the beams and actuation torques are important for the pratical feasibility of the robot configurations.

A linear approaximation of conventional strain quantities $\boldsymbol{\epsilon} \in \mathbb{R}^3$ used in beam mechanics is recovered as:

$$\boldsymbol{\epsilon}(s) = [\gamma_{xz}, \gamma_{yz}, \epsilon_z] = -\mathbf{r}(s) \times \mathbf{u}(s) \tag{2.25}$$

where $\mathbf{r} = [x_s, y_s, 0] \in \mathbb{R}^3$ is the position of a point that lies over the cross-section in s, and it has coordinate $x_s, y_s, 0$ w.r.t. the local frame $\mathcal{F}_s$. γ_{xz}, γ_{yz} represent shear strains approximation, and ϵ_z is the normal strain. The i-th actuation torque $\tau_i \in \mathbb{R}$ is computed computed as [56]:

$$\tau_i = \nabla_{q_{ai}} V_{tot}(\mathbf{q}_a, \mathbf{x}) + \nabla_{q_{ai}} \mathbf{\Phi}^T(\mathbf{q}_a, \mathbf{x})\boldsymbol{\lambda} \tag{2.26}$$

To analyze equilibrium stability and singularity conditions, a linearization of Eq. (2.20) around the solution $\mathbf{y}^\star, \mathbf{f}_p$ results in [56]:

$$\begin{bmatrix}\mathbf{A}_1\\ \mathbf{A}_2\end{bmatrix}\Delta\mathbf{q}_a + \begin{bmatrix}\mathbf{U}_1\\ \mathbf{U}_2\end{bmatrix}\Delta\mathbf{q}_d + \begin{bmatrix}\mathbf{P}_1\\ \mathbf{P}_2\end{bmatrix}\Delta\mathbf{q}_c + \begin{bmatrix}\mathbf{\Lambda}^T\\ \mathbf{0}\end{bmatrix}\Delta\boldsymbol{\lambda} + \begin{bmatrix}\mathbf{W}_1\\ \mathbf{0}\end{bmatrix}\Delta\mathbf{f}_p = \mathbf{0} \tag{2.27}$$

where $\mathbf{W}_1 = \nabla_{\mathbf{f}_P}(\nabla_{\mathbf{x}}\mathcal{L}) \in \mathbb{R}^{(m+n_c)\times 3}$. It should be noted that matrices defined in Eq. (2.27), are the same matrices necessary for the Jacobian matrix of Eq. (2.24), and thus these matrices are directly available as an output of the solver after the evaluation of the solution $\mathbf{y}^\star$.

For the singularity analysis, there is little interest in the variation $\Delta\boldsymbol{\lambda}$, since degeneracies of $\mathbf{\Lambda}$ are unlikely to occur, in practice [56]. Being $\mathbf{Z} \in \mathbb{R}^{(m+n_c)\times(m+n_c-n_\phi)}$ the matrix spanning the right nullspace of $\mathbf{\Lambda}$, that is:

$$\mathbf{\Lambda}\mathbf{Z} = \mathbf{0} \tag{2.28}$$

$\Delta\boldsymbol{\lambda}$ is eliminated by multiplying the first row of Eq. (2.27) by $\mathbf{Z}^T$ leading to:

$$\mathbf{A}\Delta\mathbf{q}_a + \mathbf{U}\Delta\mathbf{q}_d + \mathbf{P}\Delta\mathbf{q}_c + \mathbf{W}\Delta\mathbf{f}_p = \mathbf{0} \tag{2.29}$$

where:

$$\mathbf{A} = \begin{bmatrix}\mathbf{Z}^T\mathbf{A}_1\\ \mathbf{A}_2\end{bmatrix}, \mathbf{U} = \begin{bmatrix}\mathbf{Z}^T\mathbf{U}_1\\ \mathbf{U}_2\end{bmatrix}, \mathbf{P} = \begin{bmatrix}\mathbf{Z}^T\mathbf{P}_1\\ \mathbf{P}_2\end{bmatrix}, \mathbf{W} = \begin{bmatrix}\mathbf{Z}^T\mathbf{W}_1\\ \mathbf{0}\end{bmatrix} \tag{2.30}$$

To derive singularity conditions, the inverse and forward kinematostatic problems are established [56]. The inverse kinematostatic problem means to evaluate how

a variation of the controlled variables $\Delta\mathbf{q}_c$ and the external platform force $\Delta\mathbf{f}_p$ is reflected on the actuated variables $\Delta\mathbf{q}_a$ and on the passive variables $\Delta\mathbf{q}_d$, that is:

$$\begin{bmatrix}\Delta\mathbf{q}_a\\ \Delta\mathbf{q}_d\end{bmatrix} = -[\mathbf{A}\ \mathbf{U}]^{-1}(\mathbf{P}\Delta\mathbf{q}_c + \mathbf{W}\Delta\mathbf{f}_p) \tag{2.31}$$

Equation (2.31) is solvable as long as $\mathbf{T}_1 = [\mathbf{A}\ \mathbf{U}] \in \mathbb{R}^{(m+n_c)\times(m+n_c)}$ is full-rank, and rank deficiencies of $\mathbf{T}_1$ are named Type-1 singularities [56]. These singularities are related to limits of *IGSP* and impossible motions of $\mathbf{q}_c$. On the other hand, the forward kinematostatic problem consists in evaluating $\Delta\mathbf{q}_p, \Delta\mathbf{q}_d$ for given $\Delta\mathbf{q}_a, \Delta\mathbf{f}_p$, that is:

$$\begin{bmatrix}\Delta\mathbf{q}_c\\ \Delta\mathbf{q}_d\end{bmatrix} = -[\mathbf{P}\ \mathbf{U}]^{-1}(\mathbf{A}\Delta\mathbf{q}_a + \mathbf{W}\Delta\mathbf{f}_p) \tag{2.32}$$

Equation (2.32) is solvable as long as $\mathbf{T}_2 = [\mathbf{P}\ \mathbf{U}] \in \mathbb{R}^{(m+n_c)\times(m+n_c)}$ is full-rank, and degeneracies of $\mathbf{T}_2$ are named Type-2 singularities [56]. Robot configurations where $\mathbf{T}_2$ is degenerated are related to the limits of the *FGSP* solution and uncontrollable $\Delta\mathbf{q}_c$ motions.

Then, equilibrium stability is evaluated by determining the reduced Hessian matrix $\mathbf{H}^r$ of the total potential energy as [56]:

$$\mathbf{H}^r = \mathbf{Z}^T\frac{\partial^2\mathcal{L}}{\partial\mathbf{x}^2}\mathbf{Z} = \mathbf{Z}^T\mathbf{H}\mathbf{Z} \in \mathbb{R}^{(m+n_c-n_\phi)\times(m+n_c-n_\phi)} \tag{2.33}$$

where $\mathbf{H} = [\mathbf{P}_1, \mathbf{U}_1]$. The configuration is stable if $\mathbf{H}^r$ is positive definite [145]. Please note that, as long as $\mathbf{Z}$ is full rank, $\mathbf{T}_2$ is singular if and only if $\mathbf{H}^r$ is rank deficient (see [56] for the proof). Thus, Type-2 singularities are associated with the variation of the stability pattern.

2.3 Selected Discretization Techniques

This Section illustrates the two discretization techniques employed in this thesis: the assumed strain mode discretization [22] and the finite-differences discretization [127]. The core assumptions of these approaches is here briefly described as well as the final equations for the geometrico-static problems formulations. Instead, dedicated appendices illustrate the derivation of these equations in detail.

2.3.1 Assumed Strain Modes Approach

The assumed strain mode approach promises good level of accuracy (comparable to shooting-like approaches [128]), while keeping reduced the computational time (e.g. in comparison with finite-differences methodologies [80]). Thus, a good trade-off between computational performances and accuracy is expected. However, the derivation of the model terms requires the integration of ODEs, further increasing its complexity. In this Section, the final results of the geometrico-static model equations are reported, and Appendix A is dedicated to the derivation of these equations.

In the case of shear and extensibility being negligible, the i-th beams curvature $\mathbf{u}_i(s) \in \mathbb{R}^3$ is approximated as:

$$\mathbf{u}_i(s) = \mathbf{N}(s)\mathbf{q}_{ei} \tag{2.34}$$

where $\mathbf{q}_{ei} \in \mathbb{R}^{m_i}$ is the set of discretization coordinates of the i-th beam. The matrix $\mathbf{N}$, which collects basis functions, is structured as:

$$\mathbf{N}(s) = \begin{bmatrix} \mathbf{b}^T(s) & \mathbf{0}_{1\times N_f} & \mathbf{0}_{1\times N_f} \\ \mathbf{0}_{1\times N_f} & \mathbf{b}^T(s) & \mathbf{0}_{1\times N_f} \\ \mathbf{0}_{1\times N_f} & \mathbf{0}_{1\times N_f} & \mathbf{b}^T(s) \end{bmatrix} \in \mathbb{R}^{3\times m_i} \tag{2.35}$$

where $\mathbf{b} \in \mathbb{R}^{N_f\times 1}$ is a base function vector, N_f is the number of base functions employed in $\mathbf{b}$, and $m_i = 3N_f$. Common base function selections involve standard monomials [22]:

$$\mathbf{b}(s) = \begin{bmatrix} 1, & s, & s^2, & s^3, \cdots, & s^{N_f-1} \end{bmatrix} \tag{2.36}$$

However, when using standard monomials, the matrix $\mathbf{J}$ of Eq. (2.24) is frequently ill-conditioned [128], and orthogonal Legendre monomials are selected to alleviate this issue [146]:

$$\mathbf{b}(s) = \begin{bmatrix} 1, & s, & \frac{1}{2}(3s^2-1), & \frac{1}{2}(5s^3-3s), & \cdots, & (2+1/N_f)sb_{N_f-1} - b_{N_f-2} \end{bmatrix} \tag{2.37}$$

where b_j is the j-th component of $\mathbf{b}$. According to this choice of parametrization of the beam strain, the vector $\boldsymbol{\xi}$ can be computed as:

$$\boldsymbol{\xi}_i(s) = \begin{bmatrix} \mathbf{N}(s)\mathbf{q}_{ei} \\ \mathbf{e}_3 \end{bmatrix} \tag{2.38}$$

and the position and orientation of each beams cross-section is obtained by integrating Eq. (2.3). However, preserving the structure $SE(3)$ of matrix $\mathbf{G}$ during the integration of Eq. (2.3) requires the use of structure-preserving integrators, which are computationally expensive [147]. Instead, by parametrizing the cross-section orientation with unit quaternions $\mathbf{h}_i \in \mathbb{R}^4$, $\mathbf{h}_i = h_{i1} + h_{i2}\mathbf{e}_1 + h_{i3}\mathbf{e}_2 + h_{i4}\mathbf{e}_3$, $\mathbf{e}_1 = [1,0,0], \mathbf{e}_2 = [0,1,0], \mathbf{e}_3 = [0,0,1]$, simplifies Eq. (2.3) as follows:

$$\begin{cases} \mathbf{h}_i'(s)(s) = \frac{1}{2}\mathbf{Q}_i(s)\mathbf{h}_i(s) \\ \mathbf{p}_i'(s) = \mathbf{R}_i(s)\mathbf{e}_3 \end{cases} \tag{2.39}$$

with $\mathbf{p}_i(0), \mathbf{h}_i(0)$ initial values at the beams' base usually computed from q_{ai}. The matrix $\mathbf{Q}_i$ is structured as follows:

$$\mathbf{Q}_i = \begin{bmatrix} 0 & -u_{i1} & -u_{i2} & -u_{i3} \\ +u_{i1} & 0 & +u_{i3} & -u_{i2} \\ +u_{i2} & -u_{i3} & 0 & +u_{i1} \\ +u_{i3} & +u_{i2} & -u_{i1} & 0 \end{bmatrix} \tag{2.40}$$

After integrating Eq. (2.39) from $s = 0$ to $s = L_i$ for a given $\mathbf{q}_{ei}$, the position $\mathbf{p}_i(L_i)$ and the orientation matrix $\mathbf{R}_i(L_i)$ of the beam at the section $s = L_i$ are known, and the vector $\boldsymbol{\Phi}_i$ can be evaluated as in Eq. (2.18). Then, to obtain the term $\nabla_{\mathbf{x}}\mathcal{L} = [\nabla_{\mathbf{q}_e}\mathcal{L}, \nabla_{\mathbf{q}_p}\mathcal{L}]$ of Eq. (2.23), the term $\nabla_{\mathbf{q}_e}\mathcal{L}$ is first considered. In particular, the following equation is the final expressions of the i-th beam equilibrium equation, and its derivation is reported in Appendix A:

$$\nabla_{\mathbf{q}_{ei}}\mathcal{L} = \mathbf{K}_{ei}\mathbf{q}_{ei} + \mathbf{Q}_{ci} = \mathbf{0} \tag{2.41}$$

where $\mathbf{K}_{ei} \in \mathbb{R}^{m_i \times m_i}$ is a **constant** matrix obtained by integration as follows:

$$\mathbf{K}_{ei} = \int_0^{L_i} \mathbf{N}^T(s)\mathbf{K}_{BT}\mathbf{N}(s)ds \tag{2.42}$$

and $\mathbf{Q}_{ci} \in \mathbb{R}^{m_i}$ containts the influence of external loads and tip-constraints of the beam. $\mathbf{Q}_{ci}$ is obtained by integrating from $s = L_i$ to $s = 0$ the following differential equations:

$$\begin{cases} \boldsymbol{\Gamma}_i'(s) = \mathbf{ad}^T_{\boldsymbol{\xi}_i}(s)\boldsymbol{\Gamma}_i(s) - \mathbf{w}_d(s) \\ \mathbf{Q}_{ci}'(s) = \left(\mathbf{B}\mathbf{N}(s)\right)^T \boldsymbol{\Gamma}(s) \end{cases} \tag{2.43}$$

with initial values $\boldsymbol{\Gamma}_i(L) = \mathbf{C}_i\boldsymbol{\lambda}_i$ and $\mathbf{Q}_{ci}(L) = \mathbf{0}$, $\mathbf{B} = [\mathbf{I}_3; \mathbf{0}_3] \in \mathbb{R}^{6\times 3}$. It should be stressed that $\mathbf{w}_d, \boldsymbol{\lambda}_i$ are expressed in the local beam frame.

Then, concerning $\nabla_{\mathbf{q}_p}\mathcal{L}$, the following equation is derived in Appendix A, and here reported only the final expression:

$$\nabla_{\mathbf{q}_p}\mathcal{L} = \left(-\mathbf{w}_p + \sum_{i=1}^{n_b} \mathbf{Ad}_{t-i}\mathbf{C}_i\boldsymbol{\lambda}_i\right)\mathbf{M} \tag{2.44}$$

where $\mathbf{w}_p = [\mathbf{0}_{3\times 1}; \mathbf{f}_p]$ is the platform wrench in global frame coordinates, $\mathbf{Ad}_{t-i} \in \mathbb{R}^{6\times 6}$ is structured as follows:

$$\mathbf{Ad}_{t-i} = \begin{bmatrix} \mathbf{R}_i(L) & \widehat{\mathbf{p}_{fi}}\mathbf{R}_i(L) \\ \mathbf{0}_{3\times 3} & \mathbf{R}_i(L) \end{bmatrix} \tag{2.45}$$

where $\mathbf{p}_{fi} \in \mathbb{R}^3$ is a vector pointing from the i-th joint position to the platform centre, and constant w.r.t. $\mathcal{F}_p$. The matrix $\mathbf{M} \in \mathbb{R}^{6\times 6}$ relates an infinitesimal variation of $\mathbf{q}_p \in \mathbb{R}^{n_c}$ with the platform twist $\Delta\boldsymbol{\eta} \in \mathbb{R}^6$, that is:

$$\Delta\boldsymbol{\eta} = \mathbf{M}\mathbf{q}_p \tag{2.46}$$

Its expression depends on the selected orientation parametrization, and it is not reported here for brevity. Additional details on the derivation of the assumed mode equations are proposed in Appendix A.

2.3.2 Finite-Differences Discretization

Finite differences are a straightforward numerical approach for the analytical approximation of first-order derivatives [125]. When *CPRs* equations are derived by introducing a finite-difference approximation, the resulting model equations are written in an analytical way, providing the possibility to compute second and third derivatives of the energy analytically. However, to get accuracy, a large number of discretization variables is required, and the computational time drastically increases.

As in the assumed strain mode approach, it is convenient to parametrize the beam's orientation by using unit quaternions $\mathbf{h}_i = h_{i1} + h_{i2}\mathbf{e}_1 + h_{i3}\mathbf{e}_2 + h_{i4}\mathbf{e}_3$. In this case, the curvature of the i-th beam can be computed as:

$$u_{ik} = 2\mathbf{h}_i^T\mathbf{B}_k\mathbf{h}_i', \ k = 1, 2, 3 \tag{2.47}$$

where the matrix $\mathbf{B}_k \in \mathbb{R}^{4\times 4}$ is:

$$\mathbf{B}_1 = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \end{bmatrix}, \mathbf{B}_2 = \begin{bmatrix} 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \mathbf{B}_3 = \begin{bmatrix} 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \tag{2.48}$$

Then, by inserting Eq. (2.47) into Eq. (2.12), the deformation energy of the i-th beam V_{ei} simplifies as:

$$V_{ei} = \frac{1}{2}\int_0^{L_i} \left(\sum_{i=1}^{3} K_k \left(2\mathbf{h}_i^T \mathbf{B}_k \mathbf{h}_i' - u_{ik}^* \right)^2 \right) ds \tag{2.49}$$

with K_k the k-th diagonal term of $\mathbf{K}_{BT}$. In this Section, distributed loads are omitted for brevity, and the detailed expressions including also these efforts are reported in Appendix B. Then, the finite differences assumptions is introduced. First, let us discretize the rod into N_{elt} elements of equal length $L_{ei} = L_i/N_{elt}$. In this case, the expression of V_{ei} becomes the sum of each element's deformation energy, that is:

$$V_{ei} = \sum_{j=1}^{N_{elt}} V_{eij},\ V_{eij} = \frac{1}{2}\int_0^{L_{ei}} \left(\sum_{k=1}^{3} K_k \left(2\mathbf{h}_{ij}^T \mathbf{B}_k \mathbf{h}_{ij}' - u_k^* \right)^2 \right) ds \tag{2.50}$$

the term $\mathbf{h}_{ij}$ is the orientation of the j-th element, and the N_{elt} quaternions that discretize the i-th beams energy are collected into $\mathbf{q}_{ei} \in \mathbb{R}^{m_i}$, with $m_i = 4N_{elt}$. After this, $\mathbf{h}_{ij}'$ is approximated by using first-order finit differences:

$$\mathbf{h}_{ij}' = \frac{\mathbf{h}_{ij} - \mathbf{h}_{ij-1}}{L_e} \tag{2.51}$$

By inserting Eq. (2.51) into Eq. (2.50) a simplified expression of V_{eij} is obtained:

$$V_{eij} = \frac{1}{2}\left(\sum_{k=1}^{3} \frac{K_k}{L_e} \left(2\mathbf{h}_{ij}^T \mathbf{B}_k \frac{\mathbf{h}_{ij} - \mathbf{h}_{ij-1}}{L_e} - L_e u_k^* \right)^2 \right) \tag{2.52}$$

The orientation matrix $\mathbf{R}_{ij}$ of the j-th element can be recovered by the knowledge of $\mathbf{h}_{ij}$, while the position can be obtained by the use of the following formula:

$$\mathbf{p}_i(s) = \mathbf{p}_{ij} + s\mathbf{R}_{ij}\mathbf{e}_3 \tag{2.53}$$

where $\mathbf{p}_0, \mathbf{h}_0$ are the base position and orientation of the beam, computed by q_{ai}. By using Eq. (2.53), the position of the beam at $s = L$ can be computed, and the constraints $\mathbf{\Phi}_i$ are evaluated according to Eq. (2.18). Then, to derive $\nabla_{\mathbf{x}}\mathcal{L} = [\nabla_{\mathbf{q}_e}\mathcal{L}, \nabla_{\mathbf{q}_p}\mathcal{L}]$ of Eq. (2.23), the term $\nabla_{\mathbf{q}_e}\mathcal{L}$ is first considered, and the i-th component, $\nabla_{\mathbf{q}_{ei}}\mathcal{L}$ is structured as follows:

$$\nabla_{\mathbf{q}_{ei}}\mathcal{L} = \nabla_{\mathbf{q}_{ei}} V_{ei} + \nabla_{\mathbf{q}_{ei}}(\boldsymbol{\lambda}_i^T \mathbf{\Phi}_i) \tag{2.54}$$

where:

$$\nabla_{\mathbf{q}_{ei}} V_{ei} = \begin{bmatrix} \mathbf{a}_{i1} + \mathbf{b}_{i2} \\ \vdots \\ \mathbf{a}_{i(N_{elt}-2)} + \mathbf{b}_{i(N_{elt}-1)} \\ \mathbf{a}_{i(N_{elt}-1)} \end{bmatrix} \tag{2.55}$$

and the terms $\mathbf{a}_{ij}, \mathbf{b}_{ik}, A_{ijk}$ can be computed as:

$$\mathbf{a}_{ij} = +2\sum_{k=1}^{3} \frac{K_k}{L_{ei}} \mathbf{B}_k \mathbf{h}_{ij} A_{ijk},\ \mathbf{a}_{ij} \in \mathbb{R}^4 \tag{2.56}$$

$$\mathbf{b}_{ij} = -2\sum_{k=1}^{3} \frac{K_k}{L_{ei}} \mathbf{B}_k \mathbf{h}_{i(j+1)} A_{ijk},\ \mathbf{b}_{ij} \in \mathbb{R}^4 \tag{2.57}$$

$$A_{ijk} = -2\mathbf{h}_{i(j+1)}^T \mathbf{B}_k^T \mathbf{h}_{ij} - u_{ijk}^*,\ A_{ijk} \in \mathbb{R} \tag{2.58}$$

Instead, to derive the expression of $\nabla_{\mathbf{q}_{ei}}(\boldsymbol{\lambda}_i^T\boldsymbol{\Phi}_i)$, the case of a fixed constraint is considered since the other cases can be derived from these results. Consequently, $\mathbf{C}_i = \mathbf{I}_6$, $\boldsymbol{\lambda}_i \in \mathbb{R}^6$. Let us compute the j-th component of $\nabla_{\mathbf{q}_{ei}}(\boldsymbol{\lambda}_i^T\boldsymbol{\Phi}_i)$, namely, $\nabla_{\mathbf{h}_{ij}}(\boldsymbol{\lambda}_i^T\boldsymbol{\Phi}_i)$:

$$\nabla_{\mathbf{h}_{ij}}(\boldsymbol{\lambda}_i^T\boldsymbol{\Phi}_i) = \mathbf{c}_{ij}^T + \mathbf{d}_{ij}^T \tag{2.59}$$

where $\mathbf{c}_{ij} \in \mathbb{R}^4$:

$$\mathbf{c}_{ij} = L_{ei}[\lambda_{4i}, \lambda_{5i}, \lambda_{6i}]\mathbf{D}_{3ij} \tag{2.60}$$

The term $\mathbf{d}_{ij} \in \mathbb{R}^4$ is non-null only if $j = N_{elt}$, and it is structured as follows:

$$\begin{aligned}\mathbf{d}_{ij} = \lambda_{1i}\left(\mathbf{d}_{p1}^T\mathbf{D}_{2ij} - \mathbf{d}_{p2}^T\mathbf{D}_{1ij}\right) + \lambda_{2i}\left(\mathbf{d}_{p1}^T\mathbf{D}_{3ij} - \mathbf{d}_{p3}^T\mathbf{D}_{1ij}\right) + \\ + \lambda_{3i}\left(\mathbf{d}_{p2}^T\mathbf{D}_{3ij} - \mathbf{d}_{p3}^T\mathbf{D}_{2ij}\right),\ j = N_{elt}\end{aligned} \tag{2.61}$$

where $\mathbf{d}_{pi} \in \mathbb{R}^3$ are the columns of $\mathbf{R}_p = [\mathbf{d}_{p1}, \mathbf{d}_{p2}, \mathbf{d}_{p3}]$ and $\mathbf{D}_{kij} \in \mathbb{R}^{3\times4}$ are defined as follows:

$$\mathbf{D}_{1ij} = 2\begin{bmatrix} +h_{1ij} & +h_{2ij} & -h_{3ij} & -h_{4ij} \\ +h_{4ij} & +h_{3ij} & +h_{2ij} & +h_{1ij} \\ -h_{3ij} & +h_{4ij} & -h_{1ij} & +h_{2ij} \end{bmatrix} \tag{2.62}$$

$$\mathbf{D}_{2ij} = 2\begin{bmatrix} -h_{4ij} & +h_{3ij} & +h_{2ij} & -h_{1ij} \\ +h_{1ij} & -h_{2ij} & +h_{3ij} & -h_{4ij} \\ +h_{2ij} & +h_{1ij} & +h_{4ij} & +h_{3ij} \end{bmatrix} \tag{2.63}$$

$$\mathbf{D}_{3ij} = 2\begin{bmatrix} +h_{3ij} & +h_{4ij} & +h_{1ij} & +h_{2ij} \\ -h_{2ij} & -h_{1ij} & +h_{4ij} & +h_{3ij} \\ +h_{1ij} & -h_{2ij} & -h_{3ij} & +h_{4ij} \end{bmatrix} \tag{2.64}$$

Then, the term $\nabla_{\mathbf{q}_p}\mathcal{L}$ can be computed as:

$$\nabla_{\mathbf{q}_p}\mathcal{L} = \nabla_{\mathbf{q}_p}V_{tot} + \nabla_{\mathbf{q}_p}(\boldsymbol{\lambda}_i^T\boldsymbol{\Phi}_i) \tag{2.65}$$

where $\nabla_{\mathbf{q}_p}V = -[\mathbf{0}_{3\times1}; \mathbf{I}_3]$ and, being $\mathbf{q}_p = [\mathbf{p}_p, \boldsymbol{\alpha}] \in \mathbb{R}^{n_c}$ with $\mathbf{p}_p \in \mathbb{R}^3$ the platform position and $\boldsymbol{\alpha} \in \mathbb{R}^{n_c-3}$ the platform orientation parameters, the term $\nabla_{\mathbf{q}_p}(\boldsymbol{\lambda}_i^T\boldsymbol{\Phi}_i)$ is structured as follows:

$$\nabla_{\mathbf{q}_p}(\boldsymbol{\lambda}_i^T\boldsymbol{\Phi}_i) = -[\lambda_4, \lambda_5, \lambda_6]\begin{bmatrix}\mathbf{I}_3 \\ \mathbf{m}_{p1}\end{bmatrix} + \begin{bmatrix}\mathbf{0}_{3\times1} \\ \mathbf{m}_{p2}\end{bmatrix} \tag{2.66}$$

the vector $\mathbf{m}_{p1} \in \mathbb{R}^{n_c-3}$ is:

$$\mathbf{m}_{p1} = \frac{\partial}{\partial\boldsymbol{\alpha}}\left(\mathbf{R}_p\mathbf{p}_{fi}\right) \tag{2.67}$$

and its expression depends on the specific platform orientation parametrization. The expression of $\mathbf{m}_{p2} \in \mathbb{R}^{n_c-3}$ depends on the platform orientation parametrization as well, and it is structured as:

$$\begin{aligned}\mathbf{m}_{p2} = \lambda_1\left(\frac{\partial\mathbf{d}_{p1}^T}{\partial\boldsymbol{\alpha}}\mathbf{d}_{2ij} - \frac{\partial\mathbf{d}_{p2}^T}{\partial\boldsymbol{\alpha}}\mathbf{d}_{1ij}\right) + \lambda_2\left(\frac{\partial\mathbf{d}_{p1}^T}{\partial\boldsymbol{\alpha}}\mathbf{d}_{3ij} - \frac{\partial\mathbf{d}_{p3}^T}{\partial\boldsymbol{\alpha}}\mathbf{d}_{1ij}\right) + \\ + \lambda_3\left(\frac{\partial\mathbf{d}_{p1}^T}{\partial\boldsymbol{\alpha}}\mathbf{d}_{3ij} - \frac{\partial\mathbf{d}_{p3}^T}{\partial\boldsymbol{\alpha}}\mathbf{d}_{2ij}\right),\ j = N_{elt}\end{aligned} \tag{2.68}$$

with $\partial \mathbf{d}_{pk}^T / \partial \boldsymbol{\alpha}$ to be computed in relation to the selected platform orientation parametrization.

This derivation concludes the part related to the state-of-the-art analysis. In the following, the contributions of this thesis are highlighted, starting with the workspace computation problem.

Part II

Scientific Contributions to the Workspace Evaluation of Continuum Parallel Robots

Chapter 3

Full Workspace Computation and Numerical Results Certification

Contributions of this chapter: *this chapter proposes a methodology for the workspace evaluation of planar* CPRs *(*PCPRs*), with a focus on the constant-orientation workspace. An explorative algorithm, based on the iterative solution of the inverse geometrico-static problem, is proposed for the workspace computation of a generic* PCPR*. Thanks to an energy-based modelling strategy, and derivative approximation by finite differences, it is possible to apply the Kantorovich theorem to certify the existence, uniqueness in a neighborhood of the initial guess and convergence of the solution of the inverse geometrico-static problem at each step of the procedure. Three case studies are shown to demonstrate the effectiveness of the proposed approach. The contributions of this chapter have been published in [148].*

Workspace evaluation of *CPRs*, i.e. the identification of all poses where the robot may lie in a stable equilibrium, is a crucial performance assessment tool. Though several geometrical, discretization and numerical methods are available in the literature for rigid-link manipulators [43], workspace computation algorithms for *CRs* are at a preliminary stage. As highlighted in Sec. 1.3, actuation sampling approaches are applicable for general *CPRs*, but the computational time may be high. Task-space approaches perform better in relation to the required computational effort but frequently lack in generality. Even if most of the workspace exploration strategies are not dependent on the *CPR* modelling strategy, the latter determines the capability to evaluate several features, such as singularities and equilibrium stability.

A fundamental part of each workspace evaluation algorithm is the computation of the robot pose. In *CPRs*, this task is not trivial, as it requires the solution of geometrico-static problems of Eq. (2.21) or (2.22) usually solved by numerical schemes (e.g. Newton-based methods). However, it may happen that no or multiple solutions exist for the same problem, and it is important to avoid jumps between a solution to another during the workspace evaluation. Thus, for the workspace evaluation problem, it is important to certify the existence, uniqueness, and convergence of the solution in a neighbourhood of a given initial guess. Interval analysis (*IA*) [79],[149],[150] applies a branch-and-prune approach to solve a set of equations and to certify that all solutions are found in a bounded region. However, this approach is computationally expensive and time-consuming in comparison to non-certified approaches. Moreover, *IA* can efficiently handle a reduced number of independent

variables, which is usually not compatible with discretized continuum-robot equations when good accuracy is required. To obtain results in a reduced computational time and to be able to handle a large set of robot variables, the Kantorovich theorem [151] can be applied: this theorem establishes conditions on the initial guess of the Newton iteration to certify the existence, uniqueness, and convergence of the solution on a defined region (see [152] as a relevant example of the application of the Kantorovich theorem).

In this chapter, a general approach for the computation of the constant-orientation workspace of planar continuum parallel robots (*PCPRs*), a class of *CPRs* which recently obtained higher scientific interest [14],[37], is proposed. An *ad hoc* adaptive workspace-exploration algorithm, based on the iterative solution of the inverse geometrico-static problem (*IGSP*), is employed to identify stable workspace regions, to discover unstable regions, and to detect singularities that determine the workspace boundaries [56]. By employing an energy-based modelling strategy combined with finite differences, it is possible to employ the Kantorovich theorem to certify the existence, uniqueness and convergence of the solution of the *IGSP*, at each step of the workspace computation algorithm. A preconditioning matrix for the Newton iteration is proposed to enlarge the regions where the *IGSP* solution is certified and to reduce the overall computational time.

The chapter is structured as follows. Section 3.1 recalls the *PCPR* modelling strategy and discusses the application of the Kantorovich theorem. Section 3.2 describes the novel workspace exploration algorithm in detail. Section 3.3 proposes three case studies to show the effectiveness of our approach. Section 3.5 draws conclusions and outlines future work directions.

3.1 Numerical certification of the *IGSP* solution

This section focuses on the *IGSP* solution and its numerical certification for planar *CPRs* only. In particular, a finite-differences approach for the discretization of the planar flexible beams (see Sec. 2.3.2 and Appendix B.2 for its implementation). This choice is mainly due to the analytical formulation of the geometrico-static problems that finite differences bring, which allows for a more straightforward computation of the terms needed for the numerical certification of the results.

This chapter focuses on *PCPRs* only: the rigid platform position is described by $\mathbf{p}_p \in \mathbb{R}^2$, and its orientation is described by the angle $\phi \in \mathbb{R}$. As previously mentioned, the focus is directed on the task-space exploration, and the goal is to solve the *IGSP* of Eq. (2.22), reported here for clarity:

$$\mathbf{F}(\mathbf{y}) = \begin{cases} \nabla_{\mathbf{x}} V_{tot} + \nabla_{\mathbf{x}} \boldsymbol{\Phi}^T \boldsymbol{\lambda} = \mathbf{0} \\ \boldsymbol{\Phi} = \mathbf{0} \\ \mathbf{p}_P - \mathbf{p}_P^d = \mathbf{0} \\ \boldsymbol{\phi} - \boldsymbol{\phi}^d = \mathbf{0} \end{cases} \tag{3.1}$$

The detailed expression of *IGSP* equations for a *PCPR* is reported in Appendix B.2. Since no analytical solution is available, the *IGSP* is solved numerically. In particular, if a Newton method is employed for the solution of the *IGSP*, the Kantorovich theorem establishes sufficient conditions for the existence and uniqueness of the solution [153] near a given initial guess $\mathbf{y}_0$. Let $\chi, \delta, \gamma \in \mathbb{R}^+$, $\mathbf{y}_0$ be the initial

guess given to the numerical solver, and $\mathcal{B}(\mathbf{y}_0, 2\delta) = \{\mathbf{y} : \|\mathbf{y} - \mathbf{y}_0\| \leqslant 2\delta\}$ a ball of radius 2δ centred in $\mathbf{y}_0$. For convenience, let the infinite norm be used. Let $\mathbf{J}(\mathbf{y})$ be the Jacobian matrix of $\mathbf{F}(\mathbf{y})$ w.r.t. $\mathbf{y}$, that is:

$$\mathbf{J}(\mathbf{y}) = \frac{\partial \mathbf{F}(\mathbf{y})}{\partial \mathbf{y}} \tag{3.2}$$

Thanks to the analytical formulation of Eq. (3.1) when using finite-differences, all the terms of $\mathbf{J}$ can be computed analytically as well. These terms are detailed in Appendix B.2. Then, three constants χ, δ, γ are chosen such as:

$$\chi \geqslant \|\mathbf{J}^{-1}(\mathbf{y}_0)\|_\infty \tag{3.3}$$

$$\delta \geqslant \|\mathbf{J}^{-1}(\mathbf{y}_0)\mathbf{F}(\mathbf{y}_0)\|_\infty \tag{3.4}$$

$$\gamma \geqslant \max_{\mathbf{y}\in\mathcal{B}} \left(\max_{h\in[1,\cdots,n+m+n_c+n_\phi]} \left(\sum_{i,j} \left| \frac{\partial^2 F_h(\mathbf{y})}{\partial y_i \partial y_j} \right| \right) \right) \tag{3.5}$$

The Kantorovich theorem states: if χ, δ, γ exist such that $2\chi\delta\gamma \leqslant 1$, a solution $\mathbf{y}^* \in \mathcal{B}(\mathbf{y}_0, t^*)$ exists, where:

$$t^* = \frac{1 - \sqrt{1 - 2\chi\delta\gamma}}{\gamma\chi} \in [\delta, 2\delta] \tag{3.6}$$

Moreover, the solution is unique inside $\mathcal{B}(\mathbf{y}_0, 2\delta)$, and the Newton iteration, starting from $\mathbf{y}_0$, converges to $\mathbf{y}^*$. Kantorovich constants have a numerical meaning: χ is related to the absolute conditioning of the problem, δ represents the closeness to the linearized solution, and γ is influenced by the non-linearity of the problem.

However, if a finite difference discretization approach is used for modelling *PCPRs*, the Jacobian matrix $\mathbf{J}$ is usually large and ill-conditioned, resulting in high values of χ. Therefore, the certification of the existence, uniqueness, and convergence of the *IGSP* solution holds only for $\mathbf{y}_0$ sufficiently close to $\mathbf{y}^*$. To overcome this difficulty, a preconditioner can be used to reduce the value of χ. First, let us consider the first Equation in the set of Eq. 3.1, where $\mathbf{x} = [\mathbf{q}_e, \mathbf{p}_p, \phi]$. In the case a finite difference approximation is used, the vector $\mathbf{q}_e$ and ϕ physically represent orientation angles, while $\mathbf{p}_p$ is a position. The physical units of the terms of Eq. (3.1), then, are J/rad and J/m (or any equivalent energy, orientation and position units).

Concerning the second equation in the set of Eq. 3.1, it is convenient to split $\mathbf{\Phi} = [\mathbf{\Phi}_p, \mathbf{\Phi}_\theta]$, where $\mathbf{\Phi}_p, \mathbf{\Phi}_\theta$ represent position and orientation constraints, respectively. While $\mathbf{\Phi}_p$ has the units as the geometrical position constraints, namely m or any equivalent position units, $\mathbf{\Phi}_\theta$ has rad or any other orientation unit. Thus, to alleviate the ill-conditioning of the problem and to reduce χ, an equivalent and modified inverse geometrico-static problem (*MIGSP*) is solved, where each *IGSP* equation is multiplied by a physical constant related to the *PCPR* geometry, structural properties, and each equation unit:

$$\mathbf{F}_M(\mathbf{y}) = \begin{cases} \mathbf{0} = \frac{1}{EI}(\nabla_{\mathbf{q}_e} V_{tot} + \nabla_{\mathbf{q}_e} \mathbf{\Phi}^T \boldsymbol{\lambda}) \\ \mathbf{0} = \frac{L}{EI}(\nabla_{\mathbf{p}_P} V_{tot} + \nabla_{\mathbf{p}_P} \mathbf{\Phi}^T \boldsymbol{\lambda}) \\ \mathbf{0} = \frac{1}{EI}(\nabla_{\phi} V_{tot} + \nabla_{\phi} \mathbf{\Phi}^T \boldsymbol{\lambda}) \\ \mathbf{0} = \frac{1}{L}\mathbf{\Phi}_p \\ \mathbf{0} = \mathbf{\Phi}_\theta \\ \mathbf{0} = \frac{1}{L}(\mathbf{p}_P - \mathbf{p}_P^d) \\ \mathbf{0} = \boldsymbol{\phi} - \boldsymbol{\phi}^d \end{cases} \tag{3.7}$$

or equivalently:

$$\mathbf{F}_M(\mathbf{y}) = \mathbf{M}\mathbf{F}(\mathbf{y}) \tag{3.8}$$

where $\mathbf{M}$ is a diagonal matrix collecting the coefficients that multiply each equation of Eq. (3.7). The Jacobian matrix of the *MIGSP* w.r.t. $\mathbf{y}$ is:

$$\mathbf{J}_M(\mathbf{y}) = \frac{\partial \mathbf{F}_M(\mathbf{y})}{\partial \mathbf{y}} = \mathbf{M}\frac{\partial \mathbf{F}(\mathbf{y})}{\partial \mathbf{y}} = \mathbf{M}\mathbf{J}(\mathbf{y}) \tag{3.9}$$

The Kantorovich constants of the *MIGSP* can be obtained from Eqs. (3.3), (3.4), (3.5) by employing $\mathbf{F}_M, \mathbf{J}_M$ instead of $\mathbf{F}, \mathbf{J}$. While the computation of χ, δ is straightforward, the computation of γ requires analytically computing all the second derivative of *MIGSP* equations, which is long and involved, and the detailed equations are reported in Appendix D. According to Eq. (3.5), it is necessary to identify where the sum of the absolute values of the second-order derivatives of $\mathbf{F}_m$ assumes the maximum value inside the ball for each equation. However, this sum is not constant w.r.t. $\mathbf{y}$, and the maximum may be identified by solving a constrained optimization problem at the cost of high computational time. Alternatively, it is possible to approximate γ with absolute value inequalities [145]: in this way, the worst-case scenario is considered and an expression of γ it is obtained that depends on $\mathbf{y}_0, \delta$, separately. The details of this computation are reported in Appendix D.

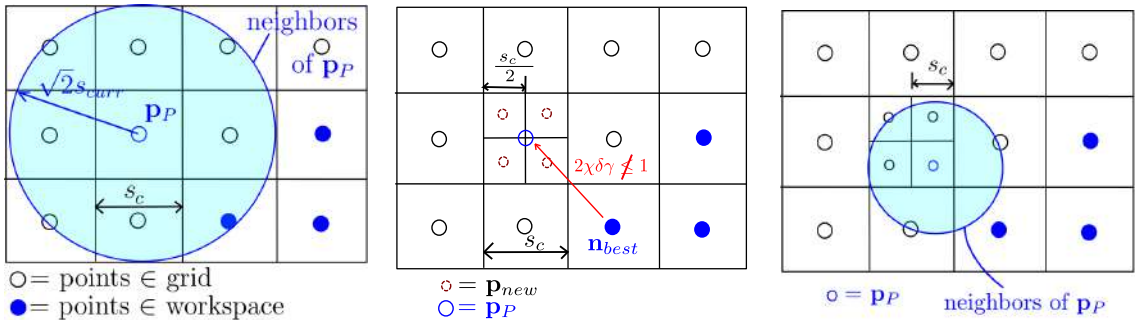
It is now important to stress why discretized robot equations are necessary for the computation of γ in Eq. (3.5), and why continuous approaches for the *CPR* geometrico-static problem solution do not fit well for this scope. According to the continuous formulation, the second derivatives of the robot equilibrium equations are analytically computed since they can be computed by employing a derivative propagation approach (as described in [75]). However, their numerical values are obtained by integrating the underlying differential equations, and it is not trivial to establish where the quantity

$$\sum_{i,j} \left| \frac{\partial^2 F_h(\mathbf{y})}{\partial y_i \partial y_j} \right| \tag{3.10}$$

assume the maximum value in $\mathcal{B}(\mathbf{y}_0, 2\delta)$, as required by Eq. (3.5) without the numerical solution of an optimization problem. Therefore, discretized robot equations are preferred because second derivatives of robot equations can be computed explicitly and the identification of the maximum inside $\mathcal{B}(\mathbf{y}_0, 2\delta)$ can be addressed by employing absolute value inequalities and trigonometric functions.

3.2 Full Workspace Evaluation

This Section describes in detail the innovative workspace-computation strategy, namely, the Adaptive Flooding Algorithm (Alg. 1 lines 1-19). In particular, the proposed algorithm is a modification of the Flooding Algorithm of [128] by introducing a grid-adaptation procedure: during each iteration, the grid size is adjusted when the certification is not feasible with the initial grid size. This is done in order to certify as much workspace as possible since the Kantorovich constant δ depends on the distance of the initial guess from the solution. The attention is restricted to the constant-orientation workspace of *PCPRs*, i.e. the set of all possible locations of the robot *EE* that can be reached with a given orientation, though our approach



(a) (b) (c)

Figure 3.1: Workspace algorithm: (a) representation of the grid, points along the grid and condition for the neighbourhood, (b) grid refinement strategy; (c) situation when neighbour radius should be enlarged.

can be easily extended to other types of workspace. A discretization approach is employed, and the *MIGSP* is solved iteratively over a grid of *EE* locations.

The first step of the algorithm requires the initialization of some entities: a 2-dimensional uniform grid is generated accordingly to an initial stepsize (s_i). The grid discretizes a user-defined box (assumed to fully include the workspace), and in the middle of each square of the grid, a point representing the *EE* position is placed (Fig. 3.1a). An initial point (named $\mathbf{p}_P$) where the explorative algorithm starts is selected within the grid. For the first solution of the *MIGSP*, an initial guess obtained through a constant-curvature modelling approach is employed. As shown in [37], the inverse problem under the constant-curvature approach admits a finite number of solutions (but usually low solution accuracy), and the solution to be used as an initial guess is decided in order to let the algorithm converge to the desired working mode [154]. The resulting output configuration is stacked in the *Config* list to be employed as a future guess. Then, neighbouring *EE* positions (Fig. 3.1a) are computed and saved in the *toDo* list, which contains points to be processed. The *toDoEnd* list, which contains points that are neighbours to Type 2 singular configurations, is initialized as empty. Then, the iterative algorithm starts, and a new $\mathbf{p}_P$ is obtained from the first element of the *toDo* list and employed as the *EE* location to be reached. The initial guess $\mathbf{y}_0$ for the *MIGSP*, the Kantorovich flag associated to $\mathbf{y}_0$ and neighbors of $\mathbf{p}_P$ are obtained through the *FindGuess* procedure. The latter, which plays an important role in the certification of the *MIGSP* solution, is detailed in subsection 3.2.1.

If the current step s_c (i.e. the edge size of the square centred in $\mathbf{p}_P$) is greater to the minimum stepsize allowed for the computation (s_m), the value of the Kantorovich constants is checked to certify the solution: if $2\chi\delta\gamma \leqslant 1$, the *MIGSP* is solved and new points to be processed are created in the *Compute* procedure (detailed in subsection 3.2.2). If the manipulator is not cuspidal [155], and the solution is certified, the working mode of the initial guess is also preserved on the configuration resulting from the *MIGSP* solution. In the case of the solution is not certifiable, the grid is refined: the square centred in $\mathbf{p}_P$ is divided into four new equal squares (Fig. 3.1b), and four new points $\mathbf{p}_{new}$ are placed in the middle of each new square. Then, the original *EE* location $\mathbf{p}_P$ is replaced with $\mathbf{p}_{new}$ and added to the *toDo* list. In the case, the minimum stepsize is reached ($s_c \leqslant s_m$), the *Compute* procedure is

Algorithm 1: Adaptive flooding algorithm.

```
1  Initialize grid, toDo, toDoEnd, Config, Results;
2  while toDo≠ ∅ or toDoEnd≠ ∅ do
3  |  if toDo= ∅ then
4  |  |  toDo ← toDoEnd; toDoEnd ← ∅ ;
5  |  end
6  |  p_P = toDo(1), toDo ← toDo\p_P;
7  |  [flag_k,y_0,n] = FindGuess(p_P, s_c);
8  |  if s_c > s_m then
9  |  |  if flag_k ⩽ 1 then
10 |  |  |  Compute(y_0,n);
11 |  |  else
12 |  |  |  p_new = Generate points(p_P,s_c);
13 |  |  |  Replace p_P with p_new;
14 |  |  |  toDo ← toDo ∪ p_new
15 |  |  end
16 |  else
17 |  |  Compute(y_0,n);
18 |  end
19 end
```

executed even if the *MIGSP* solution is not certified in order to fully compute the workspace. Finally, the algorithm restarts until some elements are present in the lists *toDo* and *toDoEnd*. If *toDo* is empty, then it is refilled with *toDoEnd* (how *toDo* and *toDoEnd* are managed is explained in subsection 3.2.2).

3.2.1 Choice of the Initial Guess

The choice of the initial guess at each iteration plays an important role in the *MIGSP* solution certification. The routine of the initial guess selection is described in lines 1-13 of Alg. 2. Given a desired *EE* location $\mathbf{p}_P$, it is necessary to identify an initial-guess configuration $\mathbf{y}_0$ to be used for the *MIGSP* solution. Initially, the distance r_N for which two *EE* locations are considered to be neighbours is set as $\sqrt{2}s_c$ (Fig. 3.1a). This way, neighbours of $\mathbf{p}_P$ are identified in the grid and stacked in the array $\mathbf{n}$. Neighbours in the workspace are extracted from $\mathbf{n}$ and collected in the array $\mathbf{n}_{WK}$. However, caused by the grid refinement process, it can happen that no neighbours are in the workspace (Fig. 3.1c): in this case, the radius r_N for which points are considered to be neighbours is multiplied by two, and the selection of $\mathbf{n}$ is repeated until workspace points are found.

In order to increase the possibility of certifying the *MIGSP* solution, the Kantorovich constants for the configurations associated with $\mathbf{n}_{WK}$ *EE* locations are computed. Then, the *EE* location that ensures the lowest value of $2\chi\delta\gamma$ (named $\mathbf{n}_{best}$) is identified, and the robot configuration $\mathbf{y}_0$ associated to $\mathbf{n}_{best}$ is extracted from *Config*. If the solution certification is not required, $\mathbf{n}_{best}$ can be chosen as the one that ensures the best inverse conditioning of $\mathbf{J}$ to speed up the computation.

Algorithm 2: Ausiliary functions for the adaptive flooding algorithm.

```
1  Function FindGuess(p_P, s_c):
2      Set neighbors radius r_N = √2 s_c;
3      do
4          n = Find neighbors EE positions to p_P ∈ grid;
5          n_WK = n ∈WK;
6          if (n_WK = ∅) then
7              r_N ← 2 r_N;
8          end
9      while (n_WK = ∅);
10     Compute Kantorovich flag for each n_WK;
11     n_best = n_WK with best Kantorovich flag;
12     y_0 = Config(n_best);
13     return [flag_k, y_0, n];
14 Function Compute(y_0,n):
15     y = Solve IGSP starting from y_0;
16     [T_1, T_2] = Singularity(y);
17     if (Solver Converged & T_1 < TOL & mechconstr(y)) then
18         Values = CalculateOutputs(y);
19         Save Values in Results, Save y in Config;
20         n_1 = select n ∉(WK,toDo,toDoEnd);
21         if T_2 < TOL then
22             toDo ← toDo ∪ n_1;
23         else
24             toDoEnd ← toDoEnd ∪ n_1;
25         end
26     end
27     return;
```

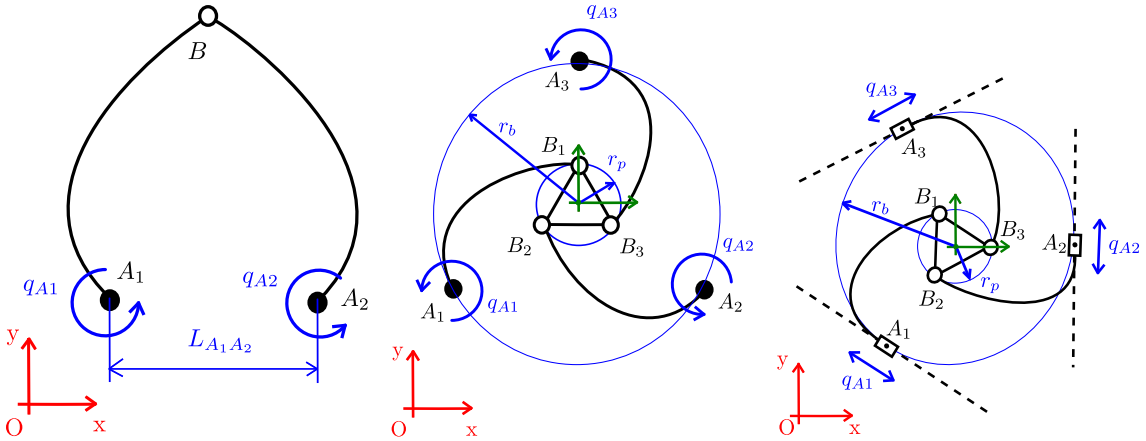
3.2.2 Computation Process

This subsection describes the routine for the computation of the *MIGSP* solution and for the creation of new points to be processed (lines 14-27 of Alg. 2). Starting from a given initial guess $\mathbf{y}_0$ and a set of neighbouring *EE* locations $\mathbf{n}$, the *MIGSP* is solved by a Newton scheme. Then, $\mathbf{T}_1, \mathbf{T}_2$ are computed according to Eq. (2.31), (2.32). In particular, the inverse condition number of $\mathbf{T}_1, \mathbf{T}_2$ is used in order to identify their degeneracy. Then, it is verified if the Newton solver converges and the resulting configuration is not Type-1 singular. Moreover, mechanical constraints are verified in *mechconstr*: these include strain limits on the legs, as well as joint limits. If the check succeeded, outputs associated with the resulting configuration $\mathbf{y}$ (e.g. internal energy of the robot, equilibrium stability, number of inflexion points) are computed in *Values*. These results, as well as singularity flags and Kantorovich constants, are saved in *Results* and $\mathbf{y}$ is stored in *Config* as a future initial guess. Subsequently, neighbours not in the workspace and not in *toDo*,*toDoEnd*, are stored in $\mathbf{n}_1$. If the actual configuration $\mathbf{y}$ is not T_2-singular, $\mathbf{n}_1$ is added to the *toDo* list, else in *toDoEnd*.

In this way, Type 1 singularities, associated with boundaries of the workspace, are not crossed but only approached. Type 2 singularities, which delimit stable from unstable regions, are crossed in a second stage of the algorithm (only when *toDo* is

empty) in order to discover possible stable regions separated by unstable transitions.

3.3 Case Studies



(a) $\underline{R}FRF\underline{R}$ (b) $3-\underline{R}FR$ (c) $3-\underline{P}FR$

Figure 3.2: Three *PCPRs* object of our case studies: the $\underline{R}FRF\underline{R}$ (a), the $3-\underline{R}FR$ (b), and the $3-\underline{P}FR$ (c). Relevant design dimensions are displayed.

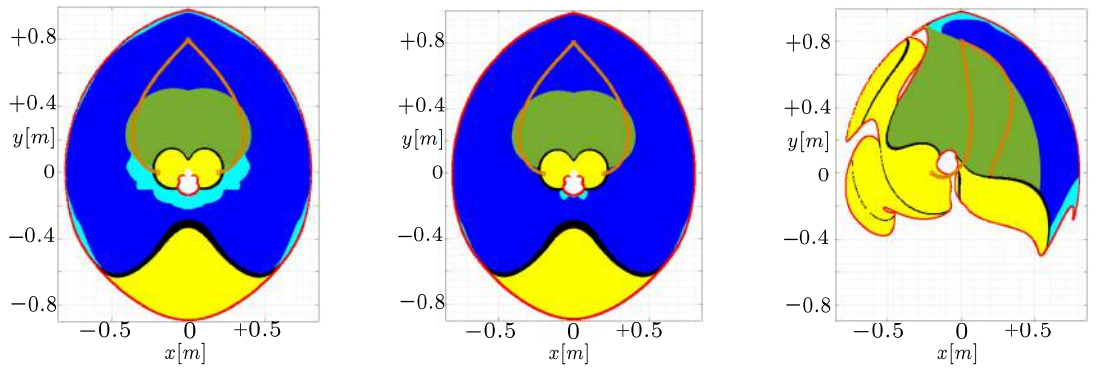
In this Section, three case studies focusing on different *PCPRs* (Fig. 3.2), are reported. Workspace evaluation is discussed with a focus on the identification of different regions (e.g. stable, unstable, regions where stress limits are exceeded) and certification of the *IGSP* solution. The algorithm is compared to the flooding algorithm of [56], to demonstrate the benefit of the grid-adaptation routine in terms of computational time. For all case studies, beams are made of harmonic steel with Young modulus $E = 210$ GPa, maximum stress $\sigma_{max} = 1800$ MPa, density $\rho = 7800$ kg/m^3, length $L = 1$m, circular cross-section of radius $r = 1$mm. Simulations are performed in the Matlab environment.

3.3.1 RFRFR robot

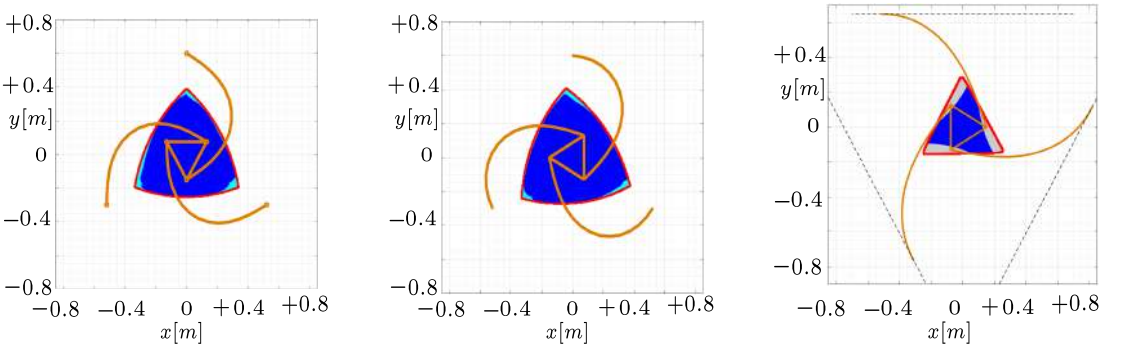
This subsection investigates the workspace of a $\underline{R}FRF\underline{R}$ robot. The aim of this case study is twofold: on the one hand, it shows the capability of the algorithm to detect singularities and unstable regions, as well as to include external loads and strain limits in the model; on the other hand, it investigates the influence of the stepsize and the preconditioner on the *IGSP* solution certification.

This manipulator, borrowed from [78],[80], has two actuated revolute joints in A_1, A_2 ($\underline{R}$) and two flexible links (F) connected by a passive revolute joint centered in B (Fig. 3.2a). The distance between the actuators is $L_{A_1A_2} = 0.4$m. An external force of 1.5N is applied on the *EE*, and legs are subjected to gravity. Simulations are performed with $N = 50$, ensuring sufficient *MIGSP* solution accuracy.

The workspace of the $\underline{R}FRF\underline{R}$, computed by the algorithm presented in Section 3.2, is shown in Figs. 3.3a, 3.3b, 3.3c. Configurations are marked as singular when the inverse condition number of matrices reported in Eq. (2.31), (2.32) is lower than a certain threshold TOL. Practically, $TOL = 10^{-6}$ correctly identify singularities when a finite-differences approximation is used. Stress limits are considered by



(a) RFRFR: working mode 1, $s_m = 2$mm (b) RFRFR: working mode 1, $s_m = 1$mm (c) RFRFR: working mode 2, $s_m = 1$mm



(d) 3-RFR: $\phi = -\frac{\pi}{3}$, $s_m = 1$mm (e) 3-RFR: $\phi = -\frac{\pi}{6}$, $s_m = 1$mm (f) 3-PFR:$\phi = +\frac{\pi}{6}$, $s_m = 1$mm

Figure 3.3: Workspaces of *PCPRs*. Type 1 and Type 2 singularities are drawn in red and black, respectively. Certified workspace is depicted in blue, non-certified workspace in light blue and non-certified unstable regions in yellow. Non-certified stable regions where stress limits or joint limits are exceeded are represented in green and grey, respectively.

evaluating whether a first-order approximation of stress σ [143] on each element does not overcome σ_{max}, that is:

$$\sigma = E\epsilon_z \leqslant \sigma_{max} \tag{3.11}$$

where ϵ_z is given by Eq. (2.25). By considering stress limits, the workspace is considerably reduced, in particular in the working mode showed in Fig. 3.3c (0.96 m^2 without considering stress limits to 0.36 m^2 with stress limit included).

Then, the influence of the minimum stepsize s_m and the influence of the preconditioner on the *MIGSP* certification of the workspace is tested, with a focus on the stable and feasible workspace (i.e where stress limits or joint bounds are not exceeded). To quantify how many configurations are computed in a certified way, the certified percentage of the workspace $C_{\%}$ is introduced as:

$$C_{\%} = 100 \cdot A_C / A_W \tag{3.12}$$

where A_C is the certified area and A_W is the workspace area. These areas are obtained by summing the area of each square of the grid that lies in the workspace (for A_W) and by summing the area of each square that is computed in a certified way and belongs to the workspace for A_C. Results are reported in Table 3.1. With

$\underline{R}FRF\underline{R}$						
s_m	With preconditioning			Without preconditioning		
	Certified [%]	Total Time [min]	N° points	Certified [%]	Total Time [min]	N° points
4	33.8	11	$1.3 \cdot 10^5$	0	13	$1.3 \cdot 10^5$
2	80.4	32	$4.0 \cdot 10^5$	5.5	80	$5.4 \cdot 10^5$
1	92.1	84	$7.1 \cdot 10^5$	35.1	383	$18.9 \cdot 10^5$

Table 3.1: Influence of the preconditioner and the stepsize on the certification of the workspace. Simulations, relative to the workspace displayed in Fig. 3.3a, 3.3b, are performed with $s_i = 4$mm, and $N = 50$.

$s_i = 4$mm, by passing from $s_m = 2$mm (Fig. 3.3a) to $s_m = 1$mm (Fig. 3.3b), $C_\%$ grows from 80.4% to 92.1%. As expected, the computational time[1] increases from 32 min to 84 min. To achieve $C_\% = 92.1\%$, the flooding algorithm of [56] required 192 min with 1 mm stepsize, which is considerably higher. If the preconditioner is not used, $C_\%$ drops to 35.1% (with $s_m = 1$mm) and at the same time, the computational time reaches 383 min. The increase of the computational time is due to the higher number of processed points ($18.9 \cdot 10^5$ compared to $7.1 \cdot 10^5$ in the preconditioned case) for the adaptation of the grid and not to a considerable increase in the *MIGSP* solution time. This is confirmed by the data relative to the case of $s_m = s_i = 4$mm (where no grid refinement is possible), with the same grid being employed with and without the preconditioner, and the resulting computational times are comparable. Also in the case of a different working mode (Fig. 3.3c), a significant amount of the workspace (70.2%) is certified in a reduced time (44 min) with $s_m = 1$mm.

3.3.2 3-RFR robot

This subsection investigates the workspace of the $3-\underline{R}FR$ robot. This case study shows the possibility of identifying the workspace and certifying the *IGSP* solution with different *EE* orientations. This manipulator borrowed from [37] has three actuated revolute joints A_1, A_2, A_3 ($\underline{R}$) and three flexible links (F) connected by passive revolute joints B_1, B_2, B_3 at a rigid *EE*. Actuators are equally placed along a circumference of radius $r_b = 0.6$m, whereas passive revolute joints are placed on a circumference of radius $r_p = 0.15$m (Fig. 3.2b). Simulations are performed with $N = 30$, and no external loads are included.

The workspace of the $3-\underline{R}FR$ robot is illustrated in Fig. 3.3d and 3.3e by fixing $\phi = -\frac{\pi}{3}$, $\phi = -\frac{\pi}{6}$, respectively. In both cases, no unstable regions and Type-2 singularities are detected. Stress limits are included, but no point exceeds σ_{max}. As before, $s_m = 1$mm is chosen to guarantee a sufficient value of $C_\%$ (81% and 84.6%) in a reasonable computational time (11 and 10 min). Again, the flooding algorithm of [56] required higher computational time to obtain the same $C_\%$ (20 and 18.5 min with 1 mm stepsize).

[1]Results are obtained by a CPU Intel Core i7-8700K,3.7GHz,32Gb RAM

3.3.3 3-PFR robot

This subsection studies the workspace of the $3-\underline{P}FR$ robot. This case study shows the possibility of analyzing *PCPRs* with different actuators and including joint limits. This manipulator is similar to the one proposed in [14], except for the connection of the flexible links with the platform (passive revolute joints in our case, in contrast with fixed connections in [14]). The $3-\underline{P}FR$ robot has three actuated prismatic joints A_1, A_2, A_3 ($\underline{P}$) and three flexible links (F) connected by passive revolute joints B_1, B_2, B_3 at a rigid EE. Actuators are equally spaced along a circumference of radius $r_b = 0.65$m, and passive revolute joints are placed on the platform of radius $r_p = 0.15$m (Fig. 3.2c). Simulations are performed with $N = 30$, and no external loads are included.

The workspace of the $3-\underline{P}FR$ robot is illustrated in Fig. 3.3f, where the EE orientation is $\phi = \frac{\pi}{6}$ and 1.4m-long rails are symmetrically placed around a circle of radius r_b. As for the $3-\underline{R}FR$ robot, maximum strain limits are included but not exceeded. With $s_m = 1$ mm, we reached $C_\% = 94.5\%$ in 5.2 min of computational time, whereas the algorithm of [56] required 17 min with 1 mm stepsize.

3.4 Discussion on the Extension to Spatial *CPRs*

The work presented in this chapter is limited to planar *CPRs* only workspaces of dimension two. However, there is no theoretical limitation that precludes the extension of this work to spatial *CPRs*:

- the modelling strategy, based on a finite-differences approximation, could be equivalently used also for spatial *CPRs* (see Appendix B.1 for the details);
- the Kantorovich theorem can be applied in the same fashion, since the geometrico-static model equations assumes the same formulation. The derivation of the constants χ, δ, γ is performed in the same manner as described in Appendix D;
- thus, the adaptive flooding algorithm can be easily extended to the spatial case. A three-dimensional grid can be generated, neighborhood analysis, and grid bisection can be performed as well.

However, preliminary author's investigations have shown that, by using a finite-differences approximation, results are not as good as in the planar case. To achieve sufficient EE position accuracy, a large number of elements is required ($\geqslant 200$ per *CPR* leg). Consequently, the *IGSP* system of equations is large and ill-conditioned, resulting is high values of χ. Therefore, the certification of the results hold only for significantly dense grids (grid stepsize grealy lower than 1mm), and the computational time increases. Under these perspectives, other approaches (e.g. inteval analysis) become competitive. A possible solution would require to investigate other modelling strategies that allows the analytical computation of derivatives, while keeping reduced the number of variables requires to accurately represent the *CPR* pose, such as differential quadrature methods [145].

3.5 Conclusions

This chapter presents an adaptive flooding algorithm for the workspace computation of *PCPRs*. The algorithm may identify unstable regions, and singularity loci, incorporate external loads, and set maximum stress limits and joint bounds. Thanks to an energy-based modelling strategy approximated through finite differences for derivatives, the *IGSP* solution was certified in terms of existence, uniqueness, and convergence of the solution by verifying Kantorovich conditions during the Newton-based problem-solving procedure. With this approach, the *IGSP* solution is certified over a large percentage of the workspace in a reduced computational time in comparison with previous algorithms, also in the case of external loads being included.

However, with large workspaces and/or small stepsizes, the flooding approach may require the computation of a large number of points, which may not be computationally efficient. This problem is further relevant in the case of three-dimensional workspaces. Preliminary investigations demonstrated that the resulting computational time for the full workspace computation in three-dimensional domains is considerably higher than many hours. This motivates the work of the next chapter, which is directed toward the computation of the workspace boundaries only.

Chapter 4

Boundary workspace computation algorithm

Contributions of this chapter: *a new algorithm for the computation of workspace boundaries of continuum parallel robots (*CPRs*) is the contribution of this chapter. The proposed approach for the computation of the workspace boundaries is based on i) a free-space exploration strategy and ii) a boundary reconstruction algorithm. The former is exploited to identify an initial workspace boundary location (exterior, interior boundaries, and holes), while the latter is used to reconstruct the complete boundary surface. Moreover, the algorithm is designed to be employed with* CPRs *modelling strategies based on general discretization assumptions in order to increase its applicability for various scopes. The proposed method is compared with two state-of-the-art algorithms in four case studies to validate the results and to establish its merits and limitations. The results of this chapter have been published in [156]*

Full workspace computation algorithms provide great applicability at the cost of high computational time. In particular, if an actuation sampling strategy is employed, the computational cost explodes with an increasing number of actuators. In the previous chapter, an algorithm for the full workspace computation of planar *CPRs* was proposed. Its application to three-dimensional task-spaces is possible, but at the cost of high computational time. For practical purposes (e.g. design iterations), it is desired to reduce the computational time of workspace computation algorithms, and a popular approach is to compute the workspace boundaries only. As illustrated in Sec. 1.3.2, boundary workspace computation algorithms offer better computational performances than full workspace algorithms but are strongly influenced by the selected modelling strategy, which affects not only the performance of the algorithm but also their applicability. Few works were directed toward the workspace boundary computation, and none of them was dedicated to *CPRs*.

For these reasons, a boundary workspace computation algorithm for *CPRs* is proposed in this chapter. The aim is reducing the computational time w.r.t. full workspace computation algorithms while preserving their general applicability. The proposed algorithm is suitable for the boundary computation of any type of planar *CPRs* workspace, but only for translational (i.e. constant orientation) and orientation (i.e. constant position) workspace of spatial *CPRs* [43], since it is based on a

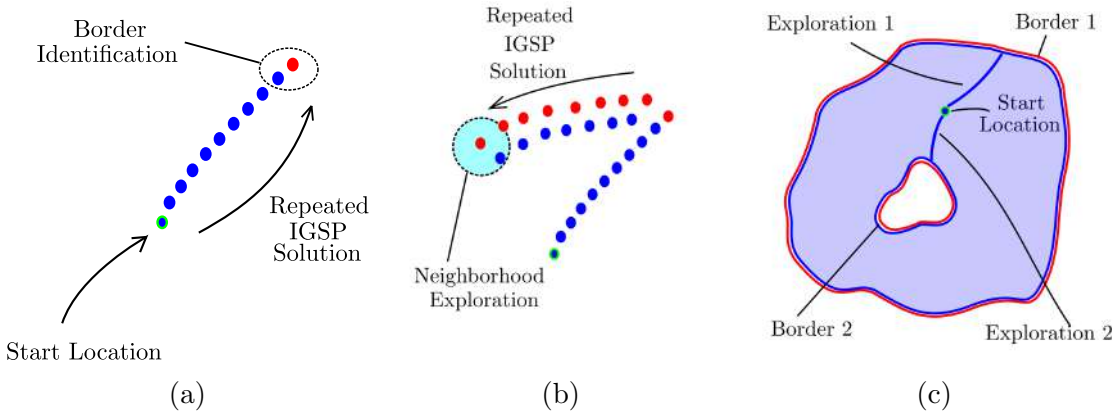


Figure 4.1: General overview of the *BFA*. (a) initial grid exploration, (b) boundary computation, and (c) successive iteration of the algorithm. Points on the task space that lie in the workspace are represented in blue, while points outside of the workspace in red.

three-dimensional grid exploration[1]. To do this, i) a suitable exploration strategy for identifying all borders and ii) a boundary reconstruction algorithm is proposed. The exploration strategy is introduced in order to identify a first boundary location, starting from a guess initial point. The exploration strategy is repeated several times in different directions, depending on the location of previously identified border locations. This way, not only exterior borders but also interior borders, voids or holes (that may occur within the *CPRs* workspace) can be identified efficiently. Then, a boundary reconstruction algorithm is proposed to compute the boundary surface: the border is identified over a fixed grid by exploring the neighbourhood of previously identified border points, with the exploration being based on the solution of the *IGSP* over several *EE* locations. In comparison to the state of the art, this algorithm requires reduced computational time w.r.t. full workspace computation algorithms while preserving general applicability.

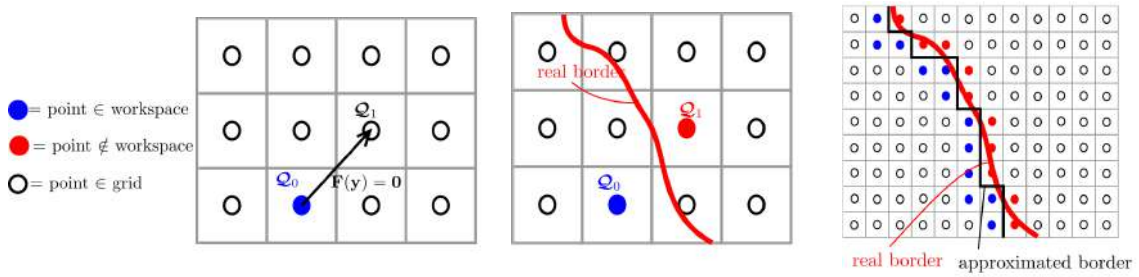
The chapter is structured as follows. Section 4.1 describes in detail the proposed workspace computation algorithm. In Section 4.2, the proposed approach is applied to four different *CPRs* and compared with state-of-the-art workspace computation algorithms. Section 4.3 draws conclusions and discusses the limitations of the proposed works.

4.1 Boundary Flooding Algorithm

This Section describes the workspace computation methodology introduced in this paper, called the boundary flooding algorithm (*BFA*). First, a general overview of *BFA* is given to ease the reader's comprehension. Then, boundary identification is discussed in Sec. 4.1.1, and details of the algorithm are given in Sec. 4.1.2.

Similarly to conventional discretization approaches (e.g. [14]), our algorithm is a task-space discretization algorithm, and a grid of dimension at most three is

[1]Reachable, total orientation and dextrous workspace would require six-dimensional grid explorations. Three-dimensional task spaces only are explored because i) this covers the most frequent cases of interest in parallel robots, ii) the possibility of a graphical representation of the results, and iii) to keep the computational cost of the algorithm reasonable.



(a) (b) (c)

Figure 4.2: Boundary identification over a discrete grid. (a) *IGSP* solution over the grid, (b) border identification, and (c) border reconstruction.

generated to discretize the task-space of the robot. The *IGSP* of *CPRs* is solved repeatedly over the grid, but not on all its locations since the goal of the algorithm is to reconstruct the workspace boundaries only. Task spaces of dimensions at most three are explored, thus excluding reachable, total orientation and dextrous workspace of spatial *CPRs*, because i) this covers the most frequent cases of interest in parallel manipulators, ii) the possibility of graphical visualization of the results, iii) to keep the computational cost of the algorithms reasonable, and iv) to compare with state-of-the-art results. The author believes that a possible extension of the *BFA* to six-dimensional cases may be developed, but this is not considered in this thesis. The algorithm is designed as a two-sequential-stage process:

1. starting from a known location, the grid is explored by repeatedly solving the *IGSP* over different points until a border point of the workspace is reached, as shown in Fig. 4.1a. This stage is described in detail in Sec. 4.1.2.1;

2. then, the border is computed thanks to a flooding algorithm specifically designed for this scope. This algorithm is based on the repeated solution of the *IGSP* (Fig. 4.1b), and the determination of new points to explore in the neighbourhood of previously identified borders. This stage is illustrated in Sec. 4.1.2.2

These two stages can be repeated several times: successive exploration directions are pointed toward different regions of the grid to possibly identify holes and voids that the *CPRs* workspace may possess (Fig. 4.1c). The exploration strategy takes advantage of previously detected borders to scan only regions on the grid where no borders were identified.

4.1.1 Boundary Identification

A crucial point of the *BFA* is how boundaries are identified. As shown in Fig. 4.2a, points are placed at the centre of each box of the grid. Then, the point $\boldsymbol{\mathcal{Q}}_0$ is assumed to lie in the workspace, and the goal is to verify whether a neighbour point $\boldsymbol{\mathcal{Q}}_1$ is included in the workspace as well. To do that, being $\mathbf{y}_0$ the robot configuration at $\boldsymbol{\mathcal{Q}}_0$, the *IGSP* is solved with initial guess $\mathbf{y}_0$, and the resulting configuration $\mathbf{y}_1$ is obtained. Then, it is verified if $\boldsymbol{\mathcal{Q}}_1$ lies within the workspace: this is done by verifying:

Algorithm 3: Space Exploration Strategy.

```
1  Function Space Exploration(Q_Ak, B, Q_start):
2      Q_init = Q_start;
3      while flag = true do
4          Q_end = getNewPoint(Q_init, Q_Ak, B, iter);
5          flag = GetWKconditions(Q_init, Q_end);
6          if flag = true then
7              Save Results;
8              iter = iter + 1;
9              Q_init = Q_end;
10         end
11     end
12     Stack [Q_init, Q_end] ∈ toDo;
13     Stack Q_end in B;
14     return
15 Function GetWKConditions(Q_0, Q_1):
16     y_0 = robot configuration at Q_0;
17     y = Solve IGSP starting from y_0;
18     flag = evaluate if Q_1 ∈ WK;
19     return flag;
```

1. singularity conditions, identified by the conditions reported in Eq. (2.31), (2.32);
2. equilibrium stability conditions, verified thanks to Eq. (2.33);
3. strain limits, checked by ensuring that each point over the beams do not exceed prescribed strain limits. Strains are computed by following Eq. (2.25);
4. actuator limits, by checking motors actions satisfies prescribed limits, in terms of actuation values q_{ai}, and actuation torques τ_i. Actuation torques are computed in accordance with Eq. (2.26).

Other conditions may delimit the robot workspace, and general inequalities (and equalities) can be considered, but the analysis is limited to the aforementioned criteria.

Then, if the configuration $\mathbf{y}_1$ violates one of the aforementioned conditions, the boundary of the workspace is crossed, and $\boldsymbol{\mathcal{Q}}_1$ is considered as an out-of-the-workspace configuration. Since the task space is discretized with a grid, the real boundary is placed in a location between $\boldsymbol{\mathcal{Q}}_0$ and $\boldsymbol{\mathcal{Q}}_1$, as represented in Fig. 4.2b. Thus, the boundary can be approximately reconstructed as shown in Fig. 4.2c, and the accuracy of the workspace boundary estimation depends on the grid sampling size.

4.1.2 Detailed Description

In this section, details of workspace computation algorithm objectives are given. Sec. 4.1.2.1 first illustrates the space exploration strategy, and Sec. 4.1.2.2 proposes the boundary computation strategy.

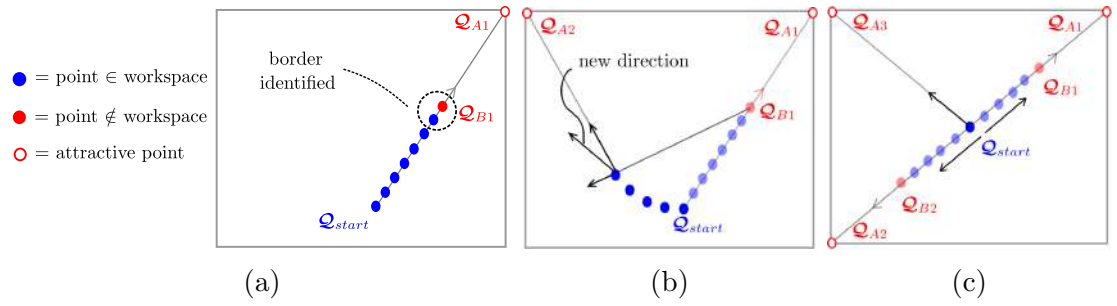


Figure 4.3: Exploration strategy: (a) first exploration, (b) subsequent exploration and influence of previously computed points, (c) conditions for which an equilibrium is reached without attractive points.

4.1.2.1 Space Exploration Strategy

The main goal of the space exploration strategy (Alg. 3) is to explore the grid and to identify a point that lies on the workspace boundary. As previously mentioned, the exploration attempts to scan unexplored grid regions by considering where previous explorations were directed. In order to explain how the exploration directions are obtained, two different concepts are introduced: the unitary space and the attractive points.

- *Unitary Space*: to explore task spaces that may involve positions and orientations at the same time, the computation of the exploration direction is performed on a unitary space (Fig. 4.4), that is, a space with $[0, 1]$ limits for each coordinate, which is algebraically similar to the Euclidean space. Given a generic point in the task-space $\boldsymbol{\mathcal{Q}}$, its similarity transformation $\mathcal{S}$ into the unitary space point $\boldsymbol{\mathcal{H}}$ is given by:

$$\boldsymbol{\mathcal{H}} = \mathcal{S}(\boldsymbol{\mathcal{Q}}) \tag{4.1}$$

  As an example, being $\boldsymbol{\mathcal{Q}} = [x, y, z]$, and $x_{lim} = [a, b]$, $y_{lim} = [c, d]$, $z_{lim} = [e, f]$ the task-space limits, the point $\boldsymbol{\mathcal{H}}$ is obtained as:

$$\boldsymbol{\mathcal{H}} = \left(\frac{x-a}{b-a}, \frac{y-c}{d-c}, \frac{z-e}{e-f} \right)^T \tag{4.2}$$

- *Attractive Points*: these points are introduced in order to define exploration directions, and they are positioned at the limits of the grid. These grid limits are usually placed at regions not reachable by the manipulator (e.g. at a distance greater than the robot leg lengths), and attractive points are placed uniformly over the grid perimeter. Being n_{exp} the total number of explorations, $\boldsymbol{\mathcal{A}} = [\boldsymbol{\mathcal{Q}}_{A1}, \cdots, \boldsymbol{\mathcal{Q}}_{An_{exp}}]$ collects the attractive points.

After unitary space and attractive points are introduced, let us explain how the exploration works. To do that, let us consider a task-point $\boldsymbol{\mathcal{Q}}_{start}$ that lies in the workspace where the exploration starts[2]. The algorithm seeks to solve the

[2]The identification of the first point is not trivial, since *IGSP* Eqs. (2.22) are nonlinear. An efficient heuristic employs constant curvature assumptions [37], where the inverse problem admits simple (and possibly not accurate) purely geometric solutions. These constant curvature solutions are employed as initial guesses for the first solution of Eqs. (2.22).

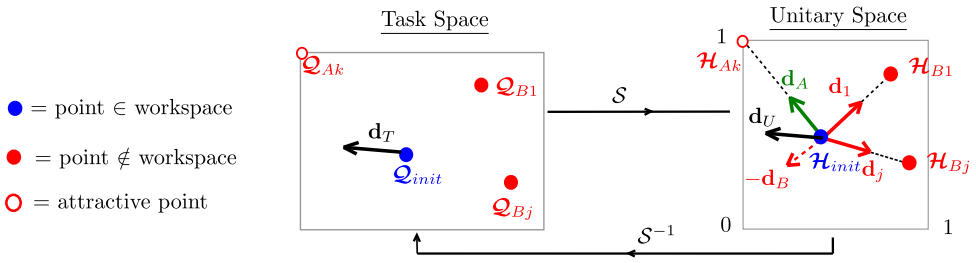


Figure 4.4: Space exploration strategy. On the left, points to the physical space. In the middle, points are converted into the unitary space. On the right computation of the new exploration direction.

IGSP over different grid locations until a border of the workspace is reached. Thus, two operations are performed sequentially: i) the solution of the *IGSP* and ii) the computation of a new task-space point where to solve the *IGSP*. The latter is the key point of the exploration strategy: at each step, it is required to find an exploration direction on the task space that defines the new *IGSP* point by taking into account the previous explorations (not to explore the same grid regions).

During the first exploration ($k = 1$, with $k = 1, \cdots, n_{exp}$ the index representing the exploration number), the grid is explored by solving the *IGSP* over sequential new locations in the direction of $\boldsymbol{\mathcal{Q}}_{A1}$ until a border is identified (Fig. 4.3a). Being $\boldsymbol{\mathcal{Q}}_{B1}$ the point where the border is identified, the exploration is then stopped, and $\boldsymbol{\mathcal{Q}}_{B1}$ is stored in $\boldsymbol{\mathcal{B}}$, which is the set of all the border points. The number of points stored in $\boldsymbol{\mathcal{B}}$ is named N_b. Then, for the second exploration, it is consider $\boldsymbol{\mathcal{Q}}_{A2}$ and the exploration restart from $\boldsymbol{\mathcal{Q}}_{start}$. However, it is important to also consider the influence of $\boldsymbol{\mathcal{Q}}_{B1}$, and the exploration should also be directed in a direction opposite to $\boldsymbol{\mathcal{Q}}_{B1}$ to explore a different region than the previous one. Therefore, the exploration direction is obtained as a combination of the direction that points toward $\boldsymbol{\mathcal{Q}}_{A2}$ and the direction opposite to $\boldsymbol{\mathcal{Q}}_{B1}$, as qualitatively represented in Fig. 4.3b.

To obtain the mathematical expression of the exploration direction, let us consider the generic k-th exploration, where $\boldsymbol{\mathcal{B}} = [\boldsymbol{\mathcal{Q}}_{B1}, \cdots, \boldsymbol{\mathcal{Q}}_{B(k-1)}]$ collects out-of-the-workspace point locations identified in the previous $k-1$ iterations. During the k-th exploration, $\boldsymbol{\mathcal{Q}}_{init}$ represents the current workspace point, and it is necessary to select a new point in its neighbourhood where to solve the *IGSP*, with initial guess $\boldsymbol{\mathcal{Q}}_{init}$. Thus, a direction $\mathbf{d}_T$ is to be identified in the task space to be used to select the next *IGSP* point. To do that, the unitary space is employed: all the points of interest in the task space are mapped into the unitary space by employing the transformation $\mathcal{S}$, and $\boldsymbol{\mathcal{H}}_{init}, \boldsymbol{\mathcal{H}}_{Ak}, \boldsymbol{\mathcal{H}}_{B1}, \cdots, \boldsymbol{\mathcal{H}}_{B(k-1)}$ are the unitary space counterparts of $\boldsymbol{\mathcal{Q}}_{init}, \boldsymbol{\mathcal{Q}}_{Ak}, \boldsymbol{\mathcal{Q}}_{B1}, \cdots, \boldsymbol{\mathcal{Q}}_{B(k-1)}$ (Fig. 4.3c). Then, the exploration direction is given by:

$$\mathbf{d}_T = \mathcal{S}^{-1}(\mathbf{d}_U) \quad , \quad \mathbf{d}_U = \frac{c_A \mathbf{d}_A - c_B \mathbf{d}_B}{\|c_A \mathbf{d}_A - c_B \mathbf{d}_B\|} \tag{4.3}$$

with $\mathbf{d}_U$ the exploration direction in the unitary space (Fig. 4.4), $\mathbf{d}_A$ the unitary vector defined by the k-th attractive point, $\mathbf{d}_B$ the unitary vector defined by the $k-1$ border points, $c_A = 1 - exp(-iter/\tau)$, $c_B = exp(-iter/\tau)$, *iter* being the cumulative number of *IGSP* solved during the k-th exploration, and τ a constant that defines the behaviour of the exploration. This way, at the start of the exploration, the influence of previous computation is relevant, and only after several *IGSP* solutions,

the direction is pointed mainly toward the attractive point.

Now the details of the terms of Eq. (4.3) are illustrated. Attractive point direction $\mathbf{d}_A$ is given by:

$$\mathbf{d}_A = \frac{\boldsymbol{\mathcal{H}}_{init} - \boldsymbol{\mathcal{H}}_{Ak}}{\|\boldsymbol{\mathcal{H}}_{init} - \boldsymbol{\mathcal{H}}_{Ak}\|} \tag{4.4}$$

Then, for the computation of $\mathbf{d}_B$ the following heuristic is proposed:

$$\mathbf{d}_B = \frac{\sum_{j=1}^{N_b} c_{Bj}\mathbf{d}_{Bj}}{\|\sum_{j=1}^{N_b} c_{Bj}\mathbf{d}_{Bj}\|}, \quad \mathbf{d}_{Bj} = \frac{\boldsymbol{\mathcal{H}}_{init} - \boldsymbol{\mathcal{H}}_{Bj}}{\|\boldsymbol{\mathcal{H}}_{init} - \boldsymbol{\mathcal{H}}_{Bj}\|} \tag{4.5}$$

Good results are obtained by employing this heuristic during the simulations of Sec. 4.2. From Eq. (4.5) it is noticeable that $\mathbf{d}_B$ is a weighted sum of the directions $\mathbf{d}_{B_j}$ defined by each border point, with weights c_{Bj} defined as:

$$c_{Bj} = \|\mathbf{1} - (\boldsymbol{\mathcal{H}}_{init} - \boldsymbol{\mathcal{H}}_{Bj})\| \tag{4.6}$$

The meaning of the coefficient c_{Bj} is not trivial: since the operations are performed in the unitary space, each component of $(\boldsymbol{\mathcal{H}}_{init} - \boldsymbol{\mathcal{H}}_{Bj})$ is bounded between $0, 1$. Thus, when $\boldsymbol{\mathcal{H}}_{init}$ approaches $\boldsymbol{\mathcal{H}}_{Bj}$ the value of c_{Bj} increases and $\boldsymbol{\mathcal{H}}_{Bj}$ have a larger influence on the calculation of $\mathbf{d}_B$. This way, if the exploration approaches a previously identified border, its weight on $\mathbf{d}_B$ increases, and $\mathbf{d}_U$ is modified accordingly.

To resume, at each step of the exploration k-th :

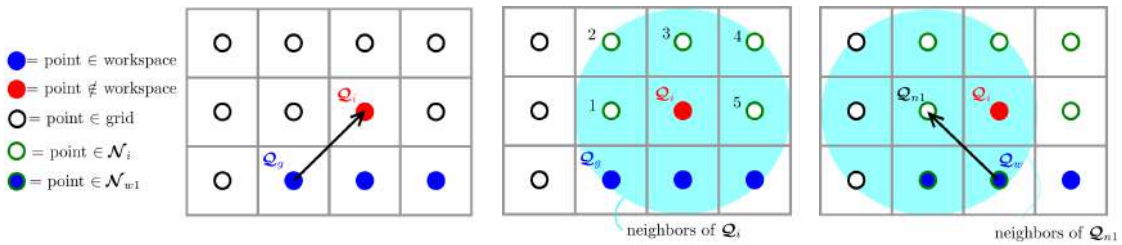
1. all the point of interest (attractive point $\boldsymbol{\mathcal{Q}}_{Ak}$, current workspace point $\boldsymbol{\mathcal{Q}}_{init}$, and border points $\boldsymbol{\mathcal{B}}$), are mapped into the unitary space;

2. then, the exploration direction is computed into the unitary space and mapped back into the task space by employing Eq. (4.3);

3. given the exploration direction $\mathbf{d}_T$, the neighbor of $\boldsymbol{\mathcal{Q}}_{init}$ that points in the direction closer to $\mathbf{d}_T$ is selected as new point. This task-space point is called $\boldsymbol{\mathcal{Q}}_{end}$.

These three steps are performed at line 4 of Alg. 3. Then, the *IGSP* is solved, and it is verified if $\mathbf{Q}_{end}$ lies on the workspace. The computation continues point-by-point until a border is identified. Once the boundary is found, $\boldsymbol{\mathcal{Q}}_{Bk}$ is saved in $\boldsymbol{\mathcal{B}}$ for the next explorations, and $\boldsymbol{\mathcal{Q}}_{init}, \boldsymbol{\mathcal{Q}}_{end}$ are stored in *toDo* for the boundary computation phase described in Sec. 4.1.2.2. Before going to the description of the boundary computation strategy, some remarks are necessary:

- *Necessity of attractive points:* even if attractive points seem to be unnecessary after the first exploration (the exploration direction could be defined by points in $\boldsymbol{\mathcal{B}}$ only), it can happen that previously computed borders define directions for which the computation stalls. An example is reported in Fig. 4.3c: since $\boldsymbol{\mathcal{Q}}_{start}$ is placed in the middle of the two border points $\boldsymbol{\mathcal{Q}}_{B1}, \boldsymbol{\mathcal{Q}}_{B2}$, the direction $\mathbf{d}_B$ of Eq. (4.5) is indeterminate. Thus, attractive points are required to ensure the algorithms do not stall and to drive the exploration away from the stall condition.



(a) (b) (c)

Figure 4.5: Boundary computation strategy: (a) boundary conditions, (b) sorting strategy, (c) point evaluation.

- *Exploration parameter.* The coefficient τ is user-defined, and it should be selected in relation to the scope of the workspace exploration. A high value of τ causes a more complex exploration path at the cost of more iterations. On the other hand, a reduced value of τ should be employed when the user has a previous knowledge of the workspace shape (e.g. slight robot design modifications), the exploration is mainly pointed toward attractive points placed at locations useful to identify all the workspace boundary components. Typical values of τ are related to the grid sampling size s_g, and the grid size. Empirically, the authors experienced optimal results when initializing τ so that $s_g\tau$ is approximately half of the grid size.

- *Number of explorations.* n_{exp} is a user-defined parameter, and it depends on the scope of the workspace exploration. In the case no previous knowledge of the workspace shape is available, the authors experienced optimal results with high values of n_{exp} ($n_{exp} \geqslant 20$); even though this number is quite high and results in a higher computational time than needed most of the times, it gives higher assurance to find all the workspace borders sought. However, if the algorithm is supposed to be used several times (e.g. when performing slight design variations), n_{exp} should and could be reduced to save computational time since there is a-priori knowledge of the workspace borders[3].

4.1.2.2 Boundary Computation Strategy

In this subsection, the algorithm for the reconstruction of the workspace boundaries is detailed. The pseudocode of this routine is reported in Alg. 4. The algorithm starts from a situation where a point $\boldsymbol{\mathcal{Q}}_i$ outside the boundary is identified after the solution of the *IGSP*, with initial guess $\boldsymbol{\mathcal{Q}}_g$ (Fig. 4.5a). These two points are extracted from *toDo*, an array that stores out-of-the-workspace points and their previously employed initial guesses. Subsequently, the goal is to explore the grid in order to identify new points outside the workspace, and to reconstruct the complete boundary. To do that, neighbouring points to $\boldsymbol{\mathcal{Q}}_i$ where the *IGSP* has not been computed are identified over the grid and stored in $\boldsymbol{\mathcal{N}}_i$ (Fig. 4.5b).

Points in $\boldsymbol{\mathcal{N}}_i$ represent possible candidates to be out-of-the-workspace points: in order to verify whether they lie or not in the workspace, the *IGSP* should be solved, and an appropriate initial guess should be identified for each point. However, the

[3]Some workspace slices can be computed *a priori* with the algorithm of [56], or [148], to detect the presence of eventual holes, and then n_{exp} adjusted consequently

Algorithm 4: Boundary Computation Strategy.

```
Function Boundary Computation(toDo):
    while toDo ≠ ∅ do
        Extract Q_g, Q_i from toDo;
        N_i = get neighbors of Q_i not yet computed.;
        Sort N_i w.r.t. distance from Q_g;
        while N_i ≠ ∅ do
            Q_n1 = N_i(1);
            N_i ← N_i \ Q_1;
            N_w1 = get neighbors of Q_1 ∈ WK;
            if N_w1 ≠ ∅ then
                Q_w = N_w1 with best conditioning of J;
                flag = GetWKconditions(Q_w, Q_n1);
                if flag = true then
                    Save Results.
                else
                    Stack [Q_w, Q_n1] ∈ toDo
                end
            end
        end
    end
    return
```

order in which points in $\boldsymbol{\mathcal{N}}_i$ are treated may influence the resulting prediction, since a point may have an appropriate initial guess that is not yet found during the workspace computation. In this work, points $\boldsymbol{\mathcal{N}}_i$ are tested w.r.t. to their distance from $\boldsymbol{\mathcal{Q}}_g$, starting from the closest one in the Euclidean sense[4] (see Fig. 4.5b). Authors experienced good results with this approach, but other heuristics may be proposed.

The next step requires solving the *IGSP* for all the points in $\boldsymbol{\mathcal{N}}_i$. Being $\boldsymbol{\mathcal{Q}}_{n1}$ the first point to be computed in $\boldsymbol{\mathcal{N}}_i$, the goal is to solve the *IGSP*, and an initial guess should be identified. To do that, neighbor points of $\boldsymbol{\mathcal{Q}}_{n1}$ that lie in the workspace are collected in $\boldsymbol{\mathcal{N}}_{w_1}$ (Fig. 4.5c). In case $\boldsymbol{\mathcal{N}}_{w_1}$ is not empty[5], the point with the best conditioning of $\mathbf{T}_1$ (Eq. (2.31)) is extracted among the possible initial guesses in $\boldsymbol{\mathcal{N}}_{w_1}$, and named as $\boldsymbol{\mathcal{Q}}_w$ (Fig. 4.5c). By selecting the initial guess with this heuristic, a high probability that the new configuration preserves the working mode of the initial guess is obtained, but no analytical proof is available. Finally, it is verified if $\boldsymbol{\mathcal{Q}}_{n1}$ lies on the workspace. In case $\boldsymbol{\mathcal{Q}}_{n1}$ is included in the robot workspace, no additional points are stored in *toDo*. On the opposite case, $\boldsymbol{\mathcal{Q}}_w, \boldsymbol{\mathcal{Q}}_{n1}$ are stored in *toDo* for next computation.

The algorithm continues by at first evaluating all the points in $\boldsymbol{\mathcal{N}}_i$ and solving the *IGSP* at all these points. Once $\boldsymbol{\mathcal{N}}_i$ is empty, new points are extracted from *toDo*. When also *toDo* is empty, the algorithm stops. In the next Section, the proposed algorithm is tested for the workspace computation of four different *CPRs*.

[4] In case points have the same distance, a random selection is performed.

[5] In case $\boldsymbol{\mathcal{N}}_{w_1} = \emptyset$, the point $\boldsymbol{\mathcal{Q}}_{n1}$ is skipped, and the *IGSP* not solved.

4.2 Case Studies

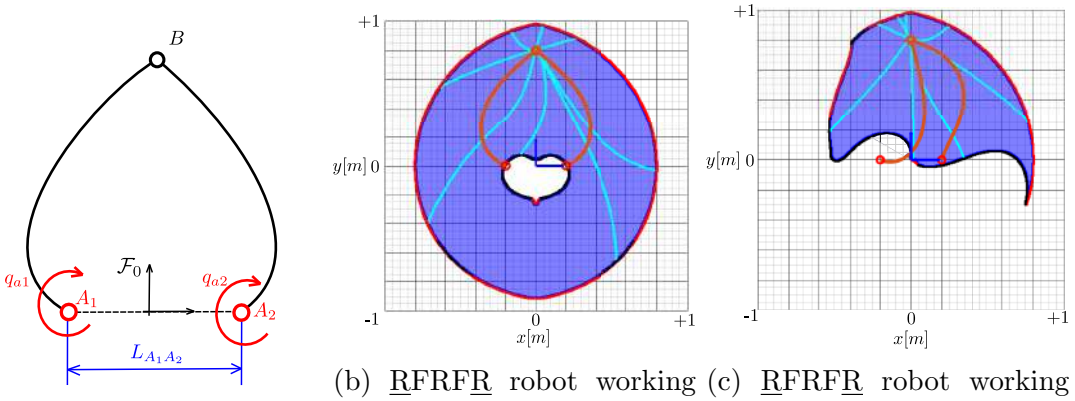
This Section proposes four case studies to show the main features of the proposed workspace border computation algorithm. A comparison between two workspace algorithms and the approach introduced in this chapter, is proposed to validate the applicability of our method, highlight its merits, and also show its main limitations. The *BFA* is compared with the full workspace algorithm [56] that supports three-dimensional investigations (in contrast to our previous work [148] that focuses on planar cases only), to verify the correctness of the results. Also, the *BFA* is compared with the boundary computation algorithm of [84], since it presents interesting features in terms of computational time and, even if it is designed for serial *CRs*, it can also be used for *CPRs*.

The assumed strain mode approach is selected as a discretized modelling technique. As the *IGSP* is to be solver several times during the workspace evaluation, computational efficiency is mandatory. For the same *EE* position accuracy, the assume strain mode approach is considerably faster in the solution of the *IGSP* in comparison to a finite-difference modelling technique [128]. At the same time, both approaches enable the equilibrium stability assessment through the Hessian matrix evaluation [128], and thus the assumed strain mode approach is preferred for its better computational performances. This same modelling strategy is kept for each workspace algorithm, and four assumed modes for each curvature component are used in Eqs. (2.34) ($u_k,\ k = 1, 2, 3$ is approximated with four orthogonal Legendre polynomials). However, any other discretized modelling strategy may be employed at the price of different computational performances. The derivation of the geometrico-static model equations for the assumed strain mode modelling approach is reported in Appendix A.1 for the interested reader.

For all case studies, beams are made of harmonic steel with Young modulus $E = 210$ GPa, beam length is equal to $L = 1$ m. Beams are of circular cross-section with radius $r = 1$ mm. The simulations of this Section are performed with a PC equipped with a CPU Intel Core i7-6700, 3.4GHz, 32Gb RAM, in a Matlab environment. The *IGSP* is solved by Matlab *fsolve* routine, and equations are precompiled as *.mex* function to speed up the computation. A trust-region algorithm is selected since it requires less computational time than other available algorithms (e.g. Levenberg-Marquardt), and the maximum allowed iteration number for the trust-region algorithm is set as 20. If the number of iterations reaches 20, no solution to the *IGSP* is considered to exist. Despite a reduced number of iterations is usually required in regions not close to the border (e.g. 4-5 iterations), the *IGSP* solution in regions near workspace boundaries related to singularities of the *IGSP* problem (Type-1 singularities [56], namely the rank deficiency of $\mathbf{T}_1$ of Eq. (2.32)) usually requires numerous iterations that could slow down the algorithm. In this way, Type-1 singularities can be detected more rapidly.

4.2.1 RFRFR robot

The $\underline{R}FRF\underline{R}$ robot has been introduced in [78], and its workspace computation was investigated in our previous work [148]. The $\underline{R}FRF\underline{R}$ robot has two revolute actuators ($\underline{R}$) placed at the base (points A_1, and A_2 in Fig. 4.6a) that rotate the base of two flexible beams (F). Beams are connected at the opposite side through



(a) RFRFR robot

(b) RFRFR robot working mode 1

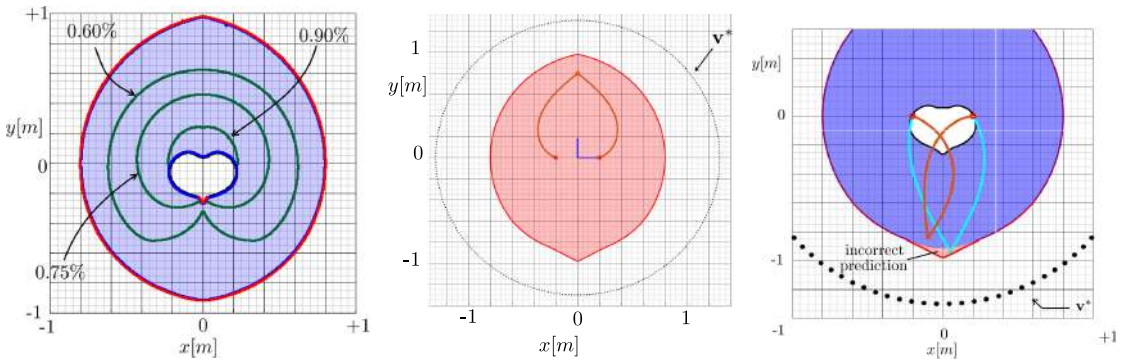
(c) RFRFR robot working mode 2

Figure 4.6: Workspace computation of the $\underline{R}FRF\underline{R}$ robot: the robot architecture (a), and the workspace computation of two different working modes (b),(c). Exploration trajectories are depicted in light blue, results obtained with the approach of [56] in dark blue, Type-1 and Type-2 singularities are represented in red, and black, respectively.

a passive revolute joint (R). For this case study, the distance between the motors $L_{A_1A_2}$ is chosen as 0.4 m. No external forces and gravitational effects are included.

At first, the results obtained by the *BFA* are compared with the flooding algorithm of [56], which computes the full workspace, and not the borders only. The xy plane is discretized with a sampling size s_g = 5mm over the range $[-1, +1]$ m in both directions. Then, τ is initialized so that $s_g\tau$ is approximately half of the grid size, that is $\tau = 200$ (in accordance with the heuristic proposed in Sec. 4.1.2.1). The results obtained by choosing $n_{exp} = 8$ are reported in Fig. 4.6b, where workspace borders caused by Type-1 and Type-2 singularities are depicted in black and red, respectively, while the results of [56] are reported in dark-blue, and the exploration trajectories in light blue. The computational time is significantly reduced, passing from the approximately 12 mins of [56] to the 48 s for the *BFA*. Also, a different working mode has been considered by starting from a second initial robot configuration obtained with a different constant-curvature initial guess. The *BFA* has been employed to obtain the workspace depicted in Fig. 4.6c, the obtained results are in accordance with the one provided by [56] (dark-blue in Fig. 4.6c), and the computational time has been reduced from 130 s to 31 s. Then, the influence of the material strain limit on the workspace size is investigated, as shown in Fig. 4.7a: the $\underline{R}FRF\underline{R}$ robot workspace is computed with strain limit equal to 0.60%, 0.75%, 0.90% and the results are superimposed. By increasing the material resistance, the workspace area increases consequently, and regions closer to the workspace centre become more accessible.

Then, the *BFA* has been compared with the optimization algorithm proposed in [84]. The proposed approach is compared with [84] because of the interesting performances of the optimization algorithm, its simplicity, and the possible application to *CPRs*. The optimization algorithm is an iterative algorithm for the computation of *CRs* boundaries, based on the selection of some points $\mathbf{v}^*$ placed in regions assumed out of the workspace (Fig. 4.7b). Then, an optimization problem is set up to find the robot configuration $\mathbf{y}$ for which the distance between the *EE* position $\mathbf{p}_p$ and $\mathbf{v}^*$ is minimum, subjected to the verification of equilibrium Eq. (2.20).

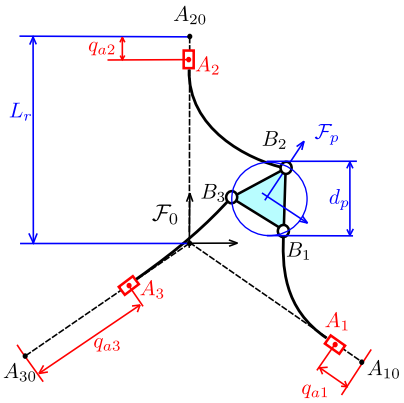


(a) Strain investigation (b) Results with [84] (c) Comparison

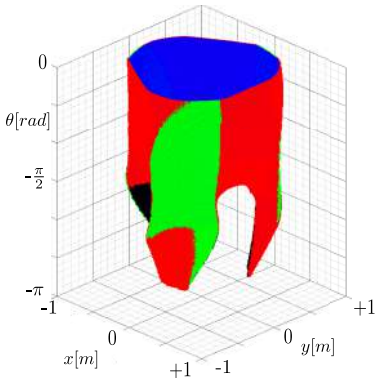
Figure 4.7: Workspace computation of the $\underline{R}FRF\underline{R}$ robot. The influence of strain limits on the robot workspace is reported in (a), where the inner borders generated with different strain limits are depicted in green. In (b), the results obtained with the approach of [84]. In (c), a comparison of the external border prediction between our approach and the approach of [84].

Additional constraints may be included in the optimization problem as well (e.g. equilibrium stability, strain limits). The solution of the constrained optimization problem is solved repeatedly with various $\mathbf{v}^*$ to reconstruct the workspace boundary. At first, the reconstruction of the external workspace border of Fig. 4.6b is considered: to achieve comparable accuracy in the workspace prediction, the optimization approach required 300 points placed over a circumference of radius of 1.3 m and a total computational time of 22 s whereas the proposed algorithm required 48 s. The optimization approach performs a reduced computational time since it requires the solution of a single optimization problem to find points on the border while the border reconstruction strategy of the *BFA* requires multiple *IGSP* solutions to approximate the border as shown in Fig. 4.2c. In the author's opinion, this limitation is the price to be paid for a more generally applicable algorithm, with the additional benefits outlined in this chapter. However, the optimization algorithm [84] is not capable of identifying holes in the workspace (as the one present in the $\underline{R}FRF\underline{R}$ robot workspace), and a different placement of points $\mathbf{v}^*$ is required to attempt the inner hole identification. This is a known limitation of the optimization algorithm, previously mentioned in [84] and, in this direction, the *BFA* performs better since it is able to identify the workspace hole by running a second exploration routine thanks to the proposed space exploration strategy.

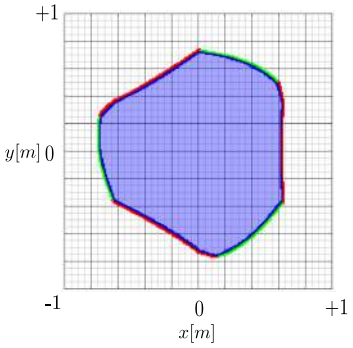
A second issue worthy to be mentioned is related to the prediction of the exterior border: the optimization approach may fail when the robot admits multiple working modes (Fig. 4.7c). In general, the optimization approach finds the robot configuration that minimizes the distance between the *EE* and a user-selected outer point, but there is no constraint that prevents the change in the robot working mode. An example of this issue is reported in Fig. 4.7c: the optimization approach individuates stable solutions that present a shorter distance from points $\mathbf{v}^*$, but these configurations are reachable only by crossing the Type-1 singularity that the optimization approach is not able to individuate. On the other hand, the *BFA* does not suffer from this issue since it is based on the exploration of the boundary by successive iteration in the vicinity of previously detected boundary configurations.



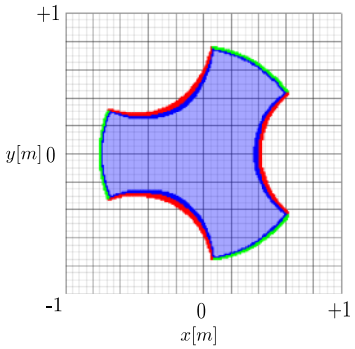
(a) 3-PFR



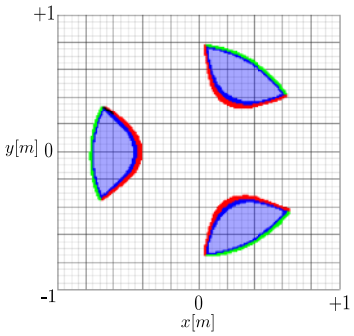
(b) Total orientation workspace of 3-PFR with $\theta \in [-\pi, 0]$



(c) Slice at $\theta = -30°$



(d) Slice at $\theta = -80°$



(e) Slice at $\theta = -90°$

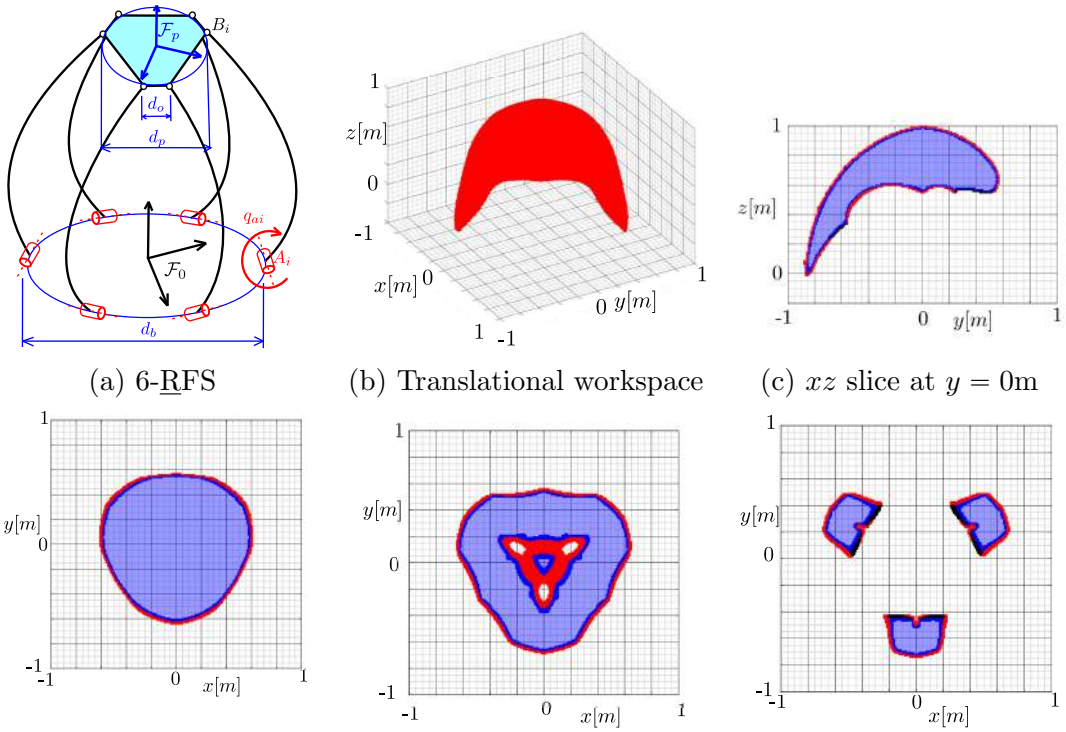
Figure 4.8: Workspace computation of the $3-\underline{P}FR$ robot: the robot schematics (a), total orientation workspace in $\theta \in [-\pi, 0]$ (b), constant orientation slices at $\theta = -30°$ (c) ,$-80°$ (d) ,$-90°$ (e). Type-1 singularities are represented in red, results obtained with the approach of [56] in dark blue, and boundaries generated by actuator limits are represented in green.

4.2.2 3-PFR robot

This subsection investigates the workspace of a $3-\underline{P}FR$ robot. This case study shows the capability of the proposed algorithm to investigate a workspace that involves the position and the orientation simultaneously. The goal is to compute the total orientation workspace of a planar CPR, following the terminology of [43]

The $3-\underline{P}FR$ robot has been the focus of Chapter 3 where its workspace has been analyzed: in this work, a different arrangement of the prismatic actuators has been employed (Fig. 4.8a) with the goal of obtaining wider workspaces. The $3-\underline{P}FR$ robot has three prismatic actuators ($\underline{P}$): $L_r = 2$ m denotes the finite length of the actuators. The beam extremity is actuated by the movement of the prismatic joint, and the beam tip is passively connected to a platform of diameter $d_P = 0.15$ m through passive revolute joints. Joint limits are taken into account during the workspace evaluation.

In order to investigate the orientation ability of the $3-\underline{P}FR$ robot over the workspace, the total orientation workspace is computed by exploring the end-effector position $\mathbf{p} = [x, y]$ and orientation θ over a three-dimensional grid that discretizes $xy\theta$. The xy plane is sampled with a grid size of 10 mm, while θ is discretized by a 2° sampling over the range $[-180, 0]°$, and $\tau = 100$, $n_{exp} = 20$ according to



(a) 6-RFS (b) Translational workspace (c) xz slice at $y = 0$m

(d) xy slice at $z = 0.650$m (e) xy slice at $z = 0.590$m (f) xy slice at $z = 0.500$m.

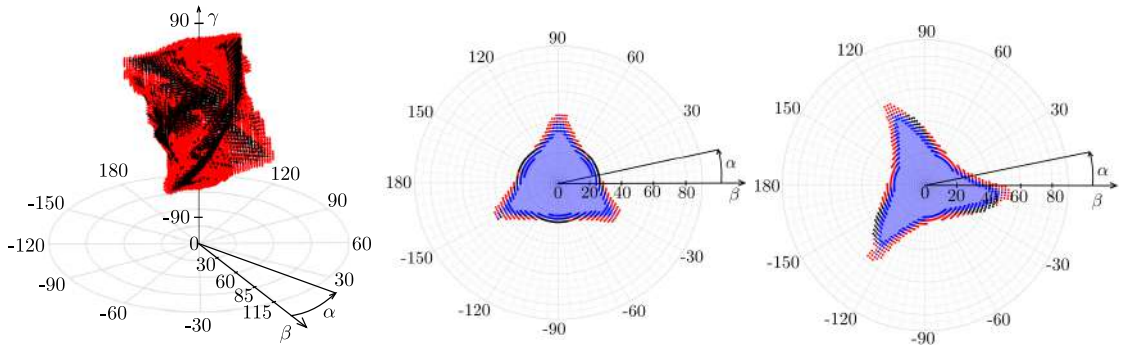
Figure 4.9: Workspace computation of the $6-\underline{R}FS$ robot: the robot schematics (a), the translational workspace at $\mathbf{R}_p = \mathbf{I}$ (b), xz slice at $y = 0$m (c), xy slices at $z = 0.65$m (d), $z = 0.59$m (e), $z = 0.50$m (f). Results obtained with the approach of [56] in dark blue, Type-1 and Type-2 singularities are represented in red, and black, respectively.

the heuristics of Sec. 4.1.2.1. The resulting total orientation workspace is displayed in Fig. 4.8b. Even if the workspace is connected in the interval $\theta \in [-90, 0]^\circ$, by increasing the *EE* orientation, the constant orientation workspace splits into three non-connected components, as highlighted in Figs. 4.8d,4.8e: these slices may be difficult to compute with state-of-the-art approaches based on the iterative computation of several xy slices with different orientations. The *BFA* required 53 mins. while 4h 33 min are required with the approach of [56].

4.2.3 6-RFS robot

This subsection investigates the workspace of a $6-\underline{R}FS$ robot. As commonly done in parallel robots [43], there is limited interest in the evaluation of the 6-*DoFs* capabilities simultaneously and frequently, position and orientation abilities are evaluated separately. Thus, following the terminology of [43], this case study shows the possibility of efficiently evaluating translational workspace and orientation workspace of spatial *CPRs*.

The $6-\underline{R}FS$ robot (Fig. 4.9a) was previously characterized in [56], and it has six revolute actuators that actuate the beams proximal section placed at the robot base over a circle of diameter $d_B = 0.8$ m. Flexible beams are connected to a rigid platform through passive spherical joints (S), such as the one employed in the design of [39]. The platform diameter is $d_P = 0.4$ m, and its overall mass is 100 g. Beams



(a) Orientation workspace at $\mathbf{p}_p = [0; 0; 0.8]m$

(b) Zero torsion workspace ($\gamma = 0°$).

(c) Constant torsion slice at $\gamma = 25°$

Figure 4.10: Workspace computation of the $6 - \underline{R}FS$ robot: the orientation workspace at $\mathbf{p}_p = [0; 0; 0.8]$ (a), the zero-torsion workspace (b), the constant torsion workspace at $\gamma = 25°$ (c). Results obtained with the approach of [56] in dark blue, Type-1 and Type-2 singularities are represented in red, and black, respectively.

are arranged over the platform as described in Fig. 4.9a, with $d_o = 0.1$ m being the distance of adjacent joints on the same corner. Platform and leg weight are considered during the simulations.

The investigation of the $6 - \underline{R}FS$ robot workspace starts with the translational workspace computation where the platform orientation is fixed as $\mathbf{R}_p = \mathbf{I}_3$. A three-dimensional uniform grid samples the xyz space with grid limits of $[-1, +1]$ m at each direction and a sampling size of 10 mm. The exploration parameters are selected as $\tau = 100$ and $n_{exp} = 20$. The translational workspace is displayed in Fig. 4.9b: by employing the *BFA*, the computational time is reduced to 2h and 30 mins in comparison to the 14 h 30 mins required by [56]. Moreover, state-of-the-art approaches based on slice evaluation of the workspace boundaries may difficultly evaluate the robot workspace in cases of slices as the one represented in Fig. 4.9e,4.9f since holes or not connected components are present.

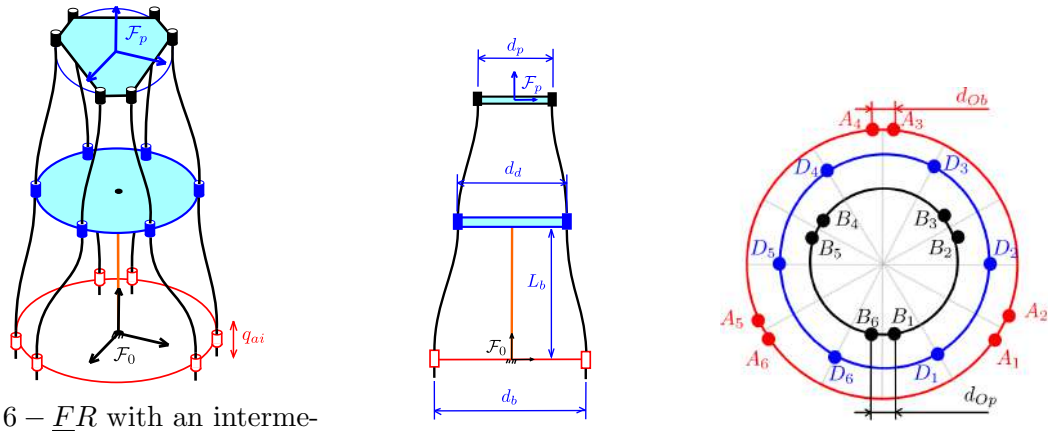
Then, the orientation workspace of the $6 - \underline{R}FS$ robot is evaluated at $\mathbf{p}_P = [0, 0, 0.8]$ m. To perform the evaluation, a Tilt-and-Torsion orientation parametrization is employed [157], and the platform rotation matrix $\mathbf{R}_p$ is defined as:

$$\mathbf{R}_p = \mathbf{R}_a(\alpha, \beta)\mathbf{R}_z(\gamma) \tag{4.7}$$

$$\mathbf{R}_a = \mathbf{R}_z(\alpha)\mathbf{R}_y(\beta)\mathbf{R}_z(-\alpha) \tag{4.8}$$

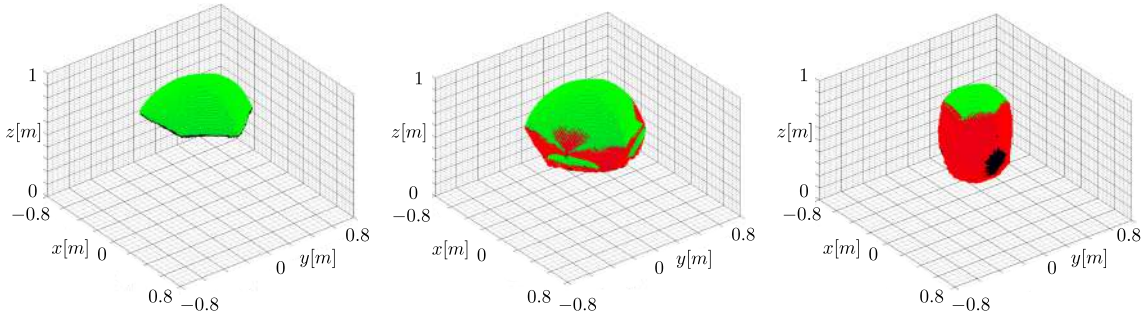
with $\mathbf{R}_z, \mathbf{R}_y$ being elementary rotation matrices and α, β, γ three orientation angles: while α, β defines the tilt of the platform, the angle γ defines it torsion. A uniform grid of step 2° discretized the angle values, with $\alpha \in [-180, 180]°, \beta \in [0, 90]°, \gamma \in [-90, 90]°$. In this case, to define τ, the larger task-space direction is considered (α direction), which results in a higher τ and more complex exploration paths. Thus, $\tau = 45$ and $n_{exp} = 20$.

The resulting orientation workspace, displayed in cylindrical coordinates, is reported in Fig. 4.10a. The *BFA* required 1 h and 52 mins while the flooding algorithm [56] employed 10 h and 3 mins. Then, the zero torsion slice is extracted from the volume and reported in Fig. 4.10b. Maximum tilt abilities ($\beta = 59°$) are reached with $\gamma = 25°$ (Fig. 4.10c).



(a) $6-\underline{F}R$ with an intermediate constaint. (b) Side view. (c) Top view.

Figure 4.11: The $6-\underline{F}R$ with an intermediate constraint. The robot architecture is shown in (a); side and top views of the robot are proposed in (b), and (c), respectively, to illustrate beam connections and relevant design dimensions.



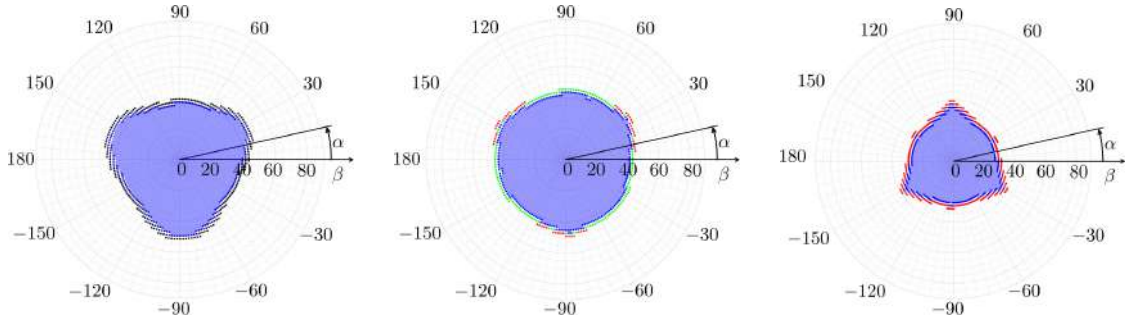
(a) $L_b = 0.6$ m (b) $L_b = 0.4$ m (c) $L_b = 0.2$ m

Figure 4.12: Comparison of translational workspaces at $\mathbf{R}_p = \mathbf{I}$ by varying L_b. Type-1 singularities are represented in red, Type-2 singularities in black, results obtained with the approach of [56] in dark blue, and boundaries generated by actuator limits are represented in green.

4.2.4 6-FR with an intermediate constraint

This subsection focuses on the workspace evaluation of a *CPR* with a more complex structure than the previous case studies to further demonstrate the generality of the *BFA*, focusing on a $6-\underline{F}R$ with an intermediate constraint (Fig. 4.11a). This architecture is similar to the prototype of [27] and, as before, position and orientation abilities are investigated separately.

The $6-\underline{F}R$ has six linear actuators that vary the lengths of the beams placed at the robot base over a circle of diameter $d_B = 0.12$ m (Fig. 4.11b). Beam distal sections are connected to a rigid platform through passive revolute joints (R), with platform diameter $d_P = 0.08$ m, and an overall mass of 100 g. An intermediate disk of diameter $d_d = 0.10$ m, which prevents large nonlinear beam deflections [27], is placed between the base and the platform. Cylindrical pairs are mounted over the disk and, differently from [27], the disk is mounted over a passive flexible beam of constant length L_b. Beams are arranged as described in Fig. 4.11c: the i-th beam is actuated by the linear motor installed on A_i. Then, the beam passes through the



(a) $L_b = 0.6$ m (b) $L_b = 0.4$ m (c) $L_b = 0.2$ m

Figure 4.13: Comparison of zero torsion workspaces at $\mathbf{p}_p = [0; 0; 0.8]$m by varying L_b. Type-1 singularities are represented in red, Type-2 singularities in black, results obtained with the approach of [56] in dark blue, and boundaries generated by actuator limits are represented in green.

cylindrical disk pair placed in D_i, and the distal section of the beam is connected to the platform revolute joint in B_i. The distances between adjacent joints on the same corner are $d_{Op} = d_{Ob} = 0.02$ m for the platform and the base, respectively. Platform, legs, and disk weights are considered during the simulations. Linear actuator bounds are considered of $[0.2, 1.0]$ m.

As done for the previous case study, position and orientation capabilities are studied separately. Focusing on the translational workspace computation, the platform orientation is fixed as $\mathbf{R}_p = \mathbf{I}$. A three-dimensional uniform grid of dimension $[-1, +1]$ in each direction samples the xyz space, with a sampling size of 10 mm in each direction and τ, n_{exp} initialized to $100, 20$, respectively.

The translational workspace is computed by considering three different values of L_b (Fig. 4.12) and, for each simulation, workspace volume (obtained by using the *boundary* Matlab function) is considered to measure the workspace extension. With L_b =0.6 m (Fig. 4.12a), the *BFA* required approximately 1h 40 mins compared to the approximately 12 h of [56]. The workspace volume, mainly delimited by the actuator limits boundary, is 0.0137 m^3. Instead, a larger volume of 0.0376 m^3 is obtained by lowering the disk to L_b =0.4 m (Fig. 4.12b). In this case, the *BFA* required 4h 46 mins and the flooding algorithm of [56] almost 28 h. By further reducing L_b to 0.2 m, the workspace represented in Fig. 4.12c is obtained. The workspace volume is reduced to 0.0175 m^3 and the required computational time is 202 min for the *BFA* (almost 13h for [56]). In both L_b =0.2, L_b =0.6 similar workspace volumes are obtained, but the computational time drastically differs: for $L_b = 0.2$, the workspace is mainly delimited by Type-1 singularities where the solver requires several iterations to converge while, for $L_b = 0.6$, the boundary is defined mostly by mechanical limits where the *IGSP* can be solved with reduced computational cost.

For evaluating the orientation capabilities, the platform orientation is described by employing a Tilt-and-torsion description and the platform orientation matrix $\mathbf{R}_p$ is obtained by Eq. (4.8). Figure 4.13 compares zero torsion workspace at $\mathbf{p}_p = [0; 0; 0.8]$ by varying L_b. The maximum tilt angle of $\beta = 48°$ is reached with L_b =0.6 m (Fig. 4.13a), but the orientation abilities are more uniform w.r.t. α if L_b =0.4 m (Fig. 4.13b). The orientation abilities are instead reduced for L_b =0.2 m (Fig. 4.13c).

4.3 Conclusions

In this chapter, an algorithm for the computation of workspace boundaries of *CPRs* is proposed. The proposed algorithm, based on a free-space exploration strategy and a boundary reconstruction algorithm, reduced the computational time w.r.t. to actuation or task-space discretization strategies by identifying only the boundaries of *CPRs*' workspace. Additionally, the *BFA* included several kinds of constraints, (singularities, equilibrium stability, joint and material limits), and the *BFA* provided the capability to identify holes in the workspace. The *BFA* works with *CPRs* modelling strategies based on general discretization assumptions, which increases the algorithm generality. Four case studies demonstrated the effectiveness of the proposed approach in terms of computational-time reduction and general applicability. The proposed algorithm is well suited for design explorations, where the reduced computational time and the algorithm generality are relevant advantages. However, some limitations need to be acknowledged. First, the use of attractive points is effective, but it may fail when parameters are defined improperly. As with any heuristic method, the parameter tuning is critical for the best performance of the algorithm and n_{exp}, τ require trial-and-error tuning. Then, even though our algorithm is able to identify internal workspace boundaries, there is no certainty of identifying all of them.

The work of this thesis on the workspace computation problem is concluded in this chapter. The next part discusses advancements proposed in this thesis related to the equilibrium stability assessment of *CPRs*.

Part III

Scientific Contributions to the Equilibrium Stability Assessment of Continuum Parallel Robots

Chapter 5

Experimental Assessment of the Equilibrium Stability

Contributions of this chapter: *this chapter proposes the experimental validation of* CPRs *equilibrium stability prediction. A new* CPR *prototype for planar applications is proposed, designed, and tested for the scope. Unstable configurations that limit the robot workspace are theoretically and experimentally investigated. A singularity type, related to out-of-the-plane uncontrolled motions of the planar* CPR*, is experimentally identified for the first time. Experiments demonstrate that, even though the prototype is theoretically planar, a planar model neglecting out-of-the-plane phenomena is inadequate to assess equilibrium stability limits. The contributions of this chapter have been published in [158]*

Experimental validation of *CRs* models received significant attention from the research community, and the literature is vast [106]. Alternative models provide a different trade-off between accuracy and computational complexity, and choosing the appropriate model for the problem at hand is an open question. To this end, experimental data and simulations of different models have been compared in many works [159], [160], [161]. Pose accuracy, namely, the model ability to correctly predict the position and orientation of the robot's end-effector (*EE*), received significant attention: lumped parameter approximations [162], [163], piecewise constant strains models [138], and shooting approaches [74], [164], are some of the most relevant examples of experimentally verified *CRs* models. For *CPRs*, the pose accuracy of the shooting method was investigated in [17] and [32] on 2 and 3-*DoFs* planar *CPRs*, respectively, and in [72] on a 6-*DoFs CPR*; additionally, [28] focused on model parameter calibration. The constant curvature approach, which is suitable for tendon-driven links, was also tested in [38] and, finally, discretization through small segments was experimentally validated for spatial *CPRs* in [35].

Although pose accuracy is a significant issue for control, other robot properties are relevant for characterization and performance evaluation. Due to the high elasticity and possibly limited payload capability of *CRs*, robot stiffness and workspace (*WS*) extension are widely investigated. A *CPRs* stiffness prediction obtained by a discretization approach was experimentally validated in [36], and the *WS* of a 6-*DoFs* prototype experimentally verified in [29] by comparing theoretical and actual posed on several *WS* slices. Recently, the phenomena limiting *CPRs* workspace became of interest since their understanding may produce better-performing designs.

At *WS* limits, *CPRs* may experience stable-to-unstable transition [58], analogously to serial *CRs* [98]. An optimal control approach was proposed in [58] to assess *CPRs* equilibrium stability, and experiments were conducted to verify the correct equilibrium stability prediction. Even though optimal control approaches bring rigorous derivation of equilibrium stability conditions, the complexity of the analysis is relevant. Conversely, discrete energy-based methods [56] bring simplicity to the equilibrium stability analysis.

The novel contribution of this chapter is related to the experimental validation of *CPRs* equilibrium stability assessment using discretized modelling methods. In particular, it is demonstrated that models based on planar displacement assumptions may fail in the equilibrium stability prediction, even though the *CPR* is nominally planar. To this end, a *CPR* prototype for planar applications is originally proposed. The prototype has a $\underline{R}FRF\underline{R}$ overall topology [78], [148] and, thanks to its novel actuation system, links interference with each other is avoided throughout the robot *WS*, leading to a large attainable Cartesian *WS* area. Moreover, the *EE* motion is planar by design since the forces exchanged between the links and the *EE* are parallel to the motion plane, and the overall torque applied on the mechanism is normal to the motion plane. The robot capability in terms of joint-space range (*JS*), Cartesian *WS* size, and equilibrium stability (verified with the energetic approach of [56]) are compared by using i) a model that assumes planar displacements, and ii) a full spatial model. Experiments are conducted to i) verify that a model using planar displacement assumptions is not adequate to predict the equilibrium stability of the proposed prototype and ii) to assess the accuracy of our equilibrium stability prediction. A singularity type, related to out-of-the-plane uncontrolled motions of the planar *CPR*, is experimentally identified for the first time.

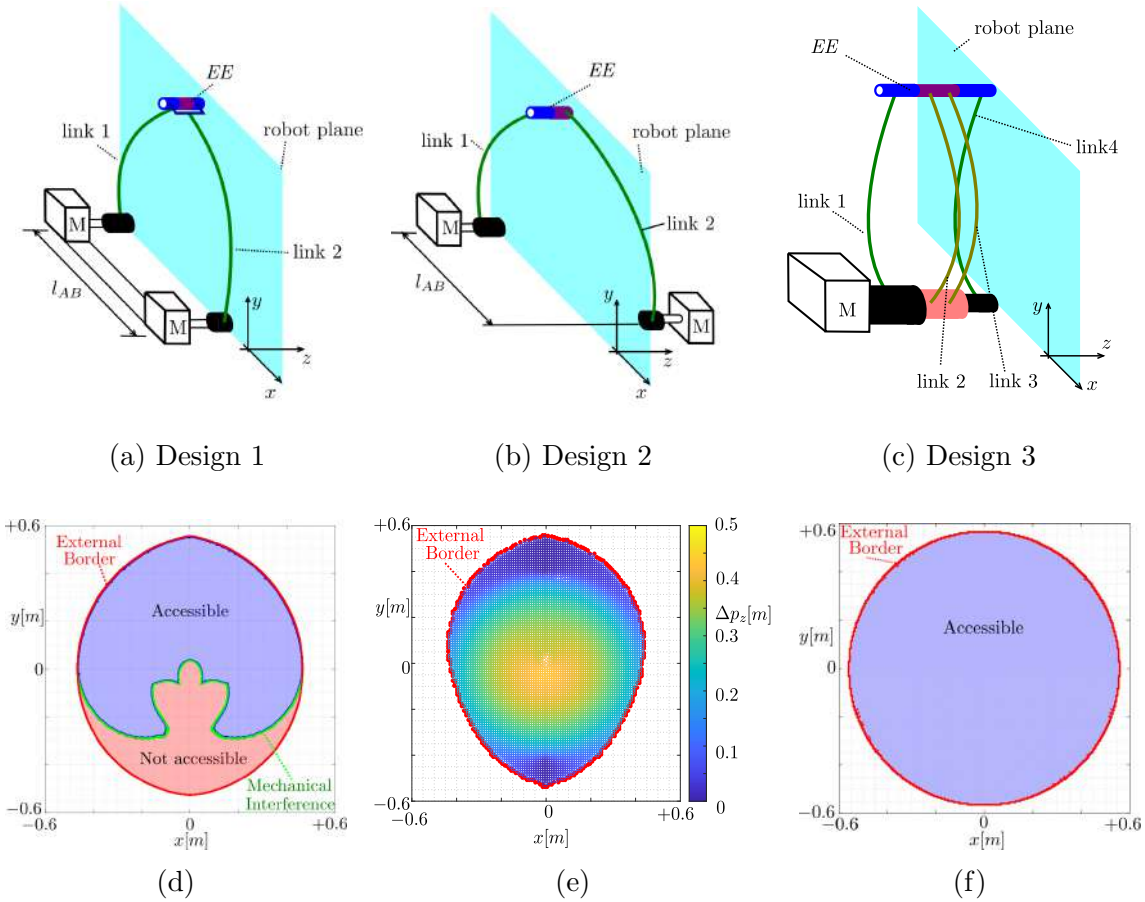
The chapter is structured as follows. Section 5.1 illustrates the robot design and its prototyping. Then, Section 5.2 recalls the specific modelling of the proposed prototype. Section 5.3 is devoted to the robot *JS*/*WS* analysis. Section 5.4 is dedicated to the experimental verification of equilibrium stability prediction, and results are discussed in Section 5.5.

5.1 Prototype Design

This Section focuses on the $\underline{R}FRF\underline{R}$ prototype design proposed in this chapter. The $\underline{R}FRF\underline{R}$ topology was introduced in [78], and its *WS* computation was studied in Chapter 4.9b. The $\underline{R}FRF\underline{R}$ robot has two rotative motors ($\underline{R}$) whose axes are attached to the proximal section of two flexible beams (F). The distal sections of the beams are connected through a passive revolute joint (R), and the robot *EE* is coincident with the passive revolute joint R. All the R joint axes are nominally parallel.

5.1.1 Architecture Selection

The proposed design aims at realizing a nominally planar *CPR* with the largest workspace possible. To keep the *EE* displacement planar, the external forces applied to the *EE* and the forces exchanged between the legs and the *EE* need to belong to the motion plane, and their resultant torque need to be orthogonal to the motion plane. In addition, the *EE*, the links, and the motor axes should not mechanically



(a) Design 1 (b) Design 2 (c) Design 3

(d) (e) (f)

Figure 5.1: Three different possible designs of the $\underline{R}FRF\underline{R}$ robot. (a) two motors on the same side in distinct locations, (b) two motors on opposite sides on distinct locations, (c) two motors on the same side, same location. Figure (d) qualitatively illustrates how mechanical interference reduces the *WS* with Design 1. Figure (e) shows the trend of out-of-the-plane *EE* displacement (Δp_z) with Design 2, and (f) illustrates the *WS* with Design 3. The links length is 0.560 m, while $l_{AB} = 0.2$ m for (d),(e), and $l_{AB} = 0$ for (f).

interfere with each other: this feature is a great limiting factor for parallel robots *WS* size [165]. Some design candidates are the 2-*DoFs* pick-and-place continuum robot of [17], and the $\underline{R}FRF\underline{R}$ of [34], but also many rigid-link five-bar mechanisms may be a source of inspiration [166]. The three most straightforward solutions are hereby discussed:

- Design 1 (Fig. 5.1a). The two $\underline{R}$ motors are attached on the same side of the working plane at a distance $l_{AB} > 0$. This design brings simplicity and great accessibility. Thanks to a clevis fastener, flexible links are aligned at the same *EE* cross-section and connected to the passive joint $\underline{R}$ [17], and the *EE* is in static equilibrium. However, mechanical interference between the links and motor shafts reduces the robot *WS*. The boundary workspace algorithm of Chapter 4 is used to compute the *WS* boundaries generated by mechanical interferences, and a reduction of the *WS* to roughly half of the xy plane occurs (Fig. 5.1d reports the robot *WS* obtained with $l_{AB} = 0.2$ m, links length 0.56 m and no external loads). Preliminary design explorations showed

that the influence of mechanical interference reduces by lowering l_{AB}; on the other hand, l_{AB} cannot be reduced to zero due to actuators encumbrance;

- Design 2 (Fig. 5.1b). The two $\underline{R}$ motors are attached on the opposite side of the working plane at a distance $l_{AB} \geqslant 0$. The flexible beams are connected at different *EE* cross-sections (as [166], [34]), and there is no potential mechanical interference between robot links and motor axes. Unfortunately, the links wrenches will generate a resultant torque which is not normal to the motion plane, and the *EE* cannot lie in the nominal plane without additional constraining systems. This phenomenon is studied by computing the robot *WS* with a spatial robots model (the assumed mode approach of Sec. 2.3.1), and the *EE* out-of-the-plane displacement Δp_z, namely the distance of the *EE* reference point from the reference motion plane, is measured. For instance, Fig. 5.1e illustrates Δp_z over the robots *WS* in the case $l_{AB} = 0.2$, links length 0.56 m, no external loads and offset between the plane of the motors 0.020 m. When the *EE* points toward the *WS* centre, Δp_z increases to unacceptable values. This issue may be solved by considering *EE*-constraining systems (e.g. the vacuum system of [14]). However, such a constraining system modifies the external actions acting on the robot. In this design, lowering l_{AB} reduces Δp_z, but even with $l_{AB} = 0$, Δp_z remains at unacceptable values;

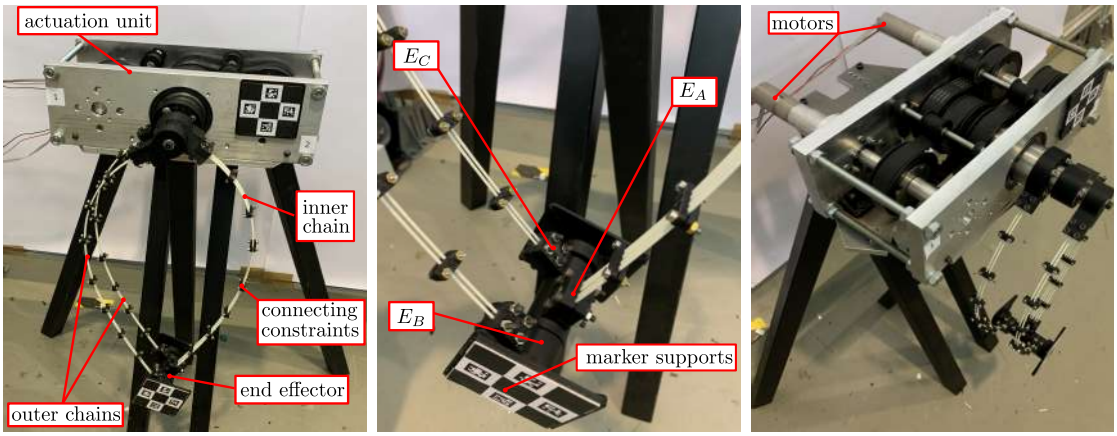
- Design 3 (Fig. 5.1c). The two actuated revolute joints $\underline{R}$ are placed on the same side of the working plane, and they are coaxial ($l_{AB} = 0$). Flexible links 1 and 4 are synchronously moved by the same motor, whereas the other actuator rotates links 2,3. This design ensures no mechanical interference, and the *EE* can maintain a planar configuration, ensuring a large accessible *WS* (Fig. 5.1f). However, the design complexity increases.

Design 3 is the most favourable for realizing a nominally planar 2-*DoFs* system with the largest workspace, without requiring the addition of external constraints, and thus is selected for experiments.

5.1.2 Prototype Manufacture

The proposed prototype is illustrated in Fig. 5.2. To facilitate its description, the robot is subdivided into three groups: flexible chains, the *EE*, and the actuation unit.

Flexible Chains: the beams that transmit the motion from the actuators to the *EE* form the flexible chains. As represented in Fig. 5.2a, one inner and two outer chains are distinguished. While four flexible beams make the former, each outer chain is made by two flexible beams. Beams are made of fibreglass rods of 2 mm diameter. Several possible materials are well suited for *CRs* (e.g. NiTinol alloys [16], Nylon [17], spring steel [36]), and fibreglass is selected mainly for its good tradeoff between lightweight, compliance, and widespread availability on the market. Even if a single beam of a larger diameter could be used to realize each flexible chain, several beams in parallel are installed. For a given flexural inertia moment, a single beam with a larger diameter is highly stressed since strains are proportional to the cross-section diameter. Instead, many small-diameter beams may guarantee an equivalent inertia moment, but the strain on each beam is reduced. Connecting constraints are



(a) Prototype's overview (b) End effector (c) Actuation unit

Figure 5.2: The *CPR* prototype is shown in (a), a top view of the prototype is given in (b) to highlight the actuation unit, and a view of the *EE* is provided in (c).

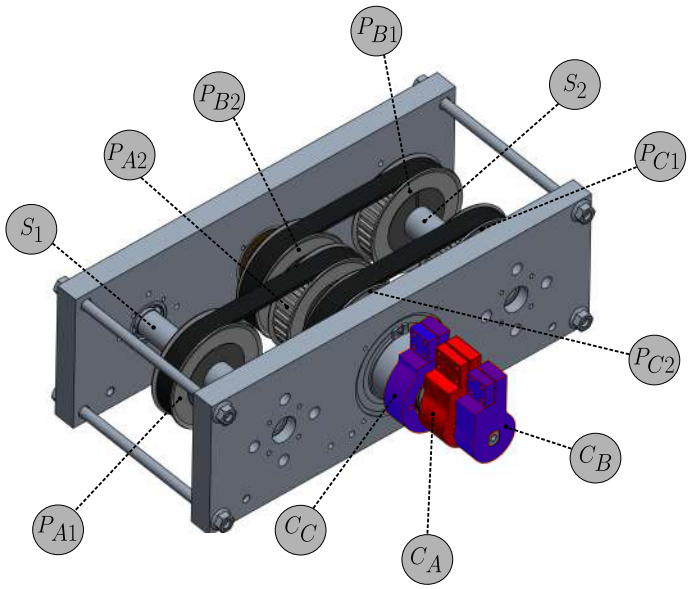


Figure 5.3: Transmission system axonometric view.

also mounted on each chain to increase the stiffness of the robot in the orthogonal direction to the working plane. This way, each flexible chain resembles a beam with a rectangular cross-section[1].

End-effector: the *EE* is illustrated in Fig. 5.2b. The distal sections of the kinematic chains are connected to rigid clamps, as is done for the proximal section. The inner chain is clamped at the *EE* on E_A, while outer chains are connected to E_B,E_C. Two marker supports are attached at both sides of the *EE* to balance the *EE* load statically. The total mass of the *EE* is 218 g.

Actuation Unit: the actuation unit (Fig. 5.2c) is composed of Two DC Maxon motors DCX32L, equipped with a three-stage planetary gearbox (reduction ratio 150:1), and a transmission system specifically designed to drive the flexible chains

[1]A single beam with a rectangular cross-section would be a good design solution, but the market availability of beams with rectangular cross-section whose constitutive material has high admissible strains is significantly lower than circular beams.

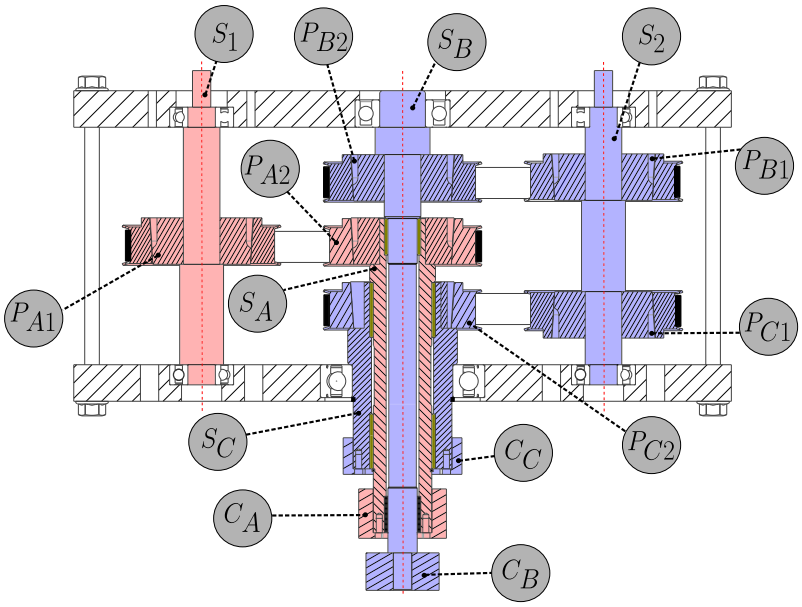


Figure 5.4: Transmission system cross-section: components that rotate at the same angular velocity are shaded with the same colour.

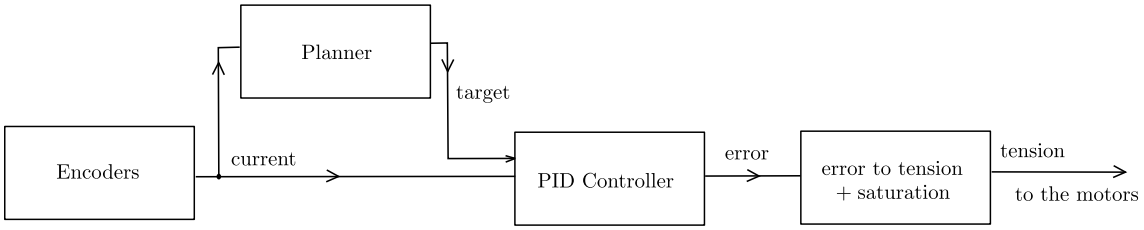
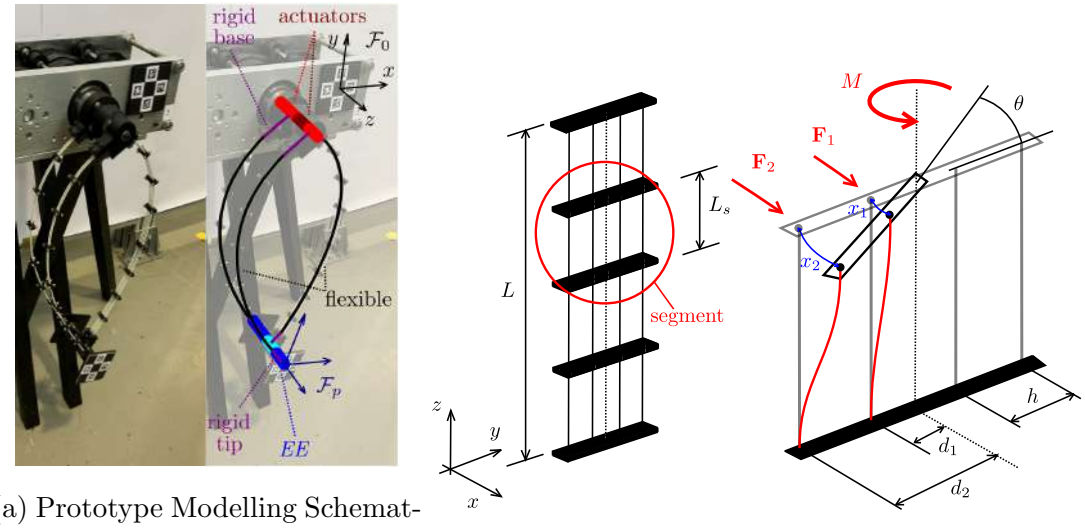


Figure 5.5: Low-level motor control of the prototype.

according to Design 3. The transmission working principle is the following (see Fig. 5.3): a rotation of the shaft S_1 causes an angular displacement of pulley P_{1A}. A synchronous belt transmits the rotation of P_{1A} to the pulley P_{2A}. Similarly, shaft S_2 actuates pulleys P_{1B}, P_{1C} and synchronous belts transmits the rotations to pulleys P_{2B}, P_{2C}, respectively. A set of three concentric shafts (see Fig. 5.4) is used to transmit the rotation of the pulley to the beam clamps (named C_A, C_B, C_C). Shaft S_A connects pulley P_{2A} to C_A and, in a similar fashion, shafts S_B, S_C connects pulleys P_{2B}, P_{2C} to C_B, C_C, respectively. Since P_{2B}, P_{2C} rotate synchronously, also C_B, C_C, display the same angular motion. The proximal section of the inner chain is installed on C_A, while the proximal sections of the outer chains are placed at C_B, C_C.

Finally, a dSPACE 2018-B board completes the automation, controlling the DC motors. The low level control scheme is illustrated in Fig. 5.5, and it is composed by a pair of encoders, a joint-space trajectory planner, a PID regulator, and an error-to-tension converter with saturation limits included. Incremental encoders ENX10 read the current motor angles with a sampling frequency of 200 Hz. Then, the planner receives the user-defined final angles, and a rest-to-rest 5th order polynomial trajectory is build to reach the final angles by starting from the current position in a user-defined time. At each sampling step, the planner provides a target angle to be reached by the motors. The PID controller compares the target angle and the



(a) Prototype Modelling Schematics

(b) Derivation of distributed parameters

Figure 5.6: Prototype Modelling: schematics of the simplified prototype representation (a), and derivation of the distributed coefficients for the flexible link (b).

current angle to calculate the PID error. The PID regulator parameters are selected according to the Ziegler-Nichols empiric procedure. An error-to-tension converter provides the tensions value to be supplied to the motors. In order to avoid too high tension values, a tension saturation is included in the error-to-tension converter.

5.2 Prototype Modelling

This Section describes the peculiarity of the modelling strategy employed for this prototype. As a modelling technique, the assumed strain mode approach of Sec. 2.3.1 is used for its good tradeoff between accuracy and computational cost. However, the prototype's flexible links are made by several flexible beams arranged in parallel, and Sec. 5.2.1 proposes a specific modelling approach. Moreover, since the prototype is theoretically planar, additional details on the validity of a model based on planar displacement are reported in Sec. 5.2.2.

5.2.1 Distributed Material Coefficients Computation

The specific design of the flexible chains needs to be accounted for in the robot model (Fig. 5.6a). Each chain comprises a rigid base attached to the motor shaft, rigidly rotating with it, some flexible beams fixed to the motor shaft, several connecting constraints between the flexible beams, and finally another rigid tip attached to the *EE* revolute joint. As previously mentioned, connecting constraints increase the robot stiffness in the orthogonal direction to the motion plane. However, accounting for their effect in the robot model is not trivial. Each beam and each connecting constraint could be simulated using the assumed mode approach [22], leading to computationally expensive models. Alternatively, other modelling strategies may be used: piecewise constant curvature approaches [7], or piecewise constant strains [85] fit well, but they may require a large number of elastic coordinates. Thus, each flexible chain is considered as a single equivalent beam (Fig. 5.6a) modelled

using the assumed strain mode approximation [22]. In this way, the number of elastic variables in $\mathbf{q}_e$ necessary to represent the flexible chains is reduced while simultaneously considering the effect of connecting constraints.

To represent each flexible chain as a single equivalent beam, it is necessary to calculate an equivalent matrix $\mathbf{K}_{BT}$ (Eq. (2.12)) that represents the overall effect of several beams in parallel connected by the intermediate constraints. Assuming linear isotropic elasticity of the equivalent beam, rules of springs in series and in parallel are used to evaluate two flexural stiffness and the torsional modulus for each flexible chain. As shown in Fig. 5.6b, the flexible chain is modelled as a series of flexible segments, with $s = 1, \cdots, n_b$ indicating the index of each segment, and n_b the number of segments. Segments are assumed to have the same length $L_s = L/n_b$ and, since the beams have circular cross-section beams, $I_x = I_y = I$. Let us consider the inner chain made by four links: assuming the width of the intermediate constraints on the z direction to be negligible, the flexible chain is equivalent to four beams in parallel, and k_x is given by:

$$k_x = \sum_{i=1}^{4} k_{xi} = 4\frac{EI}{L^3}; \quad k_{xi} = \frac{EI}{L^3} \tag{5.1}$$

After k_x, let us consider k_y: under small deformation assumptions[2], each segment is equivalent to four parallel beams clamped at both ends. Thus, the stiffness of each segment k_{ys} can be simply obtained as:

$$k_{ys} = 4\frac{12EI}{L_s^3} = n_b^3\frac{48EI}{L^3} \tag{5.2}$$

and the overall stiffness k_y is the stiffness of n_b elements in series:

$$\frac{1}{k_y} = \sum_{s=1}^{n_b} \frac{1}{k_{ys}} \Rightarrow k_y = n_b^2\frac{48EI}{L^3} \tag{5.3}$$

The computation of the torsional stiffness is not as straightforward, and it is detailed below. First, let us consider Fig. 5.6b, where h is the distance between the beams, assumed to be equal for each of them. Due to the symmetry of the system, we can obtain the k_{zs} of s-th segment by considering only half system:

$$k_{zs} = 2k_b \tag{5.4}$$

where k_b is the contribution of two beams. In order to characterize the torsional stiffness, it is necessary to relate the torsion angle θ to the external moment M. An external moment M is equivalent to two forces F_1, F_2 applied to the beams, that is:

$$M = F_1 d_1 + F_2 d_2 \tag{5.5}$$

where d_1, d_2 are the beams distances to the link centerline (see Fig. 5.6b). Then, by the application of F_1 and F_2, the beams display tip displacements x_1 and x_2, respectively. By considering beams as clamped at both ends, assuming small deformations

[2]Please note that small deformations do not implicate small displacements.

and the local beam torsion over its own axis to be negligible, F_1, F_2 are proportional to the tip displacements x_1, x_2 as follows:

$$F_1 = \frac{12EI}{L_s^3} x_1; F_2 = \frac{12EI}{L_s^3} x_2 \tag{5.6}$$

By inserting Eq. (5.6) into Eq. (5.5):

$$M = \frac{12EI}{L_s^3} (x_1 d_1 + x_2 d_2) \tag{5.7}$$

Since the prototype is nominally planar, it is legitimate to consider the torsion angle θ on each segment to be small. Thus, the displacements x_1, x_2 can approximate by considering the portion of circumference depicted by d, θ, that is $x_1 \simeq d_1\theta, x_2 \simeq d_2\theta$. Then, by introducing $d_1 = h/2, d_2 = 3h/2$, results in:

$$M = \frac{30EI}{L_s^3} h^2\theta \tag{5.8}$$

and k_b by the definition of the torsional stiffness, k_b results in:

$$k_b = \frac{M}{\theta} = \frac{30EI}{L_s^3} h^2 \tag{5.9}$$

by inserting Eq. (5.9) into Eq. (5.4):

$$k_{zs} = \frac{60EI}{L_s^3} h^2 = n_b^3 \frac{60EI}{L^3} h^2 \tag{5.10}$$

Then, k_z is the stiffness of n_b torsional springs in series, that is:

$$\frac{1}{k_z} = \sum_{s=1}^{n_b} \frac{1}{k_{zs}} \Rightarrow k_z = 60 n_b^2 h^2 \frac{EI}{L^3} \tag{5.11}$$

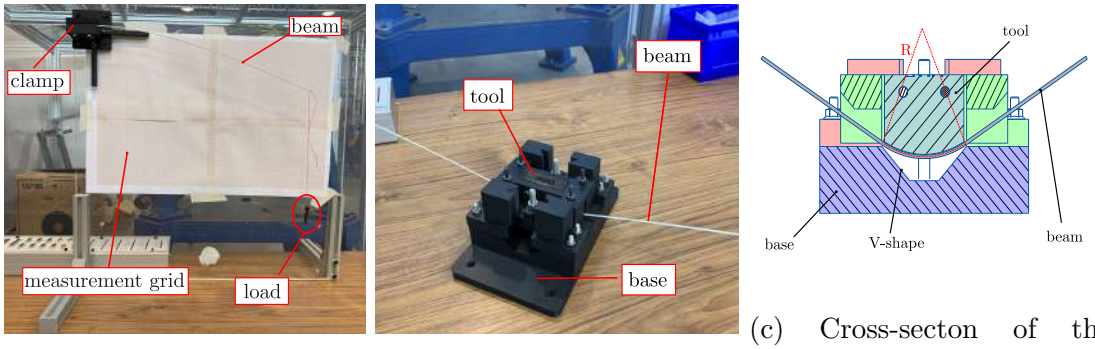
Finally, $\mathbf{K}_{BT}$ is obtained by normalizing over L:

$$\mathbf{K}_{BT-in} = EI \text{ diag}(4, 48n_b^2, 60n_b^2h^2) \tag{5.12}$$

where the subscript $()_{in}$ individuates the inner chain with four beams and diag the 3×3 diagonal matrix whose entries are placed over its principal diagonal. Similarly, the stiffness of a single outer chain $\mathbf{K}_{BT-out}$ is:

$$\mathbf{K}_{BT-out} = EI \text{ diag}(2, 24n_b^2, 12n_b^2h^2) \tag{5.13}$$

By looking at the expressions of $\mathbf{K}_{BT-inn}, \mathbf{K}_{BT-out}$, it is clear that adding connecting constraints increases the robot stiffness in the direction orthogonal to the motion plane. Also, since the torsion that acts on each beam is neglected, the equivalent parameters depend on Young's modulus only: this parameter should be identified appropriately to obtain accurate results.



(a) Young's's modulus setup (b) Strain limit setup (c) Cross-secton of the strain-limit setup

Figure 5.7: Experimental benchmarks. Figure (a) illustrates the Young's modulus estimation setup, Figure (b) shows the strain limit estimation setup and Figure (c) gives details on the strain limit estimation setup.

5.2.2 Remarks on Planarity Assumptions

The proposed prototype (Fig. 5.2) is theoretically planar, and using a planar model to simulate it appears legitimate. A planar model brings mathematical simplicity and a reduced number of variables to be considered [80]. However, as shown later with experiments, a planar model neglecting out-of-the-plane phenomena is inadequate to assess equilibrium stability limits and, ultimately, the workspace size. This Section highlights the most relevant differences between a spatial model and a planar *CPR* model.

Let us consider Fig. 5.6a. For a prescribed *CPR* motion plane, a planar *CPR* model assumes that:

- the cross-section of each beam and the *EE* only perform planar displacements belonging to the reference plane;
- the cross-section of each beam and the *EE* rotate about an axis orthogonal to the reference plane;
- all the forces belong to the reference plane, and all the torques are orthogonal to the said plane only;

If these assumptions hold, it is legitimate to use a simplified model assuming planar model displacements for the solution of the *FGSP* and *IGSP*. The interested reader is addressed to Appendix A.2 for the detailed description of the assumed strain mode approach in a planar case. A major advantage of using a planar model is the reduced number of discretization variables employed. When using Eq. (2.34), the three components of $\mathbf{u}$ need to be interpolated with assumed mode while, in a planar model, only a single scalar component $u \in \mathbb{R}$ is to be discretized. Consequently, Eq. (2.23) is formulated equivalently, but the number of equations reduces since the dimension of $\mathbf{x}$ reduces. Singularity conditions and the equilibrium stability assessment are performed in the same fashion in a planar or spatial model but with different dimensions of $\mathbf{T}_1, \mathbf{T}_2$ and $\mathbf{H}_r$.

5.3 Robot Analysis

In this Section, material characterization and *JS/WS* evaluation of the proposed prototype are described. To obtain accurate *JS/WS* predictions, model parameters should be identified appropriately.

5.3.1 Material Characterization

While robot geometric parameters can be directly measured with limited uncertainty, material parameters are affected by greater variability. Therefore, this subsection focuses on material parameters calibration. It is well known from material science that standardized tests for fibreglass may be conducted to evaluate Young's modulus E, and the strain limit ϵ_{max} [167]. These experimental setups require complex and expensive equipment that may not be available. On the other hand, a simple model-based bending test may be conducted for flexible beams to identify at least E [18], [168], [113]. In this test, a clamped beam is subjected to several known loading conditions, and Young's modulus is selected as the one that minimizes the error between model predictions and experimental measures (Fig. 5.7a). To this end, n_L loads are applied to the tip of a clamped beam. Being $j = 1, \cdots, n_L$ the index representing the j-th different load condition, $\mathbf{p}_{ej}$ is the experimentally measured tip deflection, and $\mathbf{p}_{mj}$ the model predicted tip position, which depends on E. The Young's modulus is found by solving the nonlinear least squares problem [113]:

$$E = \text{argmin}_E \sum_{j=1}^{n_L} \|\mathbf{p}_{mj} - \mathbf{p}_{ej}\|_2^2 \tag{5.14}$$

or, equivalently, as the solution of the following nonlinear equations:

$$G(E) = \frac{\partial}{\partial E} \sum_{j=1}^{n_L} \|\mathbf{p}_{mj} - \mathbf{p}_{ej}\|_2^2 = 2 \sum_{j=1}^{n_L} (\mathbf{p}_{mj} - \mathbf{p}_{ej})^T \mathbf{J}_{mj} = 0 \tag{5.15}$$

with $\mathbf{J}_{mj} = \frac{\partial \mathbf{p}_{mj}}{\partial E}$. The terms $\mathbf{p}_{mj}, \mathbf{J}_{mj}$ depend on the selected beam model: in this work, $\mathbf{p}_{mj}$, and $\mathbf{J}_{mj}$ are obtained thanks to the use of the assumed strain mode approach of [22], but their expression is not reported for brevity. The Young's's modulus evaluation is performed with ten different beams, and each beam was subjected to $n_L = 20$ different load conditions. As shown in Fig. 5.7a, a known tip load was applied to the beam, and the corresponding tip position $\mathbf{p}_{ej}$ was measured with a measurement grid. The resulting Young's modulus was determined as $E = 36.1$ GPa, in agreement with the provider range of $[25, 40]$ GPa.

Since measurement errors may potentially influence the calibrated value, a methodology to estimate how a measurement error is reflected on the calibrated E is proposed. Being $\mathbf{p}_{mes} = [\mathbf{p}_{m1}, \cdots, \mathbf{p}_{mi}, \cdots \mathbf{p}_{mn}] \in \mathbb{R}^{2n_L}$ the vector that collects the measurements, linearizing Eq. (5.15) yields:

$$\frac{\partial G}{\partial E} dE + \frac{\partial G}{\partial \mathbf{p}_{mes}} d\mathbf{p}_{mes} = 0 \tag{5.16}$$

with:

$$\frac{\partial G}{\partial E} \in \mathbb{R} \; ; \; \frac{\partial G}{\partial E} = 2 \sum_{i=1}^{n_L} \left(\mathbf{J}_{mi}^T \mathbf{J}_{mi} + \left(\mathbf{p}_{mi} - \mathbf{p}_{ei} \right)^T \frac{\partial \mathbf{J}_{mi}}{\partial E} \right) \tag{5.17}$$

$$\frac{\partial G}{\partial \mathbf{p}_{mes}} \in \mathbb{R}^{1 \times 2n_L} \; ; \; \frac{\partial G}{\partial \mathbf{p}_{mes}} = -2 \left[\mathbf{J}_{m1}, \cdots, \mathbf{J}_{mi}, \cdots, \mathbf{J}_{mn} \right] \tag{5.18}$$

By further manipulations, dE simplifies as:

$$dE = - \left(\frac{\partial G}{\partial E} \right)^{-1} \left(\frac{\partial G}{\partial \mathbf{p}_{mes}} \right) d\mathbf{p}_{mes} = \mathbf{W} d\mathbf{p}_{mes} \tag{5.19}$$

The matrix $\mathbf{W} \in \mathbb{R}^{1 \times 2n_L}$ correlates dE to $d\mathbf{p}_{mes}$, Assuming $\mathbf{W}$ as deterministic, and assuming each component of $\mathbf{p}_{mes}$ to be affected of a measurement error with normal distribution $\mathcal{N}(0, \sigma_x^2)$, the Young's modulus error follows a normal distribution $\mathcal{N}(0, \sigma_E^2)$, where σ_E [169]:

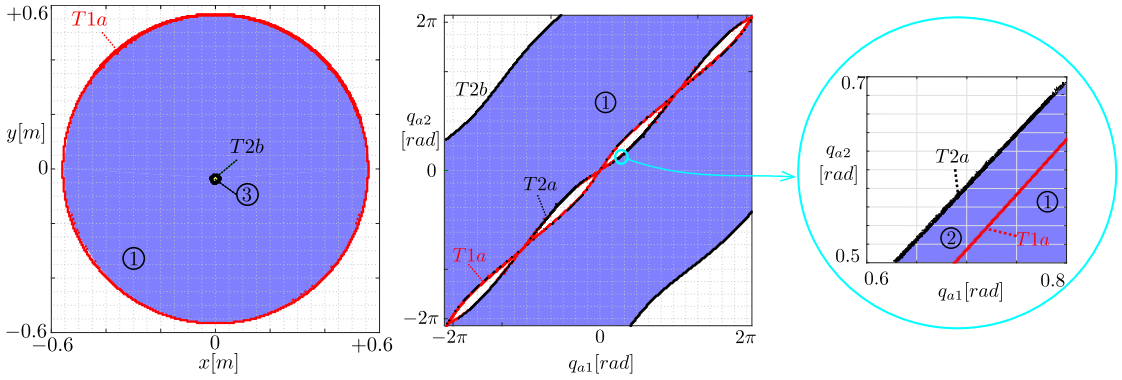
$$\sigma_E = \sqrt{\mathbf{W}\mathbf{W}^T} \sigma_x = w \sigma_x \tag{5.20}$$

By considering the measurements used for the Young's modulus calibration, $w = 12.3 \cdot 10^{-2} \frac{GPa}{mm}$, and w represent how a measurement error is projected on a variation of the calibrated E. For instance, a measurement error of 2 mm (which is realistic with the employed methodology) results in a variation of 0.246 GPa of E, which is less than 1% of the computed values of E. Thus, the simple methodology employed fits the system at hand, which would not significantly benefit from more accurate tip position measurements.

The second material parameter to be evaluated is the strain limit ϵ_{max}, which will be used during the JS/WS evaluation to verify that no leg rupture will occur. As before, instead of performing standard tests, a simplified procedure that can be easily reproduced for fragile materials is proposed. The setup is represented in Fig. 5.7b: a flexible beam of radius r is placed between a V-shaped component and a tool with a circular tip of radius R (see Fig. 5.7c). The tool is pressed onto the beam, which then assumes the same curvature of the tool $u = 1/R$, where pressed. Therefore, the strain on the constant curvature portion is:

$$\epsilon = ru = \frac{r}{R} \tag{5.21}$$

To estimate ϵ_{max}, the beam is tested with several tools characterized by decreasing R until a brittle fracture of the beam occurs. Then, the last value of R where the beam deforms without damage is used to compute ϵ_{max} with Eq. (5.21). This procedure is performed with the same beams used for the Young's modulus calibration, and we obtained $\epsilon_{max} = 2.75\%$, which is in accordance with the provider specification of $\epsilon_{max} \geqslant 2\%$. Please also note that this simplified procedure ensures an underestimation of the real ϵ_{max}: a finite number of tools is used, and the exact ϵ_{max} is only approximated by the last value of ϵ where the beam deforms without damage. In the case more accurate characterization of ϵ_{max} is required, which is not our case, standard tests are recommended [167].



(a) Cartesian Workspace (b) Jointspace Range

Figure 5.8: Jointspace and Cartesian workspace obtained by using planar displacements assumptions. The Cartesian workspace is represented in (a), and the jointspace in (b). Stable and unstable configurations are depicted in blue and yellow, respectively. Type-1 singularities are shown in red, and Type-2 singularities in black.

5.3.2 Joint space and Cartesian workspace Analysis

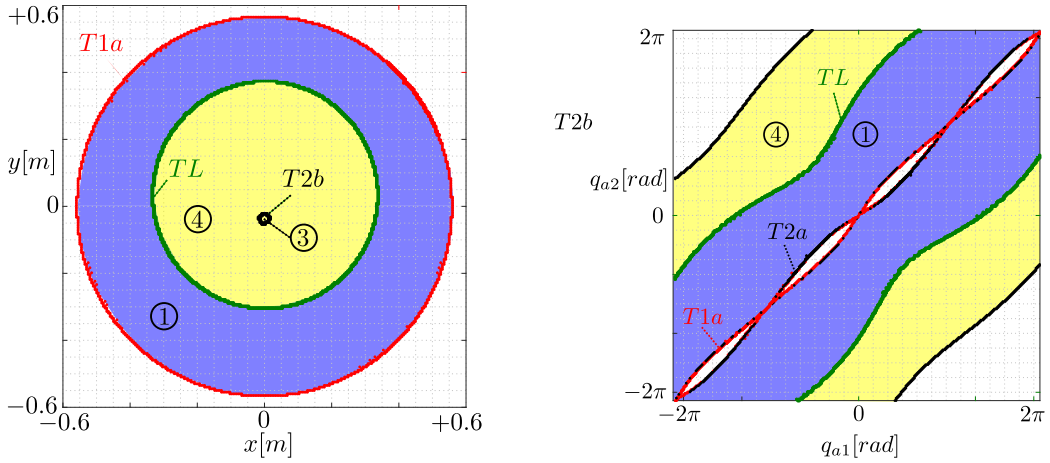
Once the robot model is established and the material parameters identified, we can evaluate the robot motion capabilities in terms of *JS*/*WS* computation. Several phenomena define the *JS*/*WS* limits, and in this Chapter the following phenomena are considered:

- Singularities. As explained in Section 2.2.4, singularities define the *JS*/*WS* boundaries. A configuration is considered singular if the inverse condition number of $\mathbf{T}_1$ or $\mathbf{T}_2$ is below a defined threshold (10^{-5} in our case)[3];
- Equilibrium stability. Stability is checked by looking at the positive definiteness of $\mathbf{H}^r$;
- Strain limits on the flexible links, evaluate if the strain on each leg does not exceed $\epsilon_{max} = 2.75$ %.

The results of the *JS* and *WS* of our prototype are reported in Figs. 5.8,5.9. The evaluation is performed by considering planar displacement assumptions (Fig. 5.8) and by using a full spatial model (Fig. 5.9). The discretization through assumed mode is performed by using four modes on each allowed deformation mode, and thus $m = 12 \cdot 3$ for the spatial model and $m = 4 \cdot 3$ for the planar model. Gravitational loads such as *EE* weight and beams distributed weight are considered. No configuration exceeded the strain limit of $\epsilon_{max} = 2.75$ %.

First, let us consider the case where planar displacement assumptions are introduced in the robot model: Fig. 5.8a illustrates the *WS*, and Fig. 5.8b the *JS*. Region ① is a region where the robot assumes stable configurations. Singularity curve $T1a$, that is, a Type-1 singularity where $\mathbf{T}_1$ is degenerate with $\mathbf{A}$ and $\mathbf{U}$ full rank, delimits ① from one side and represent the external *WS* boundary (Fig. 5.8a):

[3]Matrix $\mathbf{T}_1, \mathbf{T}_2$ have nonhomogeneous units: the use of the inverse condition number is valid as long as it is intended to detect the degeneracy of the corresponding matrices, and not to analyze robots performances.



(a) Cartesian Workspace (b) Jointspace Range

Figure 5.9: Jointspace and Cartesian workspace obtained using a full spatial model. The Cartesian workspace is represented in (a), and the jointspace in (b). Stable and unstable configurations are depicted in blue and yellow, respectively. Type-1 singularities are shown in red, and Type-2 singularities in black. Singularities, where **U** is degenerate, are plotted in green.

there is no solution to the *IGSP* at each point of $T1a$, and the robot *EE* cannot exceed $T1a$ with imposed *EE* position. Singularity curve $T2a$ is a Type-2 singularity where $\mathbf{T}_2$ is degenerate with **P** and **U** full rank, and it defines the inner *JS* limits (Fig. 5.8b): at each point of $T2a$ there exists no static solution to the *FGSP* and, by crossing $T2a$, the robot equilibrium becomes unstable [56]. Between $T1a$ and $T2a$, there exists a small stable region named ② (magnified in Fig. 5.8b) where the robot equilibrium is stable. These configurations can be reached by commanding the robot joints, but region ② cannot be reached by imposing the *EE* position since $T1a$ cannot be crossed with the imposed *EE* position. On the other side, the singularity curve $T2b$, which delimits the *JS* limits, is a parallel singularity where $\mathbf{T}_2$ is singular. Singularity curve $T2b$ encircles a small *WS* region named ③ where the robot equilibrium is unstable[4].

Second, the case where a full spatial model is used is considered: the *WS* and the *JS* are illustrated in Fig. 5.9a, Fig. 5.9b, respectively. Singularity curve $T1a, T2a, T2b$ are equally predicted by a model with planar displacement assumptions and by a full spatial model. **However**, the use of a full spatial model reveals an additional singularity curve named TL, which is a curve where both $\mathbf{T}_1$ and $\mathbf{T}_2$ are singular because **U** is singular. The singularity curve TL defines a new region ④, where the spatial model predicts an unstable equilibrium, and the extension of the stable region ① is consequently reduced.

For each point inside ④, both planar and spatial model predicts the same robot configuration in terms of $\mathbf{q}_a, \mathbf{x}$, but the equilibrium stability is predicted differently. This discrepancy between a model with planar displacement assumptions and a full spatial model is remarkable and, to the best of our knowledge, identified for the first time in *CPRs*. Thus, experiments are conducted to verify which simulation prediction is realistic.

[4]Since T_2 defines the *JS* limits, ③ is not visible in Fig. 5.8b

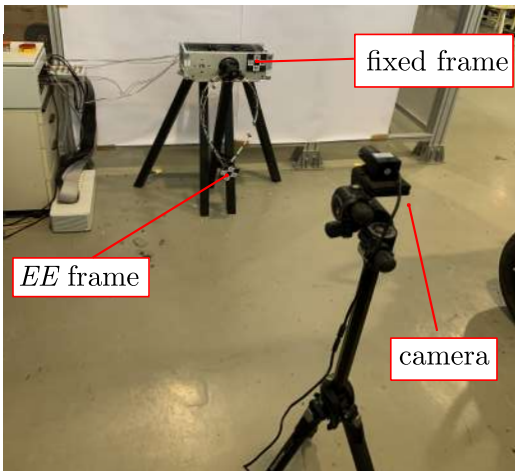


Figure 5.10: The experimental setup used for experiments.

5.4 Experiments

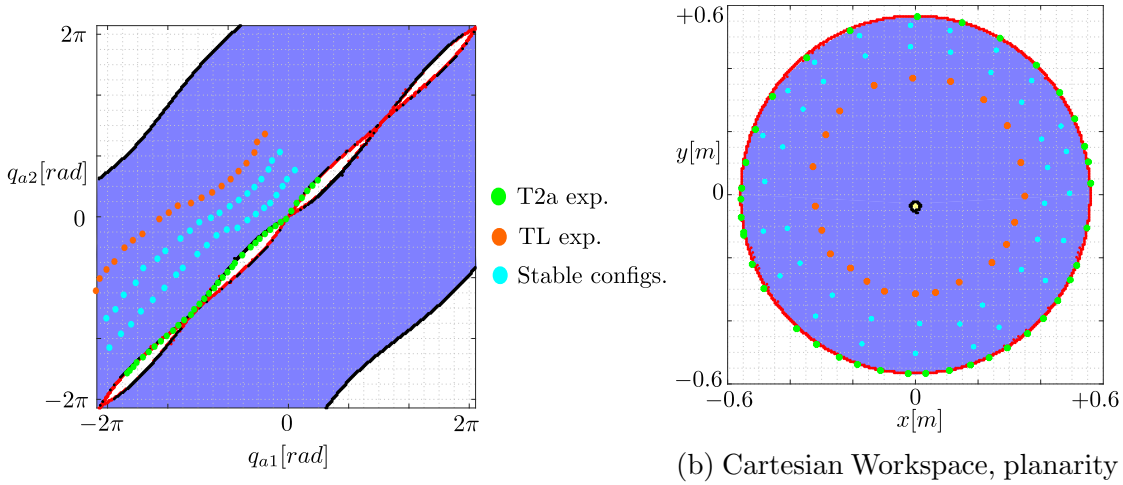
The aim of this Section is to experimentally validate the analysis conducted in Sec. 5.3 about the JS/WS prediction of the proposed prototype. First, the question of whether a model with planar assumptions is adequate or not to model our prototype is addressed. Then, the experimental reconstruction of singularity curves that delimit the prototype range of motions, namely $T2a$ and TL, is performed by comparing the simulation with experimental data to assess the accuracy of the model prediction.

To acquire experimental data, the experimental setup illustrated in Fig. 5.10 is illustrated. A fixed camera was used to record images of the Charuco Board marker attached to the EE, and these pictures were then processed using an OpenCV Python library [170] to reconstruct the EE pose. For each reconstructed pose of the EE, the motors' angular position was also logged, assuming the motors' PID controller steady-state error to be negligible.

5.4.1 *JS* and *WS* verification

In this subsection, the correctness of the JS/WS simulation performed in Sec. 5.3 is verified. To do this, the robot is moved in several stable configurations with EE positions equally distributed over the WS, and motor angles and EE positions at each configuration are stored. Also, the robot is moved to reach exterior WS limits by moving the EE as far as possible from the motor axis and inner WS limits by moving the EE toward the motor axis. Figures 5.11, 5.12 display the superimposition between experimental data and simulations. Experimental joint angles are superimposed over the computed JS, while measured EE positions over the theoretical WS. The model with planar displacements assumptions is used in Figs. 5.11a, 5.11b and the full spatial model is employed in Figs. 5.12a, 5.12b.

By looking at Fig. 5.12, experimental data qualitatively agrees with the simulations obtained by using a full spatial model is used, while the experiments are in disaccordance with a model that employs planar displacement assumptions. In particular, it is possible to state that the stable motion capabilities of the robot are delimited by singularity curves $T2a, TL$. While $T2a$ is equally predicted by



(a) Jointspace Range, planarity assumptions

(b) Cartesian Workspace, planarity assumptions

Figure 5.11: Superimposition between theoretical and experimental data. Results for the model with planar displacement assumptions are reported in (a) for the jointspace range and (b) the Cartesian workspace. Stable and unstable configurations are depicted in blue and yellow, respectively. Type-1 singularities are shown in red, and Type-2 singularities in black. Singularities, where **U** is degenerate, are plotted in green.

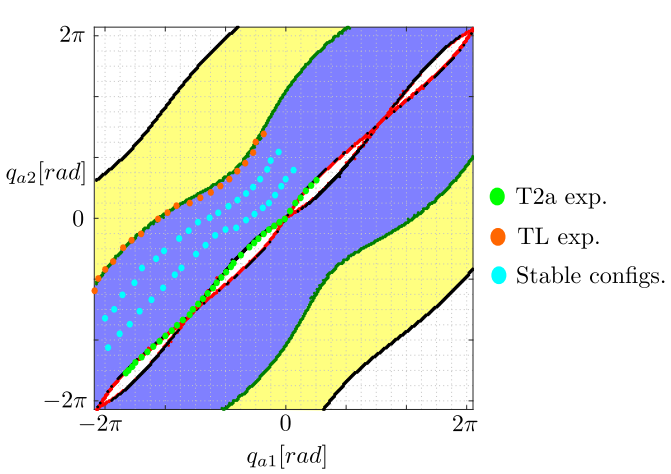
both models, TL is only visible by using a spatial model. An analogy between the constraint singularities appearing in rigid-link lower-mobility parallel robots [171], and the singularity curve TL is noticeable. For some rigid-link lower mobility parallel robots, constraint singularities do not appear in the reduced kinematics model, which neglects the possibilities of the robot platform to move along certain (*a priori*) constrained directions of the space. They may be found if and only if the complete static-equilibrium model, allowing all possible motions in 3D, is analyzed. Analogously to what happens for these constraints singularities, singularities characterized by the curve TL in the present work may be observed if and only if the full (spatial) kinemato-static model of the robot is analyzed.

In the next Sections, singularity curves $T2a$ and TL are analyzed separately, to understand the physical phenomena happening when crossing singularities and to assess the accuracy of our equilibrium stability reconstruction.

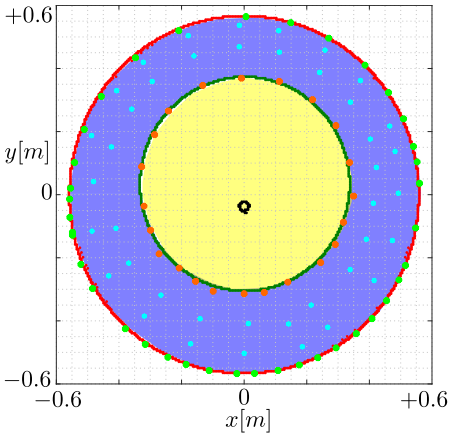
5.4.2 Exterior *WS* boundary

The exterior *WS* boundary is defined by singularity curve $T2a$, which is a Type-2. As theorized in [56], Type-2 singularity delimits stable-to-unstable transitions. In particular, it is experimentally observed that $T2a$ is associated with a snapping phenomenon[5]. When quasi-statically reaching a singular configuration, a non-null motion of the *EE* occurs even though the motors are braked, and the robot dynamically snaps, as shown in Fig. 5.13. The snapping motion occurring about the $T2a$ curve belongs to the motion plane: this is reasonable since both planar and spatial models equally predicted the phenomenon. To reconstruct the $T2a$ curve, the robot is placed in stable configurations as close as possible to the stability limit. The

[5]See the accompanying video of [158], min.0 sec.7, available at `https://doi.org/10.1016/j.mechatronics.2023.103064`

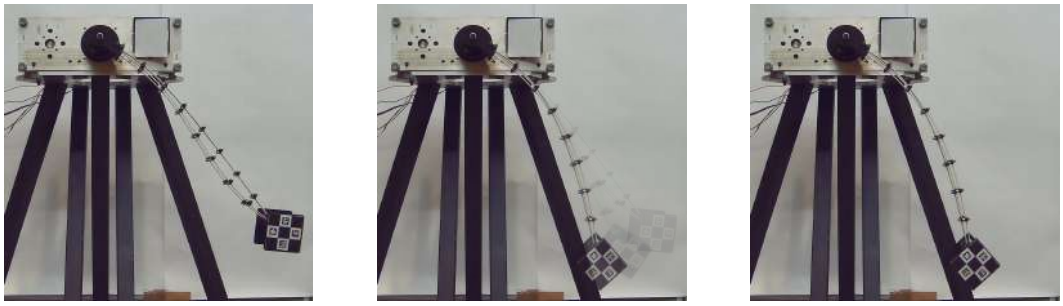


(a) Jointspace Range, full spatial model



(b) Cartesian Workspace, full spatial model

Figure 5.12: Superimposition between theoretical and experimental data. Results for the spatial model are reported in (a) the jointspace range and (b) the Cartesian workspace. Stable and unstable configurations are depicted in blue and yellow, respectively. Type-1 singularities are shown in red, and Type-2 singularities in black. Singularities, where $\mathbf{U}$ is degenerate, are plotted in green.

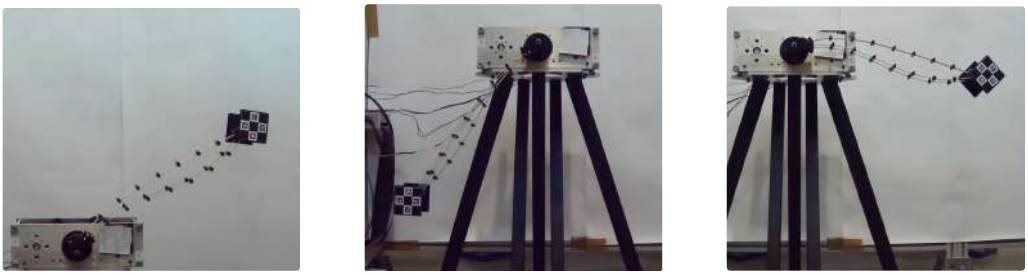


(a) (b) (c)

Figure 5.13: Snapping phenomenon at the $T2a$ curve: when quasi-statically reaching the singular configuration (a), the robot dynamically snaps (b), and it reaches a new stable configuration (c).

motor angles are slowly adjusted to move near $T2a$, aiming not to cross it. Once the robot snaps, the joint values and the Cartesian configuration prior to snapping are recorded as *JS* or *WS* border points. Some examples of these configurations near the $T2a$ are depicted in Figs. 5.14a, 5.14b, 5.14c.

To assess the accuracy of our equilibrium stability prediction, 38 different configurations are tested near $T2a$, with *EE* positions equally distributed over the *WS*. For each test, $\mathbf{q}_{exp}, \mathbf{p}_{exp}$, namely the experimental motor angle and the camera-acquired *EE* position where the singularity happens, are acquired. Also, $\mathbf{q}_t$ is introduced to represent the theoretical motors angles where the instability should happen: $\mathbf{q}_t$ is defined as the point that lies over $T2a$ closest to $\mathbf{q}_{exp}$. Finally, $\mathbf{p}_t$ defined the Cartesian point that corresponds to $\mathbf{q}_t$ (see Fig. 5.15 for a graphical illustration).



(a) (b) (c)

Figure 5.14: External border reconstruction. Three different configurations close to the singularity curve $T2a$ are illustrated.

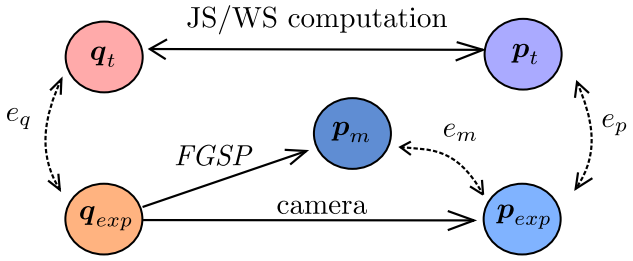


Figure 5.15: Graphical representation of the variables necessary for the errors definitions.

For each configuration, the following errors are defined:

$$e_q = \|\mathbf{q}_{exp} - \mathbf{q}_t\|_2 \tag{5.22}$$

$$e_p = \|\mathbf{p}_{exp} - \mathbf{p}_t\|_2 \tag{5.23}$$

where e_q, e_p are named motor angles error and *EE* position error, respectively. Table 5.1 summarizes the results: a mean $e_q = 2.68^\circ$ is obtained, which corresponds to a mean $e_p = 23,13$ mm (4.10 % w.r.t. total link length of 564mm).

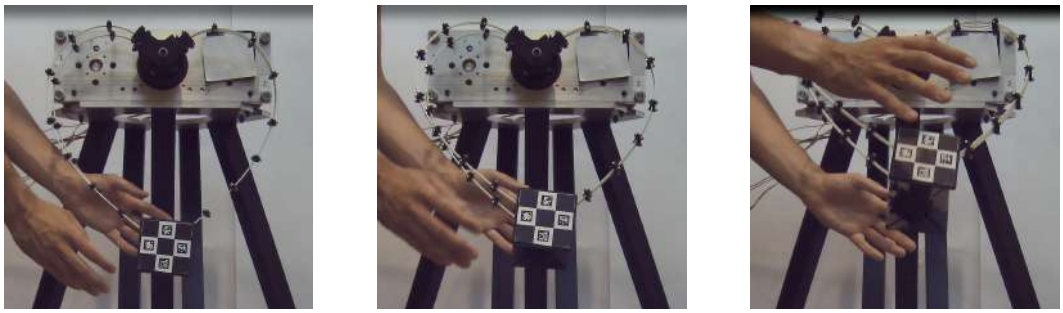
The causes of error are numerous: hardware inaccuracies (e.g., friction, belt elasticity, gearbox clearance) and model errors (such as parameter uncertainties, distributed parameter assumption, and discretization inaccuracy). To investigate the discretized model errors, the vector $\mathbf{p}_m$, that is the *EE* position obtained by solving the *FGSP* with motor angles $\mathbf{q}_{exp}$ (see Fig. 5.15), is computed. Additionally, the error e_m is defined:

$$e_m = \|\mathbf{p}_{exp} - \mathbf{p}_m\|_2 \tag{5.24}$$

where e_m represents the model error. The mean value of e_m, obtained with the use of four assumed modes [56], is 18,56 mm (3,29 %), which is comparable to e_p. To exclude the discretization model by the causes of inaccuracy, the e_m obtained by the shooting-based model of [18],[172] is compared with the one obtained with the assumed mode approach [128]. By solving the *FGSP* over each $\mathbf{q}_{exp}$, the model of [18] results in a mean $e_m = 17,71$mm, comparable to the results of our model.

	e_q [°]	e_p [mm]	e_m [mm]
Mean	2,68	23,13	19,37
Median	2,78	17,04	16,46
Max	6,56	67,53	45,50
Dev.Std	1,59	14,28	10,63

Table 5.1: Motor angles, *EE* position, and model errors for the $T2a$ reconstruction.



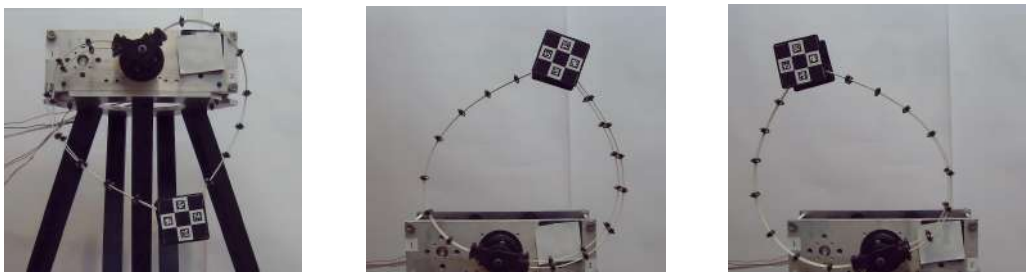
(a) (b) (c)

Figure 5.16: Instability at the TL curve: when quasi-statically reaching the singular configuration (a), the robot *EE* moves out-of-the-plane (b). The *EE* is manually blocked (c) to not brake the robots legs.

5.4.3 Inner *WS* boundary

The inner *WS* boundary is defined by singularity curve TL, where both $\mathbf{T}_1, \mathbf{T}_2$ are rank deficient since $\mathbf{U}$ is rank deficient. According to the terminology of [56], this is a leg singularity. This elastic equilibrium limit is different from the snapping phenomena of $T2a$[6]. Similarly to $T2a$, a non-null *EE* motion occurs about the singularity curve even if the motors are braked. However, the uncontrolled *EE* motion results in an out-of-the-plane link deflection and *EE* motion (as illustrated in Fig. 5.16). When the robot lies in ①, the motors rotations generate only in-plane *EE* motion but, after crossing TL, the motors rotations generate an out-of-the-plane *EE* motion that was not possible before crossing TL (even if this motion is not controllable).

[6]See the accompanying video of [158], min.0 sec.38, available at `https://doi.org/10.1016/j.mechatronics.2023.103064`

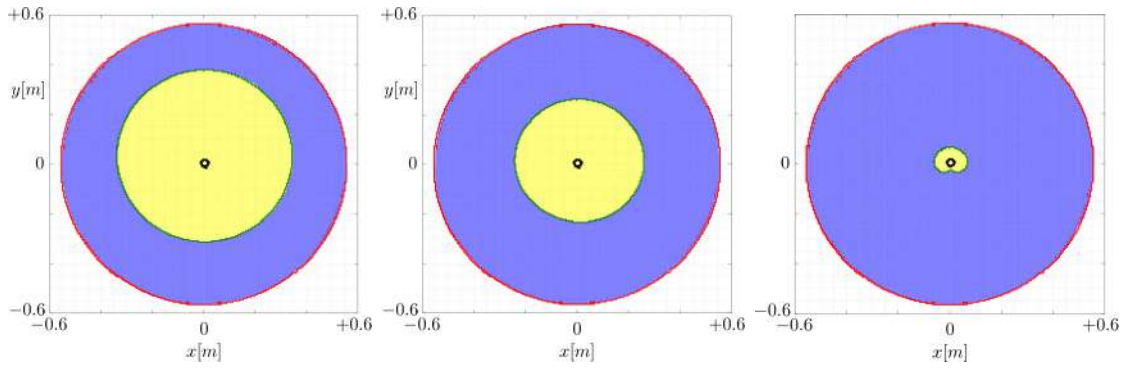


(a) (b) (c)

Figure 5.17: Inner border reconstruction. Three different configurations close to the singularity curve TL are illustrated.

	e_q [°]	e_p [mm]	e_m [mm]
Mean	4,17	27,86	20,59
Median	3,99	26,75	19,50
Max	8,03	44,77	31,52
Dev.Std	1,98	8,44	5,89

Table 5.2: Motor angles, *EE* position, and model errors for the TL reconstruction.



(a) $n_b = 8$ (b) $n_b = 20$ (c) $n_b = 50$

Figure 5.18: Comparison of *WS* by varying the number of connecting constraints (n_b). (a) the actual solution, $n_b = 8$, (b) $n_b = 20$, (c) $n_b = 50$. Stable and unstable configurations are depicted in blue and yellow, respectively. Type-1 singularities are shown in red, and Type-2 singularities in black. Singularities, where **U** is degenerate, are plotted in green.

Different configurations (22) are tested with the *EE* uniformly placed on TL (see Figs.5.17a, 5.17b, 5.17c for some near-singular configuration examples). As in the previous case, the focus is directed on the motor error e_q, the *EE* position error e_p, and the model error e_m (Table 5.2 summarizes the results). A mean $e_q = 4.17°$, which corresponds to a mean $e_p = 27.86$ mm (4.94 % w.r.t. total link length of 564 mm), is obtained.

As for $T2a$, the discretized-model error is investigated: the mean values of e_m, obtained with the use of four assumed modes [128], is 20,58 mm (3,65 %). A similar result is obtained with the shooting-based model of [18], with a mean error of $e_m =$ 19.11 mm (3,39 %).

5.5 Discussion of the results

Globally, a significant agreement between the experimental data and the equilibrium stability prediction provided by a full spatial model is obtained, as qualitatively illustrated in Fig. 5.12. In particular, singularity curve TL is correctly predicted by a spatial model only: this is reasonable since a planar model disregards out-of-the-plane phenomena. Although planar *CPRs* are frequently analyzed with planar models ([78],[80],[38]), this chapter clearly showed that planar models are insufficient in predicting stability limits, and thus JS/WS limits of such robot.

The experimental reconstruction of $T2a$ and TL curves confirm the accuracy of our equilibrium-stability prediction approach since the difference between theoretical

and experimental motor angles at singularities (e_q) is very low ($\leqslant 5^\circ$). The *EE* position error e_p, less than $\leqslant 5\%$ of the length of the link, is acceptable considering the current state-of-the-art (see [113]). As causes of errors, model simplification is one of the possible reasons, but it should be primarily considered that all the measurements are conducted in the proximity of singular configurations, where any small error (e.g. belt elasticity, gearbox clearance) may be possibly reflected in significant variations of the configuration variables. The model accuracy is tested in several stable positions far from singularities (see Fig. 5.12, stable points), and the average model error with four assumed modes is $e_m = 12.81$ mm (2.27 % of the length of the link), significantly lower than 19.37 mm and 20.59 mm of configurations near $T2a$ and TL, respectively.

To the authors' knowledge, this is the first time a singularity of matrix $\mathbf{U}$ is discovered and experimentally verified for *CPRs* with actuation at their base. In tendon-driven *CPRs*, singularities of $\mathbf{U}$ were identified in [113]. In that case, degeneracies of $\mathbf{U}$ were associated with leg singularities where multiple tendons were slack. However, the physical phenomena experienced is different. To better understand what happens in our case, let us consider the forward kinemato-static problem of Eq. (2.32), here reported for simplicity:

$$\begin{bmatrix} \Delta\mathbf{q}_c \\ \Delta\mathbf{q}_u \\ \Delta\mathbf{q}_e \end{bmatrix} = -[\mathbf{P}\ \mathbf{U}]^{-1}\left(\mathbf{A}\Delta\mathbf{q}_a + \mathbf{W}\Delta\mathbf{f}_p\right) = \mathbf{J}_k\Delta\mathbf{q}_a + \mathbf{C}\Delta\mathbf{f}_p \tag{5.25}$$

where $\mathbf{J}_k = -[\mathbf{P}\ \mathbf{U}]^{-1}\mathbf{A} \in \mathbb{R}^{(nc+m)\times n}$ is called Jacobian matrix, and $\mathbf{C} = -[\mathbf{P}\ \mathbf{U}]^{-1}\mathbf{W} \in \mathbb{R}^{(n_c+m)\times 3}$ is named compliance matrix. When the manipulator approaches the singularity curve TL, matrix $\mathbf{U}$ becomes rank deficient. Then, by inspection of $\mathbf{C}$, we noted that submatrix $\mathbf{C}_u$ relating $\Delta\mathbf{q}_u = \mathbf{C}_u\Delta\mathbf{f}_p$ is ill-conditioned, displaying negligible stiffness in the direction orthogonal to the robot motion plane. This further confirms the inability of the model with planar displacement assumptions in the identification of TL since it fails to detect the lack of stiffness in the out-of-the-plane direction.

As the manipulator displays negligible stiffness in the direction orthogonal to the robot motion plane when approaching TL, it is interesting to explore how the stiffness of the links influences this phenomenon. To do this, the number of connecting constraints n_b is varied, increasing the torsional stiffness of the beams (see Eq. (5.12), (5.13)). Starting from the current solution with $n_b = 8$, by increasing the number of n_b, the unstable *WS* area reduces, as shown in Fig. 5.18. In particular, by selecting $n_b \geqslant 50$, almost all the Cartesian *WS* is theoretically reachable.

5.6 Conclusions

This chapter addressed the experimental validation of equilibrium stability of *CPRs* predictions, and it demonstrated the inability of a model based on planar displacement assumptions to predict the equilibrium stability of a planar *CPR*. A new planar *CPR* was proposed for the scope. The prototype was designed to be nominally planar and such that no mechanical interference between robot components could occur. Because of the prototype architecture, a material parameter modelling methodology for the specific design of the flexible chains employed was originally

proposed. Finally, a singularity related to out-of-the-plane uncontrolled motions of the planar CPR is experimentally identified for the first time. The next chapter further investigates the equilibrium stability assessment of *CPRs*, by proposing a performance index to measure the distance to instability.

Chapter 6

Performance Assessment: Directional Critical Load Index

Contributions of this chapter: *this chapter proposes to use the magnitude of a force that brings instability to the* CR *equilibrium as a measure of the distance to instability. The major advantages of this metric are the intrinsic physical meaning, the practical interpretation of the results and the well-defined unit of the measurements. The proposed index (named directional critical load index) is based on energetic considerations and discretization techniques, including a wide range of currently employed models. Three different case studies (buckling of straight beams, a two-tube concentric tube robot, and a spatial continuum parallel robot) illustrate and demonstrate the main results of this chapter. The contributions of this work has been submitted as a research paper to [173]*

A major limiting factor of serial and parallel *CR* designs is the possibility of incurring in equilibrium instabilities: the elastic structure of *CRs* enables possible stable-to-unstable transitions, which ultimately depends on the *CR* loading conditions. Thus, stability metrics, measuring the distance to instability, are a potentially powerful tool for design optimization, trajectory planning and control of *CRs*. However, as shown in Sec. 1.4.2, only a few works explored this topic. According to [42], a good performance metric should i) not involve mixed units of measurement, ii) admit an analytical expression (to be used for optimization), iii) be bounded in magnitude, and iv) attain a physical meaning to enable effective comparison and quantifications. However, except for the two-tube concentric tube robot case in [104], none of the previously discussed indices ([98], [58]) satisfy these requirements. In this chapter, the focus is directed toward a methodology for the measurement of *CRs* distance to instability. A criterion for said measurement is introduced, which is based on the magnitude of the force that brings instability to the *CR* equilibrium; such criterion has a physical meaning and a well-defined unit (Newton) intuitively, and it can be analytically computed.

The index proposed in this chapter, named the *directional critical load index* (*DCLI*), estimates the magnitude of an external load that will cause a stable-to-unstable transition of the *CR* when applied in a given direction. The derivation of the *DCLI* is based on the energy-based modelling formalism described in Chapter 2, and its formulation can be applied to *CRs* of different architectures, serial and parallel alike, as it will be shown in the Case Studies in Sec. 6.3. The formulation of *DCLI* does not depend on the selected discretization strategy, and a

finite-differences modelling technique (Sec. 2.3.2) is used for its efficacy in computing successive model-equations derivatives. However, any discretized modelling approaches described in Sec. 2.1.2 could also be used. A major advantage of the *DCLI* is its well-defined measurement unit (Newton) and its intuitive lower and critical bound zero: these characteristics lead to practical physical interpretations of the index results. Although it would be beneficial to have an upper bound of *DCLI*, there may exist no force in a given direction, namely a force of infinite magnitude to cause the *CR* instability, and this is further confirmed in Sec. 6.3.

In this chapter, *DCLI* is defined by considering only external forces as a cause of the *CR* instability, as they cover many practical cases (e.g. gravitational loads applied to the *EE* or at a specific *CR* location, contacts forces). Moreover, the derivation of *DCLI* assumes to know the application point and direction of the force that causes instability. Although this may seem a limiting factor, guidelines for selecting the application point and direction of the force may come from the practical scope of the *CR*. For instance, when the *CR* is used for manipulation tasks, a possible source of instability may be an additional *EE* load aligned with the gravity. Finally, the *DCLI* is computed by investigating the influence of an external force on the equilibrium stability when the motors are fixed, considering a practical case where the robot is moved by imposing motor values. In practice, the *DCLI* is derived via a linearization of the eigenvalues of the reduced Hessian matrix of the potential energy. Accordingly, an analytical expression of the external load for which instability occurs, i.e. for which an eigenvalue vanishes, can be established. As an additional benefit, the index derivation is based on algebraic computations only, and differential equations integration typical of *OC* approaches are avoided [98], [75].

The chapter is structured as follows. Section 6.1 recalls the energy-based modelling approach by introducing slight modifications useful for the *DCLI* derivation. Section 6.2 is devoted to the stability index derivation, and case studies are proposed in Section 6.3 to verify the capability of *DCLI* to measure the distance to the instability and to quantify the closeness of *DCLI* to the exact critical load. Finally, conclusions and limitations are highlighted in Section 6.4.

6.1 Modelling and Equilibrium Stability Assessment

This section recalls the energy-based modelling approach of Chapter 2 by introducing slight modifications that simplify the *DCLI* derivation. In particular, let us consider the total *CRs* energy of Eq. (2.17), here reported for clarity:

$$V_{tot} = \sum_{i=1}^{n_b} V_{beams_i} + V_p \tag{6.1}$$

where V_{beam_i} is the i-th beam energy (comprising deformation energy and external loads acting on the beam), n_b is the number of beams, and V_p is the platform energy. Since the index proposed in this chapter estimates the magnitude of an external load that will cause a stable-to-unstable transition of the *CR*, let us consider an *additional* external concentrated force $\mathbf{f}$ applied at an application point P_{app}, denoted by the

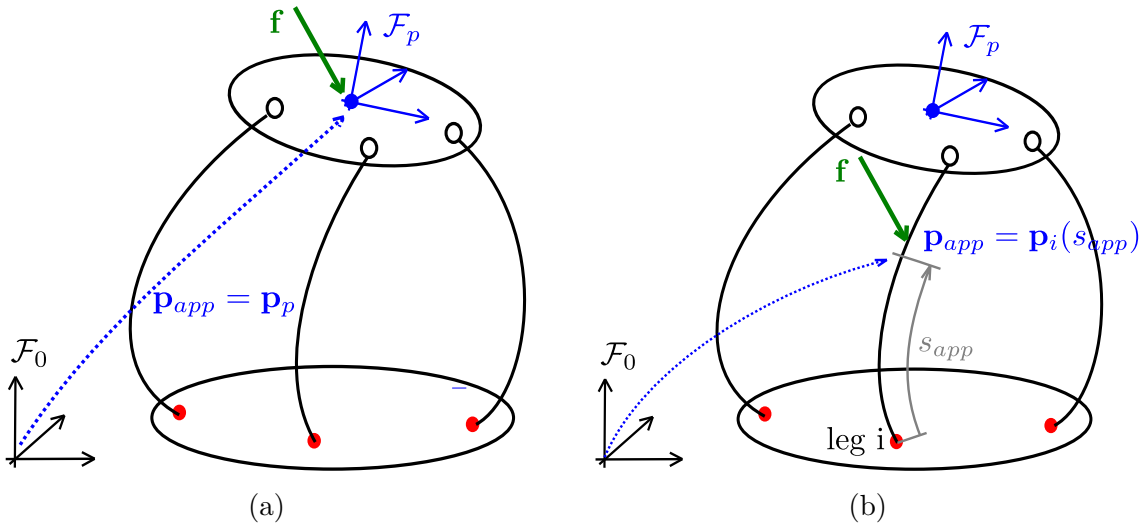


Figure 6.1: Illustration of the application point $\mathbf{p}_{app}$. (a) force applied at the platform centre ($\mathbf{p}_p$), (b) force applied at the i-th beam on a generic coordinates s_{app} ($\mathbf{p}_i(s_app)$

position vector $\mathbf{p}_{app}$, whose energetic contribution is:

$$V_f = -\mathbf{f}^T \mathbf{p}_{app} \tag{6.2}$$

For instance, when the force is applied to the *EE*, $\mathbf{p}_{app} = \mathbf{p}_p$ (Fig. 6.1a) while, if $\mathbf{f}$ is applied to a specific point of the i-th beam, $\mathbf{p}_{app} = \mathbf{p}_i(s_{app})$ (Fig. 6.1b). For the derivation of the proposed index, it is convenient to introduce f and $\mathbf{d}_f$, which are the magnitude and the direction unitary vector of $\mathbf{f}$, respectively. Thus, Eq. (6.2) becomes:

$$V_f = -\mathbf{f}^T \mathbf{p}_{app} = -f \mathbf{d}_f^T \mathbf{p}_{app} = -fh \tag{6.3}$$

with $h = \mathbf{d}_f^T \mathbf{p}_{app}$. Thus, the term V_f is added into the total potential energy V_{tot}:

$$V_{tot} = \sum_{i=1}^{n_b} V_{beams_i} + V_p + V_f \tag{6.4}$$

Once the total potential energy is defined, the derivation of the *CRs* geometrico-static model is performed as reported in Chapter 2: $\mathbf{q}_a \in \mathbb{R}^n$ collects the n actuated variables, $\mathbf{q}_c$ the n controlled variables, $\mathbf{q}_u$ the uncontrolled platform variables. Then, V_{tot} of Eq. (6.4)) is discretized with a finite set of m variables $\mathbf{q}_e \in \mathbb{R}^m$, and vector $\mathbf{x} = [\mathbf{q}_e, \mathbf{q}_u, \mathbf{q}_c] \in \mathbb{R}^{m+n_c}$ is defined accordingly to Sec. 2.2.3, where n_c is 3 for the planar case, and 6 for the spatial case. Since the goal is to evaluate the influence of an external load on the equilibrium stability for fixed motor values, a solution to the forward problem is computed by considering Lagrange conditions and Eq. (2.21), here reported for clarity:

$$\mathbf{F} = \begin{cases} \nabla_{\mathbf{x}} V_{tot} + \nabla_{\mathbf{x}} \mathbf{\Phi}^T \boldsymbol{\lambda} = \mathbf{0} \\ \mathbf{\Phi} = \mathbf{0} \\ \mathbf{q}_a - \mathbf{q}_a^d = \mathbf{0} \end{cases} \tag{6.5}$$

with $\mathbf{q}_a^d$ desired actuation values, and $\boldsymbol{\lambda} \in \mathbb{R}^{n_\phi}$ being Lagrange multipliers which enforce the constraints $\mathbf{\Phi}$. The Jacobian matrix of $\mathbf{F}$ that can be supplied to the

solver to speed up the computation was previously defined in Eq. (2.24), and it is here reported for later convenience:

$$\mathbf{J} = \frac{\partial \mathbf{F}}{\partial \mathbf{y}} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{U}_1 & \mathbf{P}_1 & \mathbf{\Lambda}^T \\ \mathbf{A}_2 & \mathbf{U}_2 & \mathbf{P}_2 & \mathbf{0} \\ \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{6.6}$$

Once a solution of Eq. (6.5) is found, the equilibrium stability is checked by verifying the positive definiteness of $\mathbf{H}^r$ introduced in Sec. 2.2.4:

$$\mathbf{H}^r = \mathbf{Z}^T \mathbf{H} \mathbf{Z}; \quad \mathbf{H}^r \in \mathbb{R}^{n_z \times n_z} \tag{6.7}$$

where $n_z = n_c + m - n_\phi$, matrix $\mathbf{Z}$ is provided in Eq. (2.28), and $\mathbf{H}$ in Eq. (2.33). In practice, equilibrium stability is assessed by verifying that all eigenvalues of $\mathbf{H}^r$ are strictly positive. Being σ_k the k-th eigenvalue of $\mathbf{H}^r$, the configuration is stable if the minimum eigenvalue σ_{min} is positive:

$$\sigma_{min} = \min_k(\sigma_k) > 0 \tag{6.8}$$

6.2 Directional Critical Load Index

This section derives the *DCLI* focus of this chapter. First, the distance to stability index is originally proposed in Section 6.2.1. Then, the index computation is discussed in Section 6.2.2, while Section 6.2.3 illustrated how to practically compute $\mathbf{Z}$ of Eq. (2.28). In order to assess the benefits and limitations of the *DCLI* performance quantification, Section 6.2.4 proposes a numerical approach that, at a greater computational cost than *DCLI*, calculates the exact load that causes instability.

6.2.1 Distance to Instability Index

As previously mentioned, the proposed index measures the distance from instability by investigating the influence of an external load $\mathbf{f}$ on the equilibrium stability of a configuration obtained by the solution of Eq. (6.5), and thus for fixed motor values[1] The force direction $\mathbf{d}_f$ and the application point $\mathbf{p}_{app}$ are considered as *known and fixed* (see Eq. (6.3)), and the goal is to compute the force magnitude f that causes instability in a given direction. The selection of $\mathbf{d}_f, \mathbf{p}_{app}$ is guided by the scope of the robot in order to consider possible scenarios where an external load is applied to the robot (e.g. an *EE* load during a pick and place operation, or a possible load on the *CPR* leg during a contact). Nevertheless, it is possible to compute exactly the critical load that causes instability by varying f and solving the *FGSP* until a value of f that causes instability is met. This approach is computationally expensive, and it will be discussed in Sec. 6.2.4. Instead, the scope of this chapter is to provide an index approximating the exact value of f with further computational efforts.

[1]It is possible also to measure the influence of an external load on the equilibrium stability of configurations obtained by the solution of the *IGSP*. However, there is little practical interest in the investigation of the case where the values of some controlled variables are assigned: maintaining fixed the value of controlled variables when varying f in practical cases is not trivial as it requires complex force-estimation techniques [20]. Instead, it is more frequent for practical applications in the case of assigned motor values. Thus, only the *FGSP* solutions are considered in the following.

To do this, let us consider Eq. (6.8). The value of σ_k for a given $f = f_k^\star$, namely $\sigma_k^\star$, can be approximated by first-order Taylor's expansion of σ_k around a generic f:

$$\sigma_k^\star = \sigma_k + S_k(f_k^\star - f) \tag{6.9}$$

where S_k can be obtained as [174]:

$$S_k = \frac{d\sigma_k}{df} = \boldsymbol{\nu}_k^T \frac{d\mathbf{H}^r}{df} \boldsymbol{\nu}_k \tag{6.10}$$

and $\boldsymbol{\nu}_k$ is the eigenvector associated with the k-th eigenvalue of $\mathbf{H}^r$, namely σ_k. Please note that σ_k, S_k are computed for a given value of f, but their dependence on f is dropped for simplicity's sake. To estimate the force magnitude for which instability occurs, let us compute the value of $f_k^\star$ for which $\sigma_k^\star = 0$. This is obtained by fixing $\sigma_k^\star = 0$ in Eq. (6.9), and rearranging its terms as:

$$f_k^\star = f - \frac{\sigma_k}{S_k} \tag{6.11}$$

$f_k^\star$ represents a value of f for which, approximately, at least one eigenvalue of $\mathbf{H}^r$ is zero, and thus the matrix is not positive definite anymore. By using Eq. (6.11) for each k, n_z different values of $f_k^\star$ are obtained. Thus, the *directional critical load index* ($DCLI$) is defined as:

$$DCLI = \min_k \left(\frac{\sigma_k}{S_k} \right) \tag{6.12}$$

$DCLI$ represents the smallest additional magnitude of f that causes a zero eigenvalue and, consequently, a limit of the stable equilibrium. $DCLI$ has a well-defined unit (Newton), and it can be used to measure and physically understand the distance from the instability, where larger values indicate greater distance. Moreover, it should be stressed that $DCLI$ is directional since $\mathbf{d}_f$ is *known and fixed*, and $DCLI$ represents the additional load applied at $\mathbf{p}_{app}$ in a given direction $\mathbf{d}_f$ that causes instability.

6.2.2 Index Computation

This Section discusses how to compute the $DCLI$ in practice, as several steps are necessary, and differentiating $\mathbf{H}^r$ is not straightforward. The required steps to calculate the $DCLI$ are schematically summarized in Alg. 5, and the detailed methodology for computing its terms is shown in the following. For a given external load, the solution of Eq. (6.5) gives the CR configuration. However, the resulting configuration depends on the value of f that is, $\mathbf{y} = \mathbf{y}(f)$. Additionally, after a solution to Eq. (6.5) is found, matrix $\mathbf{J}$ is obtained from Eq. (6.6), and $\boldsymbol{\Lambda}$ and $\mathbf{H}$, are extracted as blocks of $\mathbf{J}$ without further computations. Then, $\mathbf{Z}$ can be computed from $\boldsymbol{\Lambda}$, and $\mathbf{H}^r$ is obtained from Eq. (6.7). It is noteworthy that even though there is an infinite possibility of computing $\mathbf{Z}$, a specific one is required for properly computing $DCLI$, as it will detailed in Sec. 6.2.3. To check the positive definiteness of $\mathbf{H}^r$, an eigenvalue decomposition is performed to get i) the vector $\boldsymbol{\sigma}$ collecting n_z eigenvalues, and ii) the matrix $\mathbf{V}$ whose columns are n_z eigenvector. Suppose the equilibrium is stable (verified by Eq. (6.8)): in that case, the index computation continues. If the configuration is unstable, the $DCLI$ is not defined.

Algorithm 5: *DCLI* computation.

$[\mathbf{y}, \mathbf{J}]$ = Solve Geometrico-Static Problem;
Extract $\mathbf{\Lambda}, \mathbf{H}$ from $\mathbf{J}$;
$\mathbf{Z}$ = NullspaceComputation($\mathbf{\Lambda}$);
Compute $\mathbf{H}^r = \mathbf{Z}^T\mathbf{H}\mathbf{Z}$;
$[\boldsymbol{\sigma}, \mathbf{V}]$ = Eigenvalue decomposition of $\mathbf{H}^r$;
if *Equilibrium is Stable* **then**
    Compute $\mathbf{C}$ and $\frac{\partial \mathbf{y}}{\partial f} = -\mathbf{J}^{-1}\mathbf{C}$;
    Compute $\frac{\partial \mathbf{H}}{\partial f}$, $\frac{\partial \mathbf{Z}}{\partial f}$, $\frac{\partial \mathbf{H^r}}{\partial f}$;
    for $k = 1{:}\ n_z$ **do**
        $\sigma_k = \boldsymbol{\sigma}(k)$, $\boldsymbol{\nu}_k = \mathbf{V}(:, k)$;
        Compute $S_k = \boldsymbol{\nu}_k^T \frac{\partial \mathbf{H}^r}{\partial f} \boldsymbol{\nu}_k$;
        $\mathbf{f}^\star(k) = \frac{\sigma_k}{S_k}$
    end
    $DCLI = \min_k(\mathbf{f}^\star)$
else
    $DCLI = 0$;
end

Then, Eq. (6.10) requires computing $d\mathbf{H}^r/df$: a finite-difference approximation may be used as a straightforward solution (see [145], Chapter 8, Section 1), but an analytical formulation for $d\mathbf{H}^r/df$ can also be derived when using discretized robot model equations, as shown in the following. Employing a finite-difference approximation for $d\mathbf{H}^r/df$ is simple but time-consuming, and, depending on the selected finite-difference approximation strategy, multiple *FGSP* solutions are required at the cost of higher computational time. Instead, an analytical formulation of $d\mathbf{H}^r/df$ is preferred when *DCLI* has to be computed several times, such as for workspace characterization. To obtain an analytical formulation of $d\mathbf{H}^r/df$, the use of the product derivative rule on Eq. (6.7) results in:

$$\frac{d\mathbf{H}^r}{df} = \frac{d\mathbf{Z}^T}{df}\mathbf{H}\mathbf{Z} + \mathbf{Z}^T\frac{d\mathbf{H}}{df}\mathbf{Z} + \mathbf{Z}^T\mathbf{H}\frac{d\mathbf{Z}}{df} \tag{6.13}$$

The differentiation of $\mathbf{H}$ is addressed here first, and the computation of $d\mathbf{Z}/df$ is addressed in Sec. 6.2.3. Matrix $\mathbf{H}$ is computed after the solution of Eq. (6.5) and, in general, $\mathbf{H}$ depends on $\mathbf{y}(f)$ and f, that is:

$$\mathbf{H} = \mathbf{H}(\mathbf{y}(f), f) \tag{6.14}$$

Consequently, the total derivative of $\mathbf{H}$ w.r.t. f is obtained as the sum of two terms:

$$\frac{d\mathbf{H}}{df} = \frac{\partial \mathbf{H}}{\partial f} + \sum_{i=1}^{n+n_c+m+n_\phi} \frac{\partial \mathbf{H}}{\partial y_i}\frac{\partial y_i}{\partial f} \tag{6.15}$$

Since $\mathcal{L} = V_{tot} + \mathbf{\Phi}^T\boldsymbol{\lambda}$ and V_f only explicitly depends on f (see Eq. (6.3)), the first term of Eq. (6.15) simplifies as follows:

$$\frac{\partial \mathbf{H}}{\partial f} = \frac{\partial}{\partial f}\left(\frac{\partial^2 \mathcal{L}}{\partial \mathbf{x}\partial \mathbf{x}}\right) = -\left(\frac{\partial^2 h}{\partial \mathbf{x}\partial \mathbf{x}}\right) \tag{6.16}$$

where $h = \mathbf{d}_f^T \mathbf{p}_{app}$ is previously defined in Eq. (6.3). Instead, the term $\partial \mathbf{H}/\partial y_i$ can be computed analytically, and its expression depends on the specific discretization technique employed. Due to its lengthy expression, its formulation is reported in Appendix C in the case of the finite-difference modelling approach of Sec. 2.3.2 being used.

To compute the second term of Eq. (6.15), it is necessary to evaluate $\partial \mathbf{y}/\partial f$, and the implicit functions theorem is used for the scope. Let us consider Eq. (2.21): $\mathbf{F}$ is a set of equations in the unknowns $\mathbf{y}$ and dependent on the parameter f. $\mathbf{F}$ is assumed to be a set of continuous and differentiable functions w.r.t. $\mathbf{y}, f$. Given a pair $(\mathbf{y}, f)$ that satifies $\mathbf{F}(\mathbf{y}, f) = \mathbf{0}$, and assuming $\partial \mathbf{F}/\partial \mathbf{y}$ full rank, there exists a unique function $\mathbf{r} : \mathbb{R} \to \mathbb{R}^{n+n_c+m+n_\phi}$ such as $\mathbf{y} = \mathbf{r}(f)$. Moreover, the partial derivative of $\mathbf{r}$ w.r.t. f (and thus $\partial \mathbf{y}/\partial f$) is given by:

$$\frac{\partial \mathbf{r}}{\partial f} = \frac{\partial \mathbf{y}}{\partial f} = -\left(\frac{\partial \mathbf{F}}{\partial \mathbf{y}}\right)^{-1} \frac{\partial \mathbf{F}}{\partial f} = -\mathbf{J}^{-1}\mathbf{C} \tag{6.17}$$

where $\mathbf{J}$ is defined in Eq. (6.6), and $\mathbf{C} = \partial \mathbf{F}/\partial f = [-\nabla_{\mathbf{x}} h; \mathbf{0}]$.

6.2.3 Nullspace Computation

This subsection introduces a specific computation methodology of $\mathbf{Z}$, which allows for a streamlined derivation of $\partial \mathbf{Z}/\partial f$, that is ultimately needed in Eq. (6.13). Nullspace bases are frequently computed using numerical techniques, such as singular value decomposition. The resulting nullspace basis is orthonormal, that is, $\mathbf{Z}^T\mathbf{Z} = \mathbf{I}$ and $\mathbf{Z}$ is dense. However, since $\mathbf{Z}$ is obtained numerically, this approach leads to a cumbersome derivation of $\partial \mathbf{Z}/\partial f$, which depends on the specific numerical algorithm employed, and which may not ensure continuity of $\mathbf{Z}$ with respect to variations of f [175]. On the other side, fundamental nullspace basis [176] offers a non-orthonormal alternative that can be computed analytically. Thus, the analytical formulation of $\mathbf{Z}$ enables the possibility to calculate $\partial \mathbf{Z}/\partial f$ more easily.

To get a fundamental basis of $\mathbf{Z}$, it is necessary to permute the columns of $\mathbf{\Lambda}$ to obtain $\mathbf{\Lambda}_p = \mathbf{\Lambda}\mathbf{P}$, where $\mathbf{P} \in \mathbb{R}^{(m+n_c)\times(m+n_c)}$ is a permutation matrix. The scope of $\mathbf{P}$ is to get a full rank and well-conditioned matrix $\mathbf{\Lambda}_d \in \mathbb{R}^{n_\phi \times n_\phi}$ from:

$$\mathbf{\Lambda}_p = \begin{bmatrix} \mathbf{\Lambda}_d & \mathbf{\Lambda}_u \end{bmatrix} \tag{6.18}$$

with $\mathbf{\Lambda}_u \in \mathbb{R}^{n_\phi \times n_z}$. Matrix $\mathbf{P}$ can be found by inspecting $\mathbf{\Lambda}$ and identifying a set of n_ϕ linearly independent columns that form a well-conditioned $\mathbf{\Lambda}_d$. However, when the dimension of $\mathbf{\Lambda}$ increases, no trivial full-rank partition is available in general. As stated in [177], any choice of $\mathbf{P}$ ensuring $\mathbf{\Lambda}_d$ full-rank is adequate and, by randomly scanning the columns of $\mathbf{\Lambda}$ until a well-conditioned partition is found, matrix $\mathbf{P}$ can be computed. However, the computational cost of this strategy drastically increases with n_ϕ. For instance, when using finite-difference techniques for the geometrico-static modelling, the quaternion-unitarity constraints must be enforced for each beam's cross-section, leading to a large value of n_ϕ.

To overcome this issue, a heuristic approach is proposed in the following for the computation of $\mathbf{P}$. However, other approaches may be equivalently proposed for the scope. The proposed approach requires scanning n_ϕ times the columns of $\mathbf{\Lambda}$, aiming to determine a matrix $\mathbf{P}$ that maximizes the inverse conditioning of $\mathbf{\Lambda}_d$. A

Algorithm 6: Permutation Matrix Computation

```
1  Function [P] = GetPermutation(Λ):
2      [a, b] = size Λ;
3      P = 0, Λ_d = ∅;
4      for i = 1 : a do
5          R_v = 0;
6          for j = 1 : b do
7              Λ_m = [Λ_d(1 : i, :), Λ(1 : i, j)];
8              R_v(j) = inverse conditioning of Λ_m;
9          end
10         idx = column of Λ where max of R_v occurs;
11         Λ_d = [Λ_d, Λ(:, idx)];
12         Set P(i, idx) = 1;
13         Set b = b − 1;
14         Remove Λ(:, idx) from Λ;
15     end
16     return
```

pseudocode of the algorithm is reported in (Alg. 6). First, the algorithm starts by initializing $\mathbf{P} = \mathbf{0}$ and $\mathbf{\Lambda}_d = \varnothing$. The goal is to select n_ϕ columns of $\mathbf{\Lambda}$ to create a full-rank and well-conditioned $\mathbf{\Lambda}_d$. The algorithm starts by scanning the first row of $\mathbf{\Lambda}$ to select the term with the higher inverse conditioning, and the corresponding column of $\mathbf{\Lambda}$ is selected. This column (labelled with idx) is inserted in $\mathbf{\Lambda}_d$ and removed from $\mathbf{\Lambda}$ to avoid repetitions. Matrix $\mathbf{P}$ is updated accordingly to put the column idx as the first column of $\mathbf{\Lambda}_d$. Then, the second row is considered. For each column of $\mathbf{\Lambda}$, a (2×2) matrix $\mathbf{\Lambda}_m$ is obtained by collecting $\mathbf{\Lambda}_d$ and the considered column of $\mathbf{\Lambda}$ (see line 12 of Alg. 6). The second column to be put in $\mathbf{\Lambda}_d$ is the one that maximises the inverse conditioning of $\mathbf{\Lambda}_m$. Matrix $\mathbf{P}$ is updated to put the selected column as the second of $\mathbf{\Lambda}_d$. The algorithm proceeds in the same fashion for the next rows by building $\mathbf{\Lambda}_m$, selecting columns that maximise the inverse conditioning of $\mathbf{\Lambda}_m$, and creating $\mathbf{P}$ consequently. The algorithm stops when all the n_ϕ rows of $\mathbf{\Lambda}$ have been considered.

As long as $\mathbf{\Lambda}_d$ is full rank, a fundamental nullspace basis of $\mathbf{\Lambda}_p$ is obtained as:

$$\mathbf{Z}_p = \begin{bmatrix} \mathbf{Z}_d \\ \mathbf{I}_{n_z} \end{bmatrix} \tag{6.19}$$

with $\mathbf{Z}_d = -\mathbf{\Lambda}_d^{-1}\mathbf{\Lambda}_u$. Finally, $\mathbf{Z}$ is obtained by permutating the columns of $\mathbf{Z}_p$ as done for the rows of $\mathbf{\Lambda}_p$, that is $\mathbf{Z} = \mathbf{P}\mathbf{Z}_p$.

To analytically calculate $d\mathbf{Z}/df$, the first step requires to compute $d\mathbf{\Lambda}/df$. Since $\mathbf{\Lambda}$ depends on $\mathbf{y}(f)$ and not on f explicitly, $d\mathbf{\Lambda}/df$ is expressed as follows:

$$\frac{d\mathbf{\Lambda}}{df} = \sum_{i=1}^{m+n_c+N_e+n_\phi} \frac{\partial \mathbf{\Lambda}}{\partial y_i}\frac{\partial y_i}{\partial f} \tag{6.20}$$

The expression of $\partial\mathbf{\Lambda}/\partial y_i$ depends on the specific modelling strategy, and its detailed expression is reported in Appendix C.1 for the case of a finite-differences approximation. Then the derivative of $\mathbf{\Lambda}_d, \mathbf{\Lambda}_u$ w.r.t. f are obtained by using $\mathbf{P}$ as

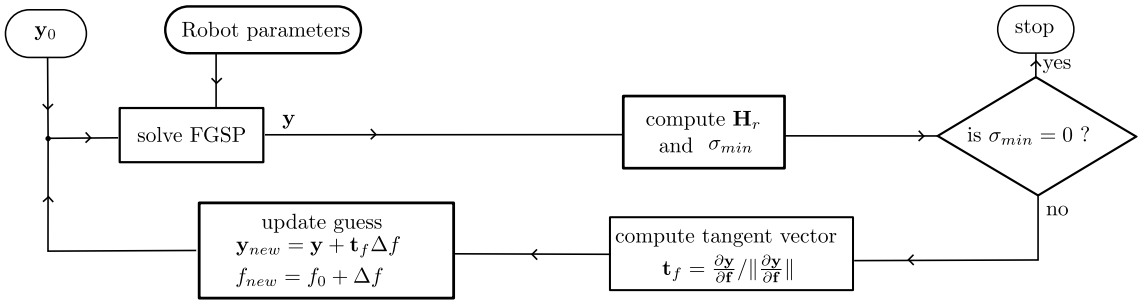


Figure 6.2: Schematics of the numerical approach for the computation of the exact critical load.

follows:

$$\frac{d\mathbf{\Lambda}_p}{df} = \frac{d\mathbf{\Lambda}}{df}\mathbf{P} = \left[\frac{d\mathbf{\Lambda}_d}{df},\ \frac{d\mathbf{\Lambda}_u}{df}\right] \tag{6.21}$$

The term $d\mathbf{Z}_p/df$ is obtained by deriving Eq. (6.19) w.r.t. f:

$$\frac{d\mathbf{Z}_p}{df} = -\begin{bmatrix} \frac{d\mathbf{\Lambda}_d^{-1}}{df}\mathbf{\Lambda}_u + \mathbf{\Lambda}_d^{-1}\frac{d\mathbf{\Lambda}_u}{df} \\ \mathbf{0} \end{bmatrix} \tag{6.22}$$

where $d\mathbf{\Lambda}_d^{-1}/df = \mathbf{\Lambda}_d^{-1}\left(d\mathbf{\Lambda}_d/df\right)\mathbf{\Lambda}_d^{-1}$. Finally, $d\mathbf{Z}/df$ is recovered as $d\mathbf{Z}/df = \mathbf{P}d\mathbf{Z}_p/df$.

6.2.4 Exact Directional Critical Load Computation

The *DCLI* estimates the magnitude of an external load for which instability occurs. Except for a few cases where analytical results are available (such as initially straight beams), the exact load is computed numerically. This section proposes a numerical approach to compute the exact critical load f_{CRIT}, based on the schematics proposed in Fig. 6.2. This approach, which requires a higher computational cost than *DCLI*, will be used to quantify how well the *DCLI* approximated the exact f_{CRIT}.

This section aims to propose a strategy for the identification of the exact value of f for which $\mathbf{H}^r$ has a null eigenvalue, that is, $\sigma_{min} = 0$. For the scope, starting from $f = 0$, the value of f is gradually incremented with a fixed increment δf until instability occurs. First, the *FGSP* is solved with a given initial guess $\mathbf{y}_0$ and $f_0 = 0$ to get the configuration $\mathbf{y}$. Then, $\mathbf{H}_r$ is built from the output of the *FGSP*, and σ_{min} is computed. If $\sigma_{min} \neq 0$, f_0 is incremented of a user-defined quantity δf, that is:

$$f_{new} = f_0 + \delta f \tag{6.23}$$

After the force update, the algorithm restarts by repeating the *FGSP* solution, and an initial guess for the robot configuration is required. The previous *FGSP* solution may be used as an initial guess, but it is convenient to update better $\mathbf{y}$ accordingly to f_{new}. In this way, at the next iteration, the convergence of the solver is faster since the given initial guess is in accordance with the new value of f. For the scope, the tangent vector $\mathbf{t}_f$ is defined as:

$$\mathbf{t}_f = \frac{\partial \mathbf{y}}{\partial f} \Big/ \left\| \frac{\partial \mathbf{y}}{\partial f} \right\| \tag{6.24}$$

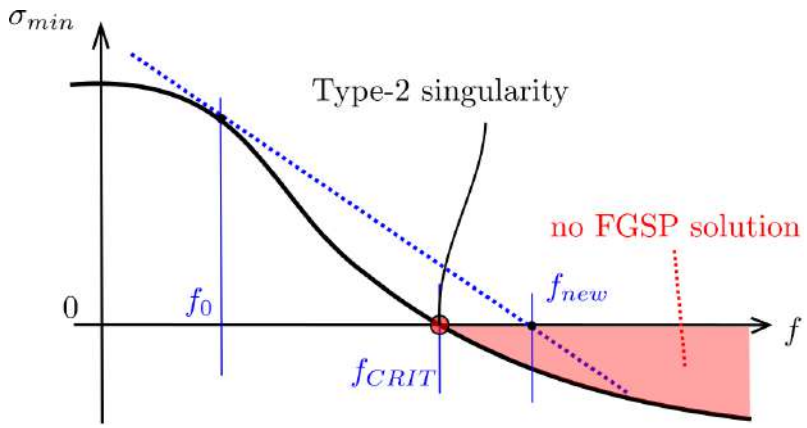


Figure 6.3: Illustration of when solving $\sigma_{min} = 0$ with a root-finding technique may not be effective.

where $\frac{\partial \mathbf{y}}{\partial f}$ is obtained by using Eq. (6.17). Then, $\mathbf{t}_f$ is used to update $\mathbf{y}$ accordingly to δf:

$$\mathbf{y}_{new} = \mathbf{y} + \mathbf{t}_f \delta f \tag{6.25}$$

The values $\mathbf{y}_{new}, f_{new}$ are used as $\mathbf{y}_0, f_0$ to solve again the *FGSP*. The algorithm is repeated until a value of f is found so that $\sigma_{min} = 0$, or the maximum number of allowed iterations is reached.

The proposed numerical approach is iterative and, depending on the choice of δf, several *FGSP* solutions are required with a consequent increase in the overall computational cost for the identification of f_{CRIT}. An alternative numerical approach, aiming at a more efficient computation of f_{CRIT}, employs a nonlinear root-finding technique (e.g. the Newton method) to identify the value of f for which $\sigma_{min} = 0$, with σ_{min} being computed at each iteration by solving the *FGSP*. Although this approach seems to be more efficient, it may be ineffective: let us consider for simplicity a Newton method for the solution of $\sigma_{min} = 0$. As shown in Fig. (6.3), starting from f_0, the force update may select a new value of f_{new} that corresponds to unstable regions where no *FGSP* solution is numerically reachable. As illustrated in [56], passing from positive to negative σ_{min}, there exists a value of f for which $\sigma_{min} = 0$, that is, f_{CRIT}. This value of f is a stable-to-unstable transition corresponding to a Type-2 singularity (see Eq. (2.32)), also defining a limit of the *FGSP* solvability. Thus, it may happen that no solution of the *FGSP* is found and incorrect values of f_{CRIT} are predicted. For this reason, a constant and limited update of f by using δf is preferred rather than using nonlinear rootfinding techniques.

6.3 Case Studies

This Section proposes three different case studies: initially straight beams and their elastic buckling (Sec. 6.3.1), a two-tubes *CTR* with four controlled *DoFs* (Sec. 6.3.2), and a spatial *CPR* with two controlled *DoFs* (Sec. 6.3.3). These case studies are selected to illustrate how the proposed formulation of the *DCLI* can be applied to different scenarios (passive elements, serial *CRs*, and parallel *CRs*). Finite differences are used [56] as discretization techniques to obtain the geometrico-static model of Eq. (2.23) for each case study. However, any other discretization technique can

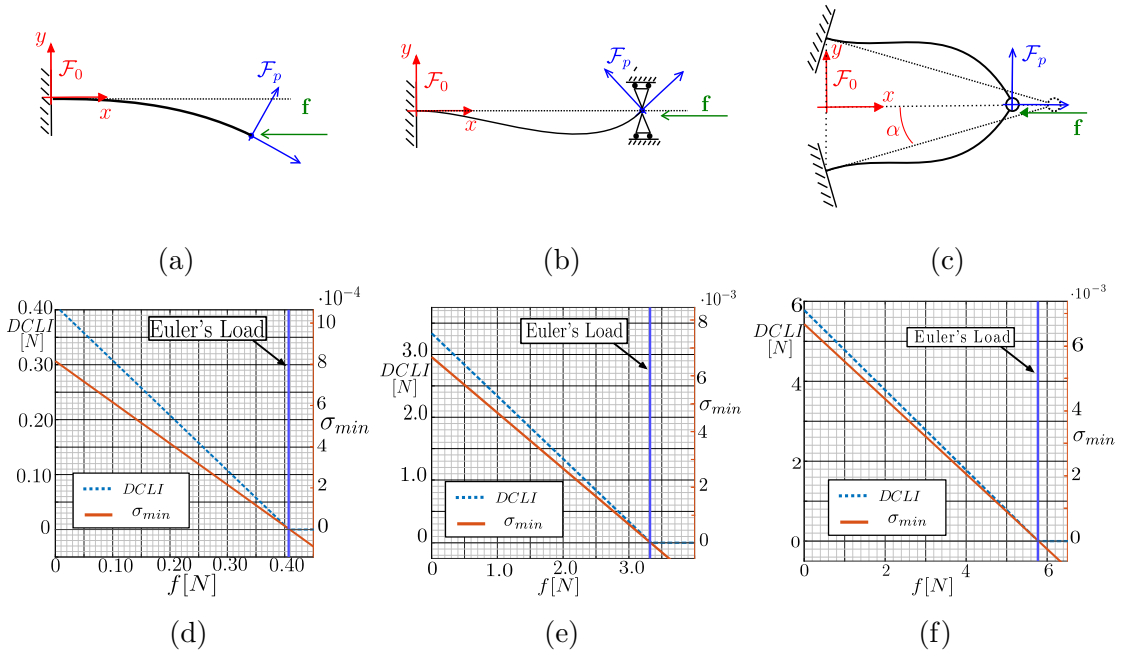


Figure 6.4: Buckling of straight beams: (a) clamped beam with free distal section, (b) clamped beam with pinned distal section, and (c) two clamped beams connected by a passive revolute joint. Then, the values of the *DCLI* (dotted line) and σ_{min} (continuous line) are displayed by varying f for the scenario (a),(b),(c) in in (d),(e),(f), respectively. The decreasing trend of *DCLI* when approaching the critical load confirms the correctness of the distance-to-instability measurement.

be used. Even though finite differences do not offer the best performances in terms of computational time [128], the analytical formulation of Eq. (2.23) considerably simplifies the computation of Equations (6.15) and (6.20).

6.3.1 Buckling of beams

This Section proposes the analysis of initially straight beam instability and the comparison of the *DCLI* with Euler's critical for the beams buckling to verify the correctness of the equilibrium stability prediction and the *DCLI*. For each beam, *DCLI* has been computed by performing a finite-differences discretization with $N_e = 200$. As shown in [80], a planar beam $N_e \geqslant 50$ ensures sufficient accuracy in the geometrico-static problems solution. The early work of Euler [94] defined analytical conditions for the buckling of ideal elastic beams subjected to axial loads. For each considered beam, $L = 1$ m, the cross-section is circular with diameter 2 mm, and $E = 210$ GPa.

First, let us consider a clamped-free beam as illustrated in Fig. 6.4a: the beam is clamped at the proximal section, initially straight, and parallel to the fixed-frame x axis. Euler's buckling formula provides the value of the axial force magnitude f to be applied at the tip of the beam to cause elastic instability. Under the assumptions that shear and extensibility are negligible, the Euler's critical load f_{EUL} for a clamped-free beam can be computed as:

$$f_{EUL} = \frac{EI\pi^2}{(2L)^2} \tag{6.26}$$

for the selected beams parameters, f_{EUL} results in 0.407 N. When no load is applied at the beam, $DCLI = 0.407$ N in accordance with f_{EUL}, and the difference between $DCLI$ and f_{EUL} is negligible up to four digits.

Then, the exact critical load f_{CRIT} is computed by using the numerical algorithm proposed in Sec. 6.2.4. The force applied at the beam tip is gradually increased with a $\delta f = 0.001$N, and, for each step, σ_{min} and $DCLI$ are measured. The results of this computation are reported in Fig. 6.4d. The exact critical load value results in $f_{CRIT} = 0.407$ N, obtained in 407 steps (and thus 407 geometrico-static problem solutions). In particular, σ_{min} becomes negative and the equilibrium unstable when the Euler's load is reached, confirming the correctness of f_{CRIT}. $DCLI$ is computed at each step of the numerical algorithm, and $DCLI$ correctly measures the distance from instability: as shown in Fig. 6.4c, when increasing f, the value of $DCLI$ decreases, reaching zero at f_{EUL}. Moreover, for each value of f, the sum $DCLI + f$ is constant and equal to f_{EUL}: as the beam remains straight and undeformed at each step, the value of $DCLI + f$ truly represents the critical load. Similar results were obtained in [75], where the index based on an equivalent integration length gives the exact beam length for which instability should occur for a given load.

Then, let us consider the beam of Fig. 6.4b: the beam is clamped at the proximal section, initially straight, parallel to the fixed-frame x axis, and pinned at the distal section. In this case, the Euler's critical load is obtained as:

$$f_{EUL} = \frac{EI\pi^2}{(0.699L)^2} \tag{6.27}$$

and, with the selected beams parameters, $f_{EUL} = 3.322$ N and $DCLI = 3.322$. The difference between $DCLI$ and f_{EUL} is negligible up to four digits. The exact critical load f_{CRIT} is computed with the numerical algorithm proposed of Sec. 6.2.4: the value of $f_{CRIT} = 3.322$ is obtained with 3322 steps, with $\delta f = 0.001$ N. As shown in Fig. 6.4e. To confirm the correctness of f_{CRIT}, it is possible to note that σ_{min} becomes negative when the Euler's load is reached. As before, $DCLI$ tends toward zero when σ_{min} decreases, and $DCLI + f$ is constant and equal to f_{EUL}.

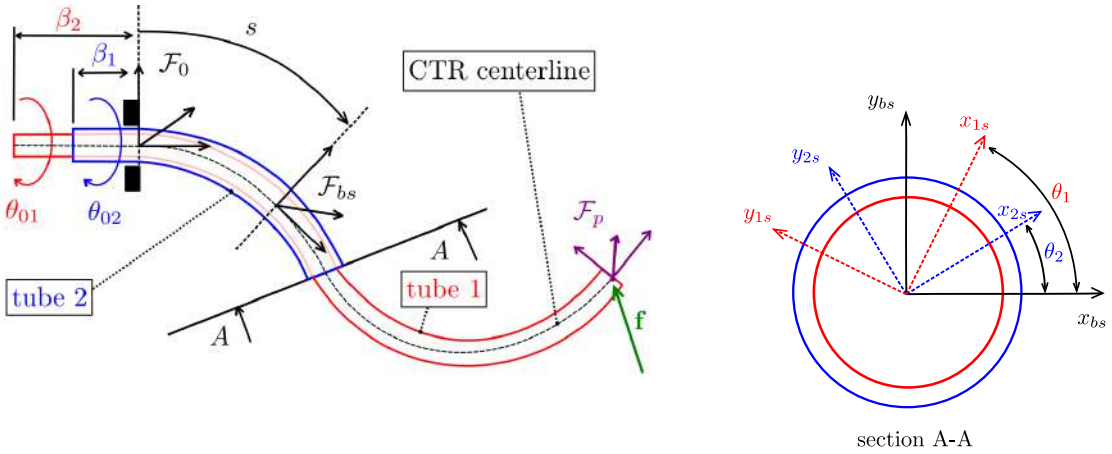
Finally, the case of two clamped beams connected by a passive revolute joint is considered (Fig. 6.4c). Beams are initially straight and arranged to form an angle α w.r.t. the x fixed-frame axis (see Fig. 6.4c), and a load $\mathbf{f} = [f, 0]$ is applied to the passive joint. According to Euler's formula, the critical load to be applied at the revolute joint to cause instability of at least one beam is:

$$f_{EUL} = 2cos(\alpha)\frac{EI\pi^2}{(0.699L)^2} \tag{6.28}$$

By considering the beams parameters and $\alpha = 30^\circ$ $DCLI = F_{crit} = 5.771$ N. The value of f_{CRIT} is computed with $\delta f = 0.001$ N, and the value of σ_{min} becomes negative exactly at the Euler's load (see Fig. 6.4f). Thus, $f_{CRIT} = 5.771$ N. $DCLI$ effectively measures the distance from the instability, reaching the zero value at the Euler's load. As before, $DCLI + f = f_{EUL}$.

6.3.2 A two tubes *CTR* with four controlled *DoFs*

This Section introduces the application of $DCLI$ for a two-tube CTR: $CTRs$ are a well-known class of CRs where instability occurs [93]. Even though the main focus



(a) (b)

Figure 6.5: a two-tube *CTR*. (a) relevant dimensions and variables, (b) a cross-section of the tubes to highlight the torsion angles.

of this thesis is *CPRs*, the formulation of this index also encompasses serial-like *CRs*, and Appendix E describes how to formulate geometrico-static problems of *CTRs* as in Eq. (2.23). The two-tube case is a simple but effective benchmark to test the *DCLI* since an analytical condition exists for the stability assessment. A *CTR* made by $n = 2$ concentric tubes is considered (Fig. 6.5a). A fixed frame $\mathcal{F}_0$ is attached to the robot base, the *CTR* centerline is parametrized with the coordinate s, and the index i represents the i-th tube, where $i = 1$ is the inner tube, and $i = 2$ is the outer tube. Tubes are of length L_i (measured from $s = 0$) and actuated at $s = -\beta_i$. The tubes are actuated in translation and rotation: θ_{0i} is the rotation of the tube's base, and β_i is called transmission length. The *CTR* energy is obtained by considering shear-less and inextensible tubes, and the discretization process is performed by using finite differences, with 50 points for *CTR* sections from 0 to L_2 and from L_2 to L_1.

As previously mentioned, the two-tube *CTR* is a well-known situation where analytical conditions exist for the global stability of the *CTR*. Assuming planar precurvature only ($\mathbf{u}_i^* = [u_{ix}^*, 0, 0]$), and no external load applied to the robot, the *CTR* equilibrium is globally stable if the following inequality is verified [93]:

$$\zeta_\gamma = \frac{\cot(\gamma)}{\sqrt{\gamma}} < \zeta_{lim} \tag{6.29}$$

where ζ_γ is computed by the knowledge of γ, defined as follows:

$$\gamma = L_2^2 u_{1x}^* u_{2x}^* \frac{k_{1b}k_{2b}(k_{1t} + k_{2t})}{k_{1t}k_{2t}(k_{1b} + k_{2b})} \tag{6.30}$$

The term k_{bi} is the flexural stiffness, and k_{ti} is the torsional stiffness of the i-th tube. Instead, the term ζ_{lim} of Eq. (6.29) is equal to zero in the case β_1 and β_2 are assumed to be zero. Equation (6.29) determines conditions for the global stability of two-tube *CTRs*, and *S-curves* were introduced to practically visualize the *CTR* motion abilities. *S-curves* describe the relationship between the base orientations and the resulting *CTR* tip orientation, providing an intuitive representation of the robot motion abilities [93]. First, let us define $\phi_0 = (\theta_{10} - \theta_{20})$ as the rotation offset

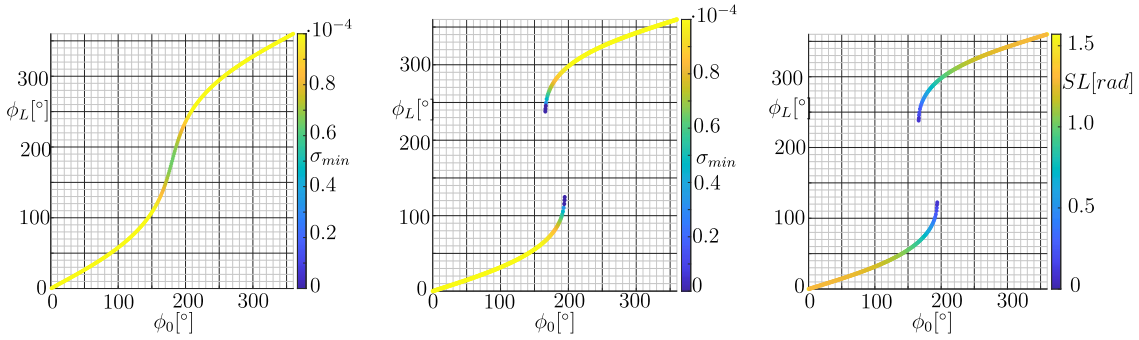


Figure 6.6: *S-curves* of a two-tube *CTR*. σ_{min} is depicted over a globally stable *S-curve*, and on a unstable case (b). Instead, for the same unstable case, (c) displays the value of the stability index of [104].

of the tubes at the base, while $\phi_L = (\theta_1(L_2) - \theta_2(L_2))$ is the rotation offset of the tubes at L_2. Then, *S-curves* are built by fixing $\theta_{20} = 0$, repeatedly solving the *FGSP* with $\phi_0 \in [0, 2\pi]$ and measuring ϕ_L for each ϕ_0.

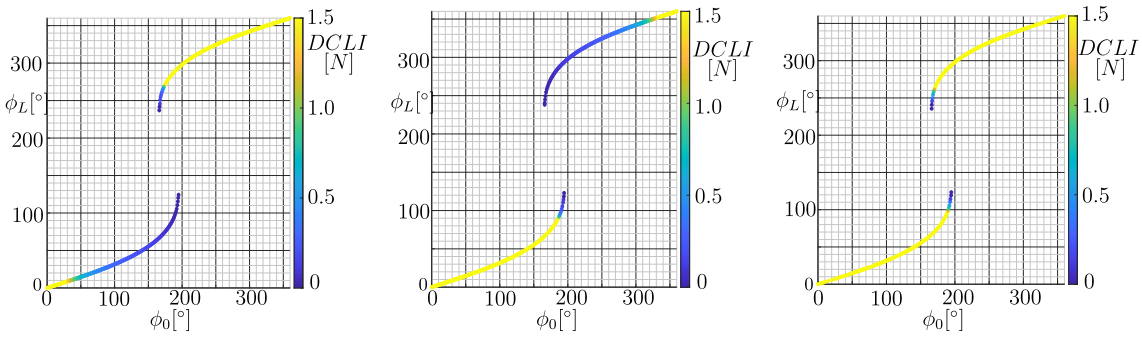
First, the correctness of equilibrium stability prediction is verified by computing *S-curves* with no load applied to the robot ($f = 0$). Tubes parameters are selected as $E = 80$ GPa, $u^*_{1x} = 1/0.50$ m^{-1}, $u^*_{2x} = 1/0.70$ m^{-1}. The first tube has inner diameter d_{1inn} =1.0 mm, and outer diameter d_{1out} =1.5 mm, while the second tube has d_{2inn} =1.5 mm, and d_{2out} = 2.0 mm. Computing the *S-curves* for $L_2 = 0.4$ m ensures globally stable equilibrium since $\zeta_\gamma = -0.164 < 0$. This is confirmed by the always positive value of σ_{min}, as shown in Fig. 6.6a. Instead, the *S-curves* for $L_2 = 0.6$ m violates the inequality of Eq. (6.29) ($\zeta_\gamma = +0.242 > 0$), and σ_{min} goes toward negative values (Fig. 6.6b). Moreover, the observation that *S-curves* of stable *CTRs* are monotonic (the slope of the curve is always positive) motivated [104] to consider the slope of *S-curves* as a stability metric. In particular, the stability metric based on the slope SL is defined as:

$$SL = \pi - \text{atan}\left(\frac{\partial \phi_L}{\partial \phi_0}\right) \tag{6.31}$$

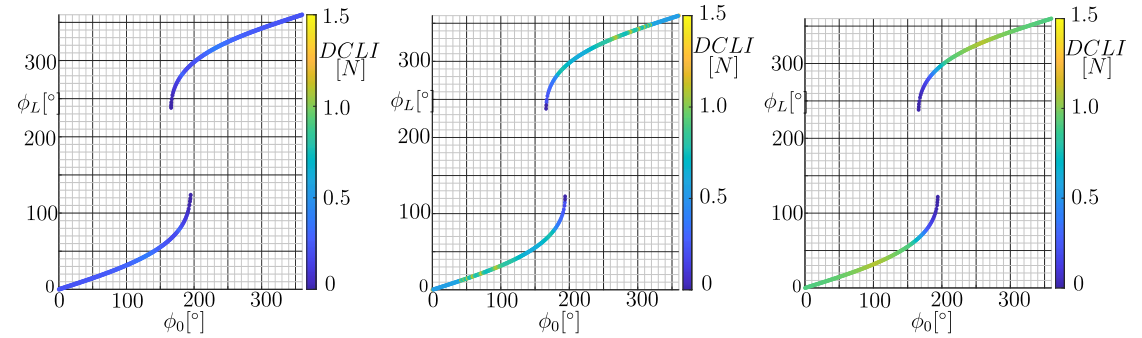
The index SL has a well-defined unit (radiants), $SL > 0$, as long as the equilibrium is stable and larger values of SL indicate greater distance from instability. Fig. 6.6c illustrates the values of SL for the case $L_2 = 0.6$ m, and these values will be later used to verify the correctness of $DCLI$ w.r.t. state-of-the-art indices, such as SL.

To compute $DCLI$, the influence of a tip load on the equilibrium stability is considered. The agreement of $DCLI$ with the state-of-the-art is checked by selecting the unstable case $L_2 = 0.6$ m, and computing the *S-curves* highlighting $DCLI$ over different directions $\mathbf{d}_f$ and no external load applied on the *CTR*. Figure 6.7 illustrates six *S-curves*: for each curve, $\mathbf{d}_f$ is aligned to one of the fixed-frame axes $\mathcal{F}_0$, considering both positive and negative directions. No matter the director $\mathbf{d}_f$ is chosen, $DCLI$ tends toward zero when the instability approaches. This is further confirmed by comparing the trend of SL (Fig. 6.6a) with $DCLI$: SL approaches null values when $DCLI$ tends to zero, confirming the coherence of $DCLI$ w.r.t. state-of-the-art results.

Then, the estimation provided by $DCLI$ is compared with the exact value of f_{CRIT} by computing the absolute error $e_{abs} = |f_{CRIT} - DCLI|$. For instance, let us



(a) $\mathbf{d}_f = +[1,0,0]$ (positive x) (b) $\mathbf{d}_f = -[1,0,0]$ (negative x) (c) $\mathbf{d}_f = +[0,1,0]$ (positive y)

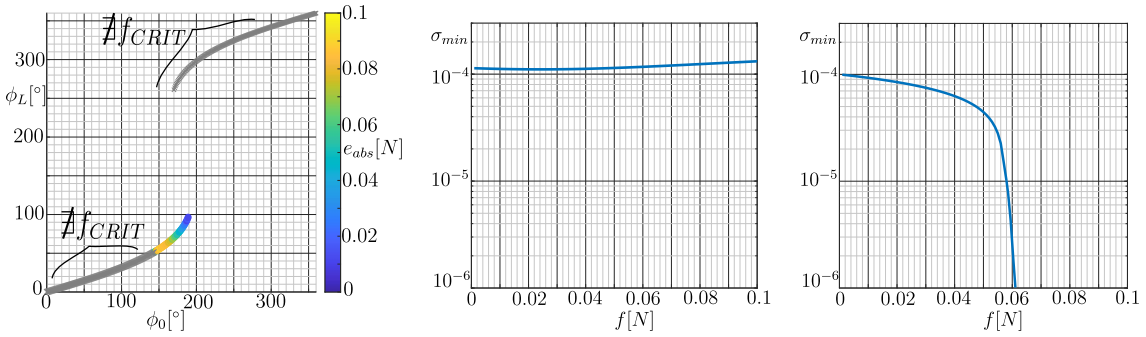


(d) $\mathbf{d}_f = -[0,1,0]$ (negative y) (e) $\mathbf{d}_f = +[0,0,1]$ (positive z) (f) $\mathbf{d}_f = -[0,0,1]$ (negative z)

Figure 6.7: For the same *S-curve*, different values of *DCLI* are obtained by changing $\mathbf{d}_f$. However, *DCLI* tends toward zero when approaching the instability, no matter the $\mathbf{d}_f$ is selected.

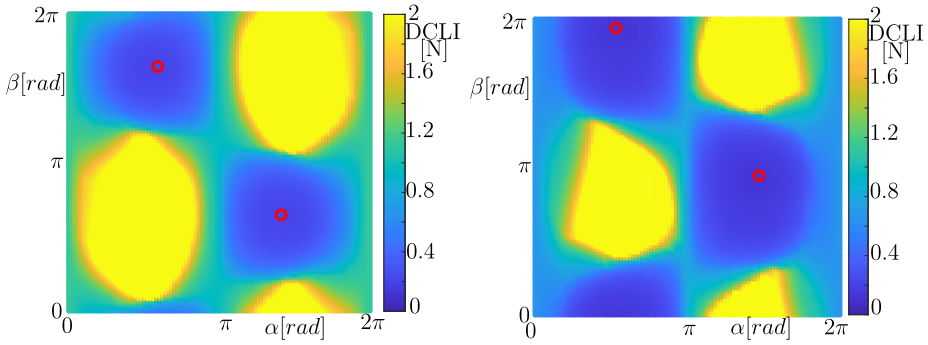
consider $\mathbf{d}_f = +[1,0,0]$, and the corresponding values of *DCLI* over the *S-curve* of Fig. 6.7a. For each point of the *S-curve*, f_{CRIT} is computed with $\delta f = 0.001$ N, and a maximum number of steps equal to 2000, thus exploring forces up to 2 N. The resulting value of e_{abs} is displayed in Fig. 6.8a, and two scenarios can be identified. In a first scenario, there exist no f_{CRIT} between [0,2] N: this happens between $\phi_0 \in [0, 140]^\circ$, and in the upper branch of the *S-curve*. As an example, Fig. 6.8b reports the trend of σ_{min} when using the numerical algorithm of Sec. 6.2.4 for $\phi_0 = 60^\circ$: the value of σ_{min} varies when changing f but it never goes to negative values. Instead, between $\phi_0 \in [140, 190]^\circ$, there exist values of $f_{CRIT} \in [0, 2]$ N: in this region, when increasing f, the value of σ_{min} reaches negative values, as illustrated in Fig. 6.8c for $\phi_0 = 160^\circ$. Thus, when there exists a load f_{CRIT} that causes instability, *DCLI* displays reduced values and, when approaching the stability limits on the *S-curve*, the difference between *DCLI* and f_{CRIT} reduces: this is confirmed by the value of e_{abs} which decreases when approaching the instability.

Instead, when *DCLI* is low, it does not necessarily means that f_{CRIT} is low. This is evident in the upper part of Fig. 6.8a where no $f_{CRIT} \in [0, 2]$ N exists, but *DCLI* is low in the proximity of the instability. However, it should be considered that the *S-curve* of Fig. 6.7 (and thus the values of *DCLI*) are computed with $f = 0$, and the instability phenomenon is happening as a cause of the torsional energy accumulated in the *CTR*, as described in [57]. When varying f during the computation of f_{CRIT}, the shape of the *S-curves* varies consequently, and it may happen that previously



(a) $\mathbf{d}_f = +[1, 0, 0]$ (b) $\phi_0 = 60°$ (c) $\phi_0 = 160°$

Figure 6.8: Comparison between f_{CRIT} and $DCLI$. Figure (a) displays the value of e_{abs} for the *S-curve* computed with $L_2 = 0.6$ and $\mathbf{d}_f = +[1, 0, 0]$. In grey zones, no f_{CRIT} exists between $[0, 2]$ N. Figure (b),(c) display the value of σ_{min} when using the numerical algorithm of Sec. 6.2.4, for $\phi_0 = 60°, 160°$, respectively.



(a) $\phi_0 = 60°$ (b) $\phi_0 = 160°$

Figure 6.9: Influence of the inclination angle α and the azimuth angle β of the tip force direction $\mathbf{d}_f$ on $DCLI$. Figure (a) is relative to $\phi_0 = 60°$, while figures (b) corresponds to $\phi_0 = 160°$. Minimums of $DCLI$ are highlighted by a red circle.

unreachable values of ϕ_0 becomes accessible: the application of the external load may increase the stability of some configurations. This aspect will be the objective of future investigations.

Then, the influence of the tip force direction $\mathbf{d}_f$ on the values of $DCLI$ is investigated. For the scope, $\mathbf{d}_f$ is parameterized by using spherical coordinates as follows:

$$\mathbf{d}_f = \begin{bmatrix} \sin\alpha\cos\beta \\ \sin\alpha\sin\beta \\ \cos\alpha \end{bmatrix} \tag{6.32}$$

where α is the inclination angle, and β is the azimuth. For a given ϕ_0 and given CTR configuration, a two-dimensional grid that discretizes uniformly $\alpha \in [0, 2\pi], \beta \in [0, 2\pi]$ is generated, and $DCLI$ computed for each pair of α, β.

First, let us consider the case of $\phi_0 = 60°$ illustrated in Fig. 6.9a. By changing α, β, the value of $DCLI$ modifies accordingly, and a minimum of $DCLI = 0.149$ N is found at $\alpha = 1.84, \beta = 5.17$ rad, corresponding to $\mathbf{d}_f = [0.47, -0.86, -0.27]$. At the minimum, $DCLI$ and f_{CRIT} are comparable, with 0.214 N and 0.149 N, respectively. Instead, the previously investigated direction $\mathbf{d}_f = [+1, 0, 0] \rightarrow \alpha = \pi/2, \beta = 2\pi$ is far from the minimum, high values of $DCLI$ are displayed, and no f_{CRIT} exists.

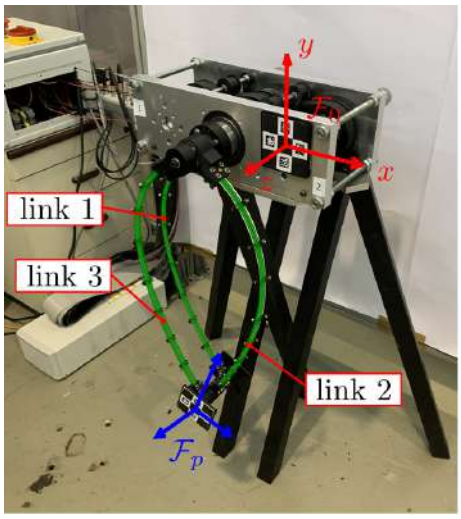


Figure 6.10: The *CPRs* prototype modelling schematics, with the terminology employed in this chapter.

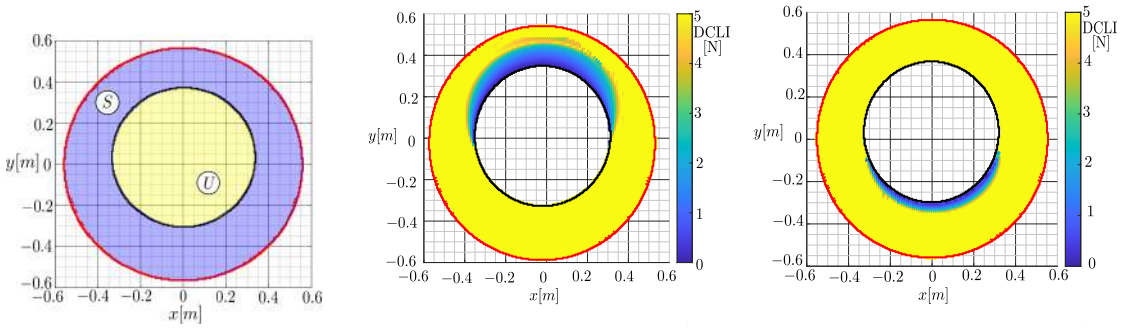
When considering a configuration closer to the instability ($\phi_0 = 160°$, Fig. 6.9b), the direction where *DCLI* is minimum modifies to $\alpha = 1.663$ rad, $\beta = 6.098$ rad and *DCLI* at the minimum is 0.136N. f_{CRIT} is 0.078 N for the same direction. In this case, the previously considered direction $\mathbf{d}_f = +[1, 0, 0]$ is much closer to the minimum, and thus a f_{CRIT} exists in Fig. 6.8a at $\phi_0 = 160°$. It should also be noted that, since polar coordinates are employed, the trend of *DCLI* is periodic, and the same minimum is found twice in Fig. 6.9.

As a summary, *DCLI* effectively measures the distance to instability since i) *DCLI* goes to zero when the instability occurs, and ii) *DCLI* is in accordance with state-of-the-art indices. Moreover, *DCLI* can also be used to characterize the directions of f for which instability easily occurs.

6.3.3 A spatial *CPR* with two controlled *DoFs*

This case study aims to illustrate the application of *DCLI* on a more complex continuum structure, that is, a *CPR*. In particular, the two-controlled *DoFs CPR* that was proposed in Chapter 5 (illustrated in Fig. 6.10 for brevity) is considered as a benchmark. This *CPR* is considered because its workspace and its equilibrium stability limits have been experimentally validated. The geometrico-static modelling of this robot has been discussed in the previous chapter, and in the following, the *EE* mass $m = 0.216$ g is considered as a concentrated *EE* load aligned with the gravitational acceleration $\mathbf{g} = [0, -9.81, 0]$, and the links weight is included as distributed loads. The resulting *EE* force is $f_{EE} = 2.11$ N.

The evaluation of the equilibrium stability of the considered *CPR* is crucial since unstable configurations define the limits of the mobility of the robot, *DCLI* is used to measure the distance from the experimentally validated instability. Let us first consider the prototype workspace (*WS*), computed in Sec. 5.3.2, and illustrated in Fig. 6.11a for clarity. Region Ⓢ is a stable equilibrium region, where the assigned motor values correspond to attainable static robot configurations, while region Ⓤ corresponds to an unstable region. The outer border of the *WS* is associated with a Type-1 singularity [56], which is a limit of the inverse geometrico-static problem



(a) Workspace (b) $\mathbf{d}_f = -[0,1,0]$ (c) $\mathbf{d}_f = +[0,1,0]$

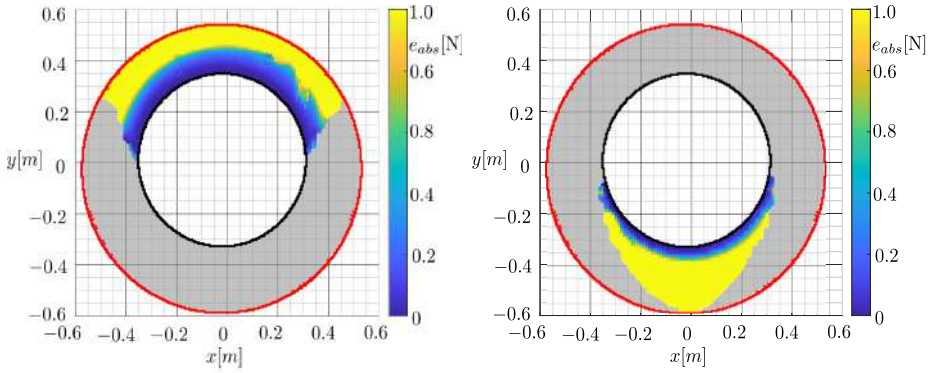
Figure 6.11: The workspace of the *CPR* prototype. Figure (a) highlights the stable region S and the unstable region U. On the same workspace, the trend of *DCLI* is reported in b),c), with $\mathbf{d}_f = [0,-1,0]$ for b), and $\mathbf{d}_f = [0,+1,0]$ for c). Type-1 singularities are depicted in red, and leg singularities in black.

solvability. Instead, the inner border of Ⓢ is associated with a leg singularity [56], and after crossing it, the robot equilibrium becomes unstable. In particular, when the equilibrium becomes unstable, the *EE* pose is not more controllable since an uncontrolled out-of-the-plane motion occurs.

This phenomenon is highly undesirable for practical applications of this prototype, and the goal is to measure the distance from the instability by using *DCLI*. To characterize the robots *WS* in terms of *DCLI*, the influence of a tip external load with direction aligned to the gravity is investigated by considering both positive and negative directions ($\mathbf{d}_f = \pm[0,1,0]$ accordingly to the $\mathcal{F}_0$ of Fig. 6.10). Results are illustrated in Fig. 6.11b, 6.11c, where the values of *DCLI* are displayed. In particular, the attention is directed to values of $DCLI \leqslant 5$ N: as *DCLI* is used to measure the distance to instability, and since $f_{EE} = 2.11$ N, variations of more than 200% of the *EE* load are not of practical interest, since the resulting *CPR* workspace may considerably differ from the one of Fig. 6.11a. Values of *DCLI* greater than 5 N are undisplayed (part in yellow of Fig. 6.11).

First, let us consider $\mathbf{d}_f = [0,-1,0]$, which corresponds to f aligned with gravity. As displayed in Fig. 6.11b, approaching the instability in the upper *WS* region causes *DCLI* values which tend to zero. Instead, the other *WS* regions display values greatly higher than f_{EE}, indicating a larger distance from the instability. On the other hand, when consider $\mathbf{d}_f = [0,+1,0]$, the resulting *DCLI* is illustrated in Fig. 6.11c. As approaching the instability at the lower *WS* region, *DCLI* goes to zero.

Then, the closeness of *DCLI* to f_{CRIT} is quantified. To do this, at each step of the workspace computation *DCLI* and f_{CRIT} and the absolute error $e_{abs} = |f_{CRIT} - CLI|$ are computed. In particular, to compare *DCLI* with f_{CRIT}, the computation of f_{CRIT} is performed as follows. First, let us consider the workspace of Fig. 6.11a obtained with $f_{EE} = 2.11$ N aligned with $-y$. At each stable workspace point, the motor angles $\mathbf{q}_a$ are extracted from the *CPR* configuration and consider these values as fixed. Then, the numerical procedure of Sec. 6.2.4 is used: the *FGSP* is repeatedly solved with desired motor angles $\mathbf{q}_a$ by increasing the tip load until instability is met or the maximum allowed iterations number is reached. Since f_{CRIT} is to be computed at each workspace point, δf is selected as 0.01 N as a trade-off between



(a) $\mathbf{d}_f = -[0, 1, 0]$ (b) $\mathbf{d}_f = +[0, 1, 0]$

Figure 6.12: Comparison between f_{CRIT} and *DCLI*. Figure (a) displays the value of e_{abs} for $\mathbf{d}_f = -[0, 1, 0]$, while Figure (b) for $\mathbf{d}_f = +[0, 1, 0]$. Grey zones are region where no f_{CRIT} exists between $[0, 10]$ N.

accuracy and computational cost, and the maximum number of iterations is set as 500 to explore forces up to 5 N.

The results of this computation are displayed in Fig. 6.12a,6.12b for $\mathbf{d}_f = -[0, 1, 0]$ and $\mathbf{d}_f = +[0, 1, 0]$, respectively. As for the two-tubes *CTR* case, some regions exist where no $f_{CRIT} \in [0, 5]$ N exists. However, it is important to note that when *DCLI* is low, f_{CRIT} is in accordance with *DCLI*, and e_{abs} is reduced. In particular, regions close to the instability where *DCLI* $\leqslant$ 1 N, display reduced absolute errors e_{abs} ($e_{abs} \leqslant$0.1 N). This confirms the capability of *DCLI* to measure the distance to the instability and, close to the instability, estimate the true load to be applied at the *CPR* to cause unstable transitions.

6.4 Conclusions

This chapter proposes a criterion to measure the distance-to-instability of *CRs*. In contrast to state-of-the-art approaches, the *DCLI* does not involve the use of mixed units, and it provides the physical meaning of the results. As *DCLI* represents the additional load to be applied at a defined robot location to cause instability, *DCLI* possesses a well-defined unit (Newton). The applicability of *DCLI* is demonstrated over different case studies, namely the buckling of straight beams, instability of *CTRs*, and the stable-to-unstable transitions of an existing *CPR* prototype. No matter whether the case study is considered, as the instability is approaching, *DCLI* tends toward zero, confirming the correctness of the distance-to-instability measurement. To further verify this, *DCLI* is compared with a state-of-the-art index for *CTRs* and, as *DCLI* goes to zero near instability, also the index of [21] vanishes. Finally, the accuracy of *DCLI* on the critical force estimation is assessed by comparing *DCLI* with the exact force that causes instability, the latter computed by an *ad hoc* numerical approach proposed in this chapter. It is shown that, as the instability is approaching, the distance between *DCLI* and the exact load reduces, and *DCLI* can be used to effectively estimate the critical instability load. However, further tests should be conducted to verify whether the estimation of *DCLI* is safer. This aspect will be the objective of future development.

Chapter 7

Conclusions

Continuum parallel robots (*CPRs*) are manipulators that employ multiple flexible beams arranged in parallel and connected to a rigid platform. *CPRs* promise to solve the main disadvantages of serial *CRs*, such as the reduced payload capacity and accuracy, while keeping great flexibility by design. As *CPRs* are relatively new, several research directions still need to be explored, and this thesis focused on the performance analysis of *CPRs*. In this Chapter, conclusions of this thesis are drawn. Section 7.1 summarizes the contributions of this thesis, while Sec. 7.2 describes open research directions and future development of the contents of this thesis.

7.1 Summary of the Contributions

In this Section, the main scientific contributions of this thesis are resumed and listed about the specific topics.

- **Full Workspace Computation and Numerical Results Certification**. Chapter 3 presented an adaptive flooding algorithm for the workspace computation of *PCPRs*. The algorithm may identify unstable regions and singularity loci, incorporate external loads, and set maximum stress limits and joint bounds. Thanks to an energy-based modelling strategy approximated through finite differences for derivatives, the *IGSP* solution was certified in terms of existence, uniqueness, and convergence of the solution by verifying Kantorovich conditions during the Newton-method-based problem-solving procedure. With this approach, the *IGSP* was solved in a certified way over a large percentage of the workspace in a reduced computational time compared to state-of-the-art algorithms. The main limitation of the proposed approach is the high computational cost where large workspaces and small stepsizes are investigated, and the flooding approach may require the computation of many points.

- **Boundary workspace computation** Chapter 4 proposed an algorithm for the computation of workspace boundaries of *CPRs*. The algorithm, based on a free-space exploration strategy and a boundary reconstruction algorithm, reduced the computational time w.r.t. to actuation or task-space discretization full-workspace computation strategies by identifying only the boundaries of *CPRs*' workspace. Additionally, the algorithm included several constraints, such as singularities, equilibrium stability, and joint and material limits, all

simultaneously during the workspace computation. Compared to state-of-the-art boundary computation approaches, the algorithm provided a useful tool to identify holes in the workspace that may occur in *CPRs*: this is possible thanks to the proposed space exploration strategy. Intending to be a general workspace evaluation tool, the algorithm works with *CPRs* modelling strategies based on general discretization assumptions, which increases the algorithm generality. However, some limitations need to be acknowledged. First, attractive points are effective, but they may fail when their values and numbers are set improperly. As with any heuristic method, parameter tuning is critical for the best performance of the algorithm. Then, even though the algorithm can identify internal workspace boundaries (namely, holes in the workspace) thanks to the proposed exploration strategy, there is no certainty of identifying all of them. However, to the author's knowledge, no state-of-the-art algorithm for the boundary computation of continuum robots demonstrated the ability to identify holes in the workspace.

- **Experimental Equilibrium Stability Assessment** Chapter 5 addressed the experimental validation of equilibrium stability of *CPRs* predictions. The results demonstrated the inability of a model based on planar displacement assumptions to predict the equilibrium stability of a planar *CPR*. A new planar *CPR* was proposed for the scope. The prototype was designed to be nominally planar so no mechanical interference between robot components could occur. Because of the prototype architecture, a material parameter modelling methodology was originally proposed for the specific design of the flexible chains. Finally, a new singularity related to out-of-the-plane uncontrolled motions of the planar *CPR* is experimentally identified for the first time.

- **Distance-to-Instability measurement** Chapter 6 proposed a criterion to measure the distance-to-instability of *CRs*. In contrast to state-of-the-art approaches, the *directional critical load index* (*DCLI*) does not involve the use of mixed units, and it provides the physical meaning of the results. As *DCLI* represents the additional load to be applied at a defined robot location to cause instability, *DCLI* possesses a well-defined unit (Newton). The applicability of *DCLI* was demonstrated over different case studies, namely the buckling of straight beams, instability of *CTRs*, and the stable-to-unstable transitions of an existing *CPR* prototype. No matter the case study considered, as the instability is approaching, *DCLI* tends toward zero, confirming the correctness of the distance-to-instability measurement. To further verify this, *DCLI* was compared with a state-of-the-art index for *CTRs*, and, as *DCLI* goes to zero near instability, state-of-the-art indices vanish. Finally, the accuracy of *DCLI* on the critical force estimation is assessed by comparing *DCLI* with the exact force that causes instability, the latter computed by an *ad hoc* numerical approach proposed chapter 6. It is shown that, as the instability is approaching, the distance between *DCLI* and the exact load reduces, and *DCLI* can be used to estimate the critical instability load.

7.2 Future Perspectives

In this section, future perspectives and possible extensions of the work of this thesis are detailed in the following list:

- **Workspace Computation**. Two chapters of this thesis focused on the workspace computation problem of *CPRs*. Still, there is a great margin for improvement concerning workspace computation algorithms. The development of fast-workspace computation tools is essential to enable optimal designs of continuum robots. However, the proposed approaches require between several minutes (for planar *CPRs*) and hours (for spatial *CPRs*) to compute the workspace boundary once. Thus, using these algorithms for optimal design is impossible, and the application of this algorithm is limited to design explorations only. Developing an algorithm that accurately computes the *CPRs* workspace in times of few seconds would be a great contribution. Concerning the numerical certification of the results during the workspace computation, the strategy proposed in Chapter 3 can be easily extended to spatial cases. However, preliminary tests demonstrated that the results were not as good as in the planar case. Thus, a modelling technique alternative to finite differences may be investigated to certify the results of spatial *CPRs*.

- **Distance-to-instability measurement**. The directional critical load index of Chapter 6 is proposed in this thesis to measure the distance-to-instability. However, measurement is directional, and measuring the overall distance-to-instability, regardless of the force direction to be selected, would be interesting. In this direction, a distance-to-instability ellipsoid may be envisioned to capture the overall stability performances together. At the same time, it is not trivial how to represent the distance-to-instability performance in such a way. Then, it would also be interesting to investigate better how to compute the real critical load that causes instability efficiently. Continuation techniques may offer a possible tool for the scope.

- **Design for large payload capabilities**. The design of a *CPR* with large payload capabilities would be a great future contribution. The tools proposed in this thesis concerning the workspace evaluation and the distance-to-instability measurement may be used to guide the design of a new *CPR*. Although a design only based on flexible components is preferable for safety reasons, some preliminary author's investigation has shown that it is not trivial to build such a prototype, and a hybrid design including both rigid and flexible components may considerably increase the *CPR* payload capability while keeping the flexibility by design.

- **Other Performance aspects**. This thesis focused on two major performance aspects: the workspace computation problem and the equilibrium stability assessment. However, other performance characteristics still need to be explored, such as the accuracy and the force transmission ratio. In particular, it would be interesting to develop a power transmission ratio for *CPRs* to represent how the input power is transferred to the *CPR* end-effector. Since the *CPR* configurations depend on both forces and geometry, the power would be a great performance indicator since it includes geometry and forces simoultaneosly.

Part IV

Appendices

Appendix A

Geometrico and Kinemato Static Models Derivation: Assumed Strain Mode Approach

This appendix derives the geometrico-static model equations and the necessary steps for the derivation of the kinemato-static model equations when using the assumed strain mode approach of [128], for the spatial case in Sec. A.1, and for the planar case in Sec. A.2.

A.1 Spatial Case

This Section derives the geometrico-static and kinemato-static terms when using the assumed strain mode approach for a spatial *CPR* case. Additional details on this derivation are reported in [178] for the interested reader. The first part (Sec. A.1.1) focuses on the geometrico-static model derivation, while the second part (Sec. A.1.2) derives the terms necessary for the kinemato-static model evaluation.

A.1.1 Geometrico-static model

This section derives the geometrico-static model equations of Eq.(2.23) of a *CPR* when the assumed strain mode approach is used, here reported for clarity:

$$\begin{cases} \nabla_{\mathbf{x}}\mathcal{L} = \mathbf{0} \\ \nabla_{\boldsymbol{\lambda}}\mathcal{L} = \mathbf{0} \end{cases} \rightarrow \begin{cases} \nabla_{\mathbf{q}_e} V_{tot} + \nabla_{\mathbf{q}_e}\left(\boldsymbol{\lambda}^T \boldsymbol{\Phi}\right) = \mathbf{0} \\ \nabla_{\mathbf{q}_p} V_{tot} + \nabla_{\mathbf{q}_p}\left(\boldsymbol{\lambda}^T \boldsymbol{\Phi}\right) = \mathbf{0} \\ \boldsymbol{\Phi} = \mathbf{0} \end{cases} \tag{A.1}$$

To derive the expression of these equations, let us consider the deformation energy of the i-th beam of Eq. (2.12) that assumed shear and extensibility to be negligible. The expression is reported here for clarity:

$$V_{ei} = \frac{1}{2}\int_0^{L_i} \left(\mathbf{u}_i(s) - \mathbf{u}_i^*(s)\right)^T \mathbf{K} \left(\mathbf{u}_i(s) - \mathbf{u}_i^*(s)\right) ds \tag{A.2}$$

In the case of an assumed strain mode approach is employed, the i-th beams curvature $\mathbf{u}_i(s) \in \mathbb{R}^3$ is approximated by the use of basis functions as follows:

$$\mathbf{u}_i(s) = \mathbf{N}(s)\mathbf{q}_{ei} \tag{A.3}$$

where $\mathbf{q}_{ei} \in \mathbb{R}^{m_i}$ is the set of discretization coordinates of the i-th beam and the matrix $\mathbf{N}$ collects basis functions. In particular, $\mathbf{N}$ is structured in the following manner:

$$\mathbf{N}(s) = \begin{bmatrix} \mathbf{b}^T(s) & \mathbf{0}_{1\times N_f} & \mathbf{0}_{1\times N_f} \\ \mathbf{0}_{1\times N_f} & \mathbf{b}^T(s) & \mathbf{0}_{1\times N_f} \\ \mathbf{0}_{1\times N_f} & \mathbf{0}_{1\times N_f} & \mathbf{b}^T(s) \end{bmatrix} \in \mathbb{R}^{3\times m_i} \tag{A.4}$$

where $\mathbf{b} \in \mathbb{R}^{N_f\times 1}$ is a base function vector, N_f is the number of base functions employed in $\mathbf{b}$, and $m_i = 3N_f$. Common base function selections involve standard monomials [22]:

$$\mathbf{b}(s) = \begin{bmatrix} 1, & s, & s^2, & s^3, \cdots, & s^{N_f-1} \end{bmatrix} \tag{A.5}$$

However, when using standard monomials, the matrix $\mathbf{J}$ of Eq. (2.24) is frequently ill-conditioned [128], and orthogonal Legendre monomials are selected to alleviate this issue [146]:

$$\mathbf{b}(s) = \begin{bmatrix} 1, & s, & \frac{1}{2}(3s^2-1), & \frac{1}{2}(5s^3-3s), & \cdots, & (2+1/N_f)sb_{N_f-1} - b_{N_f-2} \end{bmatrix} \tag{A.6}$$

where b_j is the j-th component of $\mathbf{b}$. Then, by inserting Eq. (A.3) into Eq. (A.2), and by assuming $\mathbf{u}_i^* = \mathbf{0}$ for simplicity[1], the following expression of V_{ei} is obtained:

$$V_{ei} = \frac{1}{2}\int_0^{L_i} \mathbf{q}_{ei}^T \mathbf{N}^T(s)\mathbf{K}\mathbf{N}(s)\mathbf{q}_{ei} ds \tag{A.7}$$

since $\mathbf{q}_{ei}$ does not depend on s, it is possible to further simplify V_{ei}:

$$V_{ei} = \frac{1}{2}\mathbf{q}_{ei}^T \left(\int_0^{L_i} \mathbf{N}^T(s)\mathbf{K}\mathbf{N}(s) ds \right) \mathbf{q}_{ei} = \frac{1}{2}\mathbf{q}_{ei}^T \mathbf{K}_{ei}\mathbf{q}_{ei} \tag{A.8}$$

where $\mathbf{K}_{ei} \in \mathbb{R}^{m_i\times m_i}$ is a constant matrix obtained by numerically integrating the following expression:

$$\mathbf{K}_{ei} = \int_0^{L_i} \mathbf{N}^T(s)\mathbf{K}_{BT}\mathbf{N}(s) ds \tag{A.9}$$

Once the term V_{ei} is obtained, to compute $\mathcal{L}$, it is necessary to evaluate the influence of distributed loads V_{di} (Eq. (2.13)) and the geometric constraints of the i-th beam $\mathbf{\Phi}_i$. Since $V_{di}, \mathbf{\Phi}_i$ depends on the beam's pose, it is necessary to recover the position and orientation of the i-th beam at each s. For the scope, let us first recover the beam strain $\boldsymbol{\xi}_i$ as follows:

$$\boldsymbol{\xi}_i(s) = \begin{bmatrix} \mathbf{N}(s)\mathbf{q}_{ei} \\ \mathbf{e}_3 \end{bmatrix} \tag{A.10}$$

and the position and orientation of each beam's cross-section are obtained by integrating Eq. (2.3). However, preserving the structure $SE(3)$ of matrix $\mathbf{G}$ during the integration of Eq. (2.3) requires the use of structure-preserving integrators, which are computationally expensive [147]. Instead, by parametrizing the cross-section orientation with unit quaternions $\mathbf{h}_i \in \mathbb{R}^4$, $\mathbf{h}_i = h_{i1} + h_{i2}\mathbf{e}_1 + h_{i3}\mathbf{e}_2 + h_{i4}\mathbf{e}_3$, $\mathbf{e}_1 = [1,0,0], \mathbf{e}_2 = [0,1,0], \mathbf{e}_3 = [0,0,1]$, simplifies Eq. (2.3) as follows:

$$\begin{cases} \mathbf{h}_i'(s) = \frac{1}{2}\mathbf{Q}_i(s)\mathbf{h}_i(s) \\ \mathbf{p}_i'(s) = \mathbf{R}_i(s)\mathbf{e}_3 \end{cases} \tag{A.11}$$

[1]It is possible to use the assumed strain mode approach also for initially-curved beams, such as for concentric tube robots. However, this is out of the scope of this thesis, and the interested reader is addressed to [179]

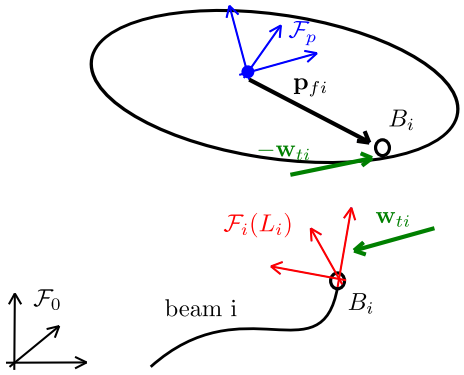


Figure A.1: Virtual opening of the *CPR* closed loop

with $\mathbf{p}_i(0), \mathbf{h}_i(0)$ initial values at the beams' base usually computed from q_{ai}. Eq. (A.11) are named forward equations in the following.

At this stage, the derivation of the geometrico-static model equations starts. Computing $\mathbf{\Phi}_i$ is possible after integrating the forward equations, since it is sufficient to compute the rotation matrix $\mathbf{R}_i(L_)$ from $\mathbf{h}_i(L)$, and insert the values of $\mathbf{p}_i(L), \mathbf{R}_i(L)$ in Eq. (2.18), here reported for clarity:

$$\mathbf{\Phi}_i = \mathbf{C}_i \begin{bmatrix} \left(\mathbf{R}_p^T \mathbf{R}_i(L_i) - \mathbf{R}_i^T(L_i)\mathbf{R}_p\right)^\vee \\ \mathbf{p}_i(L_i) - (\mathbf{p}_p + \mathbf{R}_p \mathbf{p}_{fi}) \end{bmatrix} \tag{A.12}$$

Then, let us derive the first of Eq.(A.1) for the i-th beam by considering separately the contribution of the deformation energy, distributed loads, and constraints:

$$\nabla_{\mathbf{q}_{ei}} \mathcal{L} = \nabla_{\mathbf{q}_{ei}} V_{ei} + \nabla_{\mathbf{q}_{ei}} \left(V_{di} + \boldsymbol{\phi}_i^T \boldsymbol{\lambda}_i\right) \tag{A.13}$$

The term $\nabla_{q_{ei}} V_{ei}$ can be computed by deriving Eq. (A.8) w.r.t. $\mathbf{q}_{ei}$:

$$\nabla_{\mathbf{q}_{ei}} \left(V_{ei}\right) = \mathbf{K}_{ei} \mathbf{q}_{ei} \tag{A.14}$$

Instead, for the derivation of the second term, it is convenient to virtually open the *CPRs* closed loops, as illustrated in Fig. A.1. This representation simplifies the model derivation and implementation. At each passive platform joint, the exchanged wrench between the platform and the beam is named as $\mathbf{w}_{ti}$. In particular, to satisfy the joint connection, $\mathbf{w}_{ti} = \mathbf{C}_i \boldsymbol{\lambda}_i$ [128], where $\mathbf{C}_i$ is illustrated in Tab. 2.2, and $\boldsymbol{\lambda}_i$ is the set of Lagrange multipliers associated with the i-th joint. Let us simplify the notation by introducing $\mathbf{Q}_{ci}$ as

$$\mathbf{Q}_{ci} = \nabla_{\mathbf{q}_{ei}} \left(V_{di} + \boldsymbol{\phi}_i^T \boldsymbol{\lambda}_i\right) \tag{A.15}$$

$\mathbf{Q}_{ci} \in \mathbb{R}^{m_i}$ contains the influence of external loads and the geometrical constraints on the beam equilibrium. To compute $\mathbf{Q}_{ci}$, it is convenient first to evaluate the stress $\mathbf{\Gamma}$ that balances the distributed loads $\mathbf{f}_d$ and the beam's tip wrench $\mathbf{w}_{ti}$. This is done by numerically integrating Eq. (2.7) from $s = L_i$ to $s = 0$, here reported for the reader's convenience:

$$\mathbf{\Gamma}_i'(s) = \mathbf{ad}_{\boldsymbol{\xi}_i}^T(s) \mathbf{\Gamma}_i(s) - \mathbf{w}_d(s) \tag{A.16}$$

with initial values $\mathbf{\Gamma}_i(L) = \mathbf{w}_{ti} = \mathbf{C}_i \boldsymbol{\lambda}_i$[2] Then, according to [22], $\mathbf{Q}_{ci}$ is obtained by integrating the following differential equation from $s = L_i$ to $s = 0$:

$$\mathbf{Q}_{ci}'(s) = \left(\mathbf{BN}(s)\right)^T \mathbf{\Gamma}(s) \tag{A.17}$$

[2]Please note that, in this way, the components of $\mathbf{w}_{ti}$ are in the local beam's tip frame by default.

with $\mathbf{Q}_{ci}(L) = \mathbf{0}$, $\mathbf{B} = [\mathbf{I}_3; \mathbf{0}_3] \in \mathbb{R}^{6\times 3}$. It should be stressed that $\mathbf{w}_d$ is expressed in the local beam frame. For instance, in the case of gravitational loads naturally defined in the global frame, its values must be recovered in the local beam frame at each integration step by considering the cross-section orientation matrix $\mathbf{R}_i$ obtained by the forward integration of $\mathbf{h}_i$. Being $\mathbf{w}_d^f$ the distributed load wrench in the global frame, $\mathbf{w}_d(s)$ is obtained as follows:

$$\mathbf{w}_d(s) = \begin{bmatrix} \mathbf{R}_i(s) & \mathbf{0}_{3\times 3} \\ \mathbf{0}_{3\times 3} & \mathbf{R}_i(s) \end{bmatrix} \mathbf{w}_d^f \tag{A.18}$$

The, let us compute the second of Eq.(A.1) ($\nabla_{\mathbf{q}_p}\mathcal{L}$). To do this, it is convenient to compute the platform equilibrium by considering a generic platform twist variation $\Delta\boldsymbol{\eta} \in \mathbb{R}^6$. Being $\mathbf{w}_p = [\mathbf{0}_{3\times 1}; \mathbf{f}_p]$ the platform wrench in global frame coordinates, and $-\mathbf{w}_{ti}$ the i-th beam's wrench applied to the platform, imposing the platform equilibrium results in:

$$-\mathbf{w}_p + \sum_{i=1}^{n_b} \mathbf{Ad}_{t-i}\mathbf{w}_{ti} = -\mathbf{w}_p + \sum_{i=1}^{n_b} \mathbf{Ad}_{t-i}\mathbf{C}_i\boldsymbol{\lambda}_i = \mathbf{0} \tag{A.19}$$

where $\mathbf{Ad}_{t-i} \in \mathbb{R}^{6\times 6}$ is the matrix that expresses $\mathbf{w}_{ti}$ in global frame coordinates and translates the wrench to the platform origin, which is structured as follows:

$$\mathbf{Ad}_{t-i} = \begin{bmatrix} \mathbf{R}_i(L) & \widehat{\mathbf{p}_{fi}}\mathbf{R}_i(L) \\ \mathbf{0}_{3\times 3} & \mathbf{R}_i(L) \end{bmatrix} \tag{A.20}$$

and $\mathbf{p}_{fi} \in \mathbb{R}^3$ is a vector pointing from the i-th joint position to the platform centre, constant w.r.t. $\mathcal{F}_p$. Moreover, it should be noted that imposing the platform equilibrium is equivalent to calculating $\nabla_{\Delta\boldsymbol{\eta}}\mathcal{L} = \mathbf{0}$ [128]. Thus, to calculate $\nabla_{\mathbf{q}_p}\mathcal{L}$ required for the geometrico-static model solution, the composite function rule derivatives can be used as follows:

$$\nabla_{\mathbf{q}_p}\mathcal{L} = (\nabla_{\Delta\boldsymbol{\eta}}\mathcal{L}) \left(\frac{\partial \Delta\boldsymbol{\eta}}{\partial \mathbf{q}_p}\right) = (\nabla_{\Delta\boldsymbol{\eta}}\mathcal{L})\ \mathbf{M} \tag{A.21}$$

where the matrix $\mathbf{M} \in \mathbb{R}^{6\times 6}$ relates $\mathbf{q}_p \in \mathbb{R}^{n_c}$ with an infinitesimal variation of the platform twist $\Delta\boldsymbol{\eta}$, that is:

$$\Delta\boldsymbol{\eta} = \mathbf{M}\mathbf{q}_p \tag{A.22}$$

Since matrix $\mathbf{M}$ is full-rank by definition, **for the geometrico-static problem solution only**, it is equivalent to impose $\nabla_{\boldsymbol{\eta}}\mathcal{L} = \mathbf{0}$ or $\nabla_{\mathbf{q}_p}\mathcal{L} = \mathbf{0}$. However, when assessing the equilibrium stability, it is important to use $\nabla_{\mathbf{q}_p}\mathcal{L}$ to obtain correct equilibrium stability predictions.

The final expression of the geometrico-static model equations of Eq. (2.23) is obtained by collecting $\nabla_{\mathbf{q}_{ei}}\mathcal{L}, \nabla_{\mathbf{q}_p}\mathcal{L}, \nabla_{\boldsymbol{\lambda}}\mathcal{L}$ together:

$$\begin{cases} \nabla_{\mathbf{q}_{ei}}\mathcal{L} = \mathbf{K}_{ei}\mathbf{q}_{ei} + \mathbf{Q}_{ci} = \mathbf{0}; \quad \forall i = 1, \cdots, n_b \\ \nabla_{\boldsymbol{\lambda}_i}\mathcal{L} = \boldsymbol{\Phi}_i = \mathbf{0}; \quad \forall i = 1, \cdots, n_b \\ \nabla_{\Delta\boldsymbol{\eta}}\mathcal{L} = -\mathbf{w}_p + \sum_{i=1}^{n_b} \mathbf{Ad}_{t-i}\mathbf{C}_i\boldsymbol{\lambda}_i = \mathbf{0} \end{cases} \tag{A.23}$$

A.1.2 Kinemato-static model

The derivation of the kinemato-static model requires the computation of the first derivative of each of Eq. (A.1) w.r.t. $\mathbf{y} = [\mathbf{q}_a, \mathbf{q}_e, \mathbf{q}_p, \boldsymbol{\lambda}]$. In particular, the derivative of Eq.(A.1) w.r.t. $\mathbf{y}$ is structured as follows:

$$\begin{bmatrix} \nabla_{\mathbf{q}_a}(\nabla_{\mathbf{q}_e}\mathcal{L}) & \nabla_{\mathbf{q}_e}(\nabla_{\mathbf{q}_e}\mathcal{L}) & \nabla_{\mathbf{q}_e}(\nabla_{\mathbf{q}_e}\mathcal{L}) & \nabla_{\boldsymbol{\lambda}}(\nabla_{\mathbf{q}_e}\mathcal{L}) \\ \nabla_{\mathbf{q}_a}(\nabla_{\mathbf{q}_p}\mathcal{L}) & \nabla_{\mathbf{q}_e}(\nabla_{\mathbf{q}_p}\mathcal{L}) & \nabla_{\mathbf{q}_e}(\nabla_{\mathbf{q}_p}\mathcal{L}) & \nabla_{\boldsymbol{\lambda}}(\nabla_{\mathbf{q}_p}\mathcal{L}) \\ \nabla_{\mathbf{q}_a}\boldsymbol{\Phi} & \nabla_{\mathbf{q}_e}\boldsymbol{\Phi} & \nabla_{\mathbf{q}_e}\boldsymbol{\Phi} & \nabla_{\boldsymbol{\lambda}}\boldsymbol{\Phi} \end{bmatrix} \tag{A.24}$$

- The first of Eq.(A.1) $\mathbf{q}_{ei}, q_{ai}, \boldsymbol{\lambda}_i$ and $\nabla_{\mathbf{q}_p}\left(\nabla_{\mathbf{q}_{ei}}\mathcal{L}\right) = \mathbf{0}$.
- The second of Eq.(A.1) depends on all the variables in $\mathbf{y}$.
- The last of Eq.(A.1) depends on $q_{ai}, \mathbf{q}_e i$ and $\mathbf{q}_p$, and thus $\nabla_{\boldsymbol{\lambda}_i} = \mathbf{0}$.

In the following, the derivatives of $\mathbf{p}_i, \mathbf{h}_i, \boldsymbol{\Gamma}_i, \mathbf{Q}_{ci}$ are calculated since are the necessary terms for the derivation of the kinemato-static model terms, while the expression of the terms in Eq.(A.24) are not reported here for brevity sake. A derivative propagation approach is used in the following, as done in [58]. For instance, let us consider the first of Eq. (A.11), and the goal is to compute $\partial\mathbf{h}_i/\partial\mathbf{q}_{ei}$. Since derivatives w.r.t. s and $\mathbf{q}_{ei}$ are commutative, and $\mathbf{h}_i'$ is governed by Eq. (A.11), the following expression governing the evolution of $\partial\mathbf{h}_i/\partial\mathbf{q}_{ei}$ over s is obtained:

$$\left(\frac{\partial\mathbf{h}_i}{\partial\mathbf{q}_{ei}}\right)' = \frac{\partial}{\partial\mathbf{q}_{ei}}\left(\mathbf{h}_i'\right) = \frac{\partial}{\partial\mathbf{q}_{ei}}\left(\frac{1}{2}\mathbf{Q}_i\mathbf{h}_i\right) = \frac{1}{2}\left(\mathbf{D}_{ki}\mathbf{N} + \mathbf{Q}_i\frac{\partial\mathbf{h}_i}{\partial\mathbf{q}_{ei}}\right) \tag{A.25}$$

where:

$$\mathbf{D}_{ki} = \begin{bmatrix} -h_{i2} & -h_{i3} & -h_{i4} \\ +h_{i1} & -h_{i4} & +h_{i3} \\ +h_{i4} & +h_{i1} & -h_{i2} \\ -h_{i3} & +h_{i2} & +h_{i1} \end{bmatrix} \tag{A.26}$$

Eq. (A.25) is integrated from $s = 0$ to $s = L$ to get $\partial\mathbf{h}_i/\partial\mathbf{q}_{ei}(L)$, with initial value $\partial\mathbf{h}_i/\partial\mathbf{q}_{ei}(0) = \mathbf{0}$. Proceeding in a similar fashion, and considering for simplicity only actuators placed at the robot's base, the following differential equations that govern the derivatives of $\mathbf{p}_i, \mathbf{h}_i$ can be obtained:

$$\left(\frac{\partial\mathbf{p}_i}{\partial\mathbf{q}_{ei}}\right)' = \mathbf{D}_{hi}\frac{\partial\mathbf{h}}{\partial\mathbf{q}_{ei}} \tag{A.27}$$

$$\left(\frac{\partial\mathbf{h}_i}{\partial\mathbf{q}_{ei}}\right)' = \frac{1}{2}\left(\mathbf{D}_{ki}\mathbf{N} + \mathbf{Q}_i\frac{\partial\mathbf{h}_i}{\partial\mathbf{q}_{ei}}\right) \tag{A.28}$$

$$\left(\frac{\partial\mathbf{p}_i}{\partial q_{ai}}\right)' = \mathbf{D}_{hi}\frac{\partial\mathbf{h}_i}{\partial q_{ai}} \tag{A.29}$$

$$\left(\frac{\partial\mathbf{h}_i}{\partial q_{ai}}\right)' = \frac{1}{2}\mathbf{A}_i\frac{\partial\mathbf{h}_i}{\partial q_{ai}} \tag{A.30}$$

with

$$\mathbf{D}_{hi} = 2\begin{bmatrix} +h_{i3} & +h_{i4} & +h_{i1} & +h_{i2} \\ -h_{i2} & -h_{i1} & +h_{i4} & +h_{i3} \\ +h_{i1} & -h_{i2} & -h_{i3} & +h_{i4} \end{bmatrix} \tag{A.31}$$

Eq. (A.30) is integrated from $s = 0$ to $s = L$ with initial values $\partial\mathbf{p}_i/\partial\mathbf{q}_{ei}(0) = \partial\mathbf{h}_i/\partial\mathbf{q}_{ei}(0) = \mathbf{0}$. Instead, $\partial\mathbf{p}_i/\partial q_{ai}(0)$, $\partial\mathbf{h}_i/\partial q_{ai}(0)$ can be obtained by the knowledge of the specific actuator to be employed. The derivative of $\mathbf{p}_i, \mathbf{h}_i$ w.r.t. $\boldsymbol{\lambda}_i$ are null since $\boldsymbol{\lambda}_i$ does not influence the integration of the forward equations.

Proceeding in the same fashion, the following set of differential equations governing the evolution of the derivatives of $\boldsymbol{\Lambda}_i, \mathbf{Q}_{ci}$ can be obtained by using a derivative propagation

approach:

$$\left(\frac{\partial \mathbf{\Gamma}_i}{\partial \mathbf{q}_{ei}}\right)' = \frac{\partial\left(\mathbf{ad}_{\boldsymbol{\xi}_i}^T \mathbf{\Gamma}_i\right)}{\partial \mathbf{q}_{ei}} - \frac{\partial \mathbf{w}_d}{\partial \mathbf{q}_{ei}} \tag{A.32}$$

$$\left(\frac{\partial \mathbf{Q}_{ci}}{\partial \mathbf{q}_{ei}}\right)' = (\mathbf{BN})^T \frac{\partial \mathbf{\Gamma}_i}{\partial \mathbf{q}_{ei}} \tag{A.33}$$

$$\left(\frac{\partial \mathbf{\Gamma}_i}{\partial q_{ai}}\right)' = \mathbf{ad}_{\boldsymbol{\xi}_i}^T \frac{\partial \mathbf{\Gamma}_i}{\partial q_{ai}} - \frac{\partial \mathbf{w}_d}{\partial q_{ai}} \tag{A.34}$$

$$\left(\frac{\partial \mathbf{Q}_{ci}}{\partial q_{ai}}\right)' = (\mathbf{BN})^T \frac{\partial \mathbf{\Gamma}_i}{\partial q_{ai}} \tag{A.35}$$

with the equations integrated from $s = L$ to $s = 0$ with all null initial values. The term $\partial\left(\mathbf{ad}_{\boldsymbol{\xi}}\mathbf{\Gamma}\right)/\partial \mathbf{q}_e$ can be obtained by first computing the product $\mathbf{ad}_{\boldsymbol{\xi}_i}^T \mathbf{\Gamma}_i$ component wise, and then calculating the derivative w.r.t. $\mathbf{q}_{ei}$, resulting in the following final expression:

$$\frac{\partial\left(\mathbf{ad}_{\boldsymbol{\xi}_i}^T \mathbf{\Gamma}_i\right)}{\partial \mathbf{q}_{ei}} = \begin{bmatrix} \widehat{\mathbf{\Gamma}_{1:3-i}}^T \\ \widehat{\mathbf{\Gamma}_{4:6-i}}^T \end{bmatrix} \mathbf{N} + \mathbf{ad}_{\boldsymbol{\xi}_i}^T \frac{\partial \mathbf{\Gamma}_i}{\partial \mathbf{q}_{ei}} \tag{A.36}$$

and the terms $\partial \mathbf{w}_d/\partial \mathbf{q}_e, \partial \mathbf{w}_d/\partial q_{ai}$ can be computed by the knowledge of $\partial \mathbf{h}/\partial \mathbf{q}_e$,$\partial \mathbf{h}/\partial q_{ai}$ from the forward integration. Finally, the last necessary term is $\partial \mathbf{Q}_{ci}/\partial \boldsymbol{\lambda}_i$ obtained by first considering the following composite functions rule derivative:

$$\frac{\partial \mathbf{Q}_{ci}}{\partial \boldsymbol{\lambda}_i} = \frac{\partial \mathbf{Q}_{ci}}{\partial \mathbf{w}_{ti}} \frac{\partial \mathbf{w}_{ti}}{\partial \boldsymbol{\lambda}_i} = \frac{\partial \mathbf{Q}_{ci}}{\partial \mathbf{w}_{ti}} \mathbf{C}_i \tag{A.37}$$

and the term $\partial \mathbf{Q}_{ci}/\partial \mathbf{w}_{ti}$ obtained by a derivative propagation approach:

$$\left(\frac{\partial \mathbf{\Gamma}_i}{\partial \mathbf{w}_{ti}}\right)' = \mathbf{ad}_{\boldsymbol{\xi}_i} \frac{\partial \mathbf{\Gamma}_i}{\partial \mathbf{w}_{ti}} \tag{A.38}$$

$$\left(\frac{\partial \mathbf{Q}_{ci}}{\partial \mathbf{w}_{ti}}\right)' = (\mathbf{BN})^T \frac{\partial \mathbf{\Gamma}_i}{\partial \mathbf{w}_{ti}} \tag{A.39}$$

with initial values $\partial \mathbf{\Gamma}_i/\partial \mathbf{w}_{ti}(L) = \mathbf{I}_6$, and $\partial \mathbf{Q}_{ci}/\partial \mathbf{w}_{ti}(L) = \mathbf{0}$.

A.2 Planar Case

This Section derives the geometrico-static and kinemato-static terms when using the assumed strain mode approach for a planar *CPR* case. Additional details on this derivation are reported in [178] for the interested reader. The first part (Sec. A.2.1) focuses on the geometrico-static model derivation, while the second part (Sec.A.2.2) derives the terms necessary for the kinemato-static model evaluation.

A.2.1 Geometrico-static model

This section derives the geometrico-static model equations of Eq.(2.23) of a *CPR* when the assumed strain mode approach is used in a planar case, here reported for clarity:

$$\begin{cases} \nabla_{\mathbf{x}} \mathcal{L} = \mathbf{0} \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L} = \mathbf{0} \end{cases} \rightarrow \begin{cases} \nabla_{\mathbf{q}_e} V_{tot} + \nabla_{\mathbf{q}_e}\left(\boldsymbol{\lambda}^T \mathbf{\Phi}\right) = \mathbf{0} \\ \nabla_{\mathbf{q}_p} V_{tot} + \nabla_{\mathbf{q}_p}\left(\boldsymbol{\lambda}^T \mathbf{\Phi}\right) = \mathbf{0} \\ \mathbf{\Phi} = \mathbf{0} \end{cases} \tag{A.40}$$

To derive the expression of these equations, let us consider the deformation energy of the i-th beam of Eq. (2.12) when i) shear and extensibility are neglected, and ii) in a planar displacement case, here reported for clarity:

$$V_{ei} = \frac{1}{2}\int_0^{L_i} K_b \left(u_i(s) - u_i^*(s)\right)^T \left(u_i(s) - u_i^*(s)\right) ds \tag{A.41}$$

with K_b being the material stiffness (usually equal to EI). In the case of an assumed strain mode approach is employed, th i-th beams curvature $u_i(s) \in \mathbb{R}$ is approximated by the use of basis functions as follows:

$$u_i(s) = \mathbf{N}(s)\mathbf{q}_{ei} \tag{A.42}$$

where $\mathbf{q}_{ei} \in \mathbb{R}^{m_i}$ is the set of discretization coordinates of the i-th beam and the vector $\mathbf{N} = \mathbf{b}^T \in \mathbb{R}^{1\times N_f}$ collects basis functions, N_f is the number of base functions employed in $\mathbf{b}$, and $m_i = N_f$. The definition of $\mathbf{b}$ is the same for the spatial case of the previous section. Then, by inserting Eq. (A.42) into Eq. (A.41), and by assuming $u_i^* = 0$ for simplicity, the following expression of V_{ei} is obtained:

$$V_{ei} = \frac{1}{2}K_b \int_0^{L_i} \mathbf{q}_{ei}^T \mathbf{N}^T(s)\mathbf{N}(s)\mathbf{q}_{ei} ds \tag{A.43}$$

since $\mathbf{q}_{ei}$ does not depend on s, it is possible to further simplify V_{ei}:

$$V_{ei} = \frac{1}{2}\mathbf{q}_{ei}^T \left(\int_0^{L_i} K_b \mathbf{N}^T(s)\mathbf{N}(s) ds\right) \mathbf{q}_{ei} = \frac{1}{2}\mathbf{q}_{ei}^T \mathbf{K}_{ei}\mathbf{q}_{ei} \tag{A.44}$$

where $\mathbf{K}_{ei} \in \mathbb{R}^{m_i \times m_i}$ is a constant matrix obtained by numerically integrating the following expression:

$$\mathbf{K}_{ei} = \int_0^{L_i} K_b \mathbf{N}^T(s)\mathbf{N}(s) ds \tag{A.45}$$

Once the term V_{ei} is obtained, to compute $\mathcal{L}$, it is necessary to evaluate the influence of distributed loads V_{di} (Eq. (2.13)) and the geometric constraints of the i-th beam $\mathbf{\Phi}_i$. Since $V_{di}, \mathbf{\Phi}_i$ depends on the beam's pose, it is necessary to recover the position and orientation of the i-th beam at each s. For the scope, let us first recover the beam strain $\boldsymbol{\xi}_i$ as follows:

$$\boldsymbol{\xi}_i(s) = \begin{bmatrix} u_i \\ \mathbf{e}_2^p \end{bmatrix} \tag{A.46}$$

with $u_i = \mathbf{N}(s)\mathbf{q}_{ei}$, $\mathbf{e}_2^p = [0; 1]$, and the tangent to the beam's centerline is assumed to be aligned to the local y axis. Since the beam is planar, the pose of each cross-section is described by a position vector $\mathbf{p}_i(s) \in \mathbb{R}^2$, and an orientation angle $\theta(s) \in \mathbb{R}$. To recover position and orientation of the beam, the following differential equations are integrated

$$\begin{cases} \theta_i(s)' = u_i = \mathbf{N}(s)\mathbf{q}_{ei} \\ \mathbf{p}_i(s)' = \mathbf{R}_i(s)\mathbf{e}_2^p \end{cases} \tag{A.47}$$

with $\mathbf{p}_i(0), \theta_i(0)$ initial values at the beams' base usually computed from q_{ai}. Eq. (A.47) are named forward equations in the following.

At this stage, the derivation of the geometrico-static model equations starts. Computing $\mathbf{\Phi}$ is possible after the forward equations integration, since it is sufficient to insert the values of $\mathbf{p}_i(L), \theta_i(L)$ in Eq. (2.18), simplified for the planar case and reported here for brevity:

$$\mathbf{\Phi}_i = \mathbf{C}_i \begin{bmatrix} \theta_i(L) - \theta_p \\ \mathbf{p}_i(L) - (\mathbf{p}_p + \mathbf{R}_p \mathbf{p}_{fi}) \end{bmatrix} \tag{A.48}$$

with θ_p the platform orientation angle, and $\mathbf{C}_i \in \mathbb{R}^{n_{\phi_i} \times 3}$ the joint matrix. Let us derive $\nabla_{\mathbf{q}_{ei}} \mathcal{L}$, by considering separately the contribution of the deformation energy and the remaining terms:

$$\nabla_{\mathbf{q}_{ei}} \mathcal{L} = \nabla_{\mathbf{q}_{ei}} V_{ei} + \nabla_{\mathbf{q}_{ei}} \left(V_{di} + \boldsymbol{\phi}_i^T \boldsymbol{\lambda}_i \right) \tag{A.49}$$

The term $\nabla_{\mathbf{q}_{ei}} V_{ei}$ can be computed by deriving Eq. (A.44) w.r.t. $\mathbf{q}_{ei}$:

$$\nabla_{\mathbf{q}_{ei}} \left(V_{ei} \right) = \mathbf{K}_{ei} \mathbf{q}_{ei} \tag{A.50}$$

Instead, for the derivation of the second term, it is convenient to virtually open the *CPRs* closed loops, as illustrated in Fig. A.1. This representation simplifies the model derivation and implementation. At each passive platform joint, the exchanged wrench between the platform and the beam is named as $\mathbf{w}_{ti} \in \mathbb{R}^3$. As in the previous case, to satisfy the joint connection, $\mathbf{w}_{ti} = \mathbf{C}_i \boldsymbol{\lambda}_i \in \mathbb{R}^3$ [128]. The vector $\mathbf{Q}_{ci}$ is introduced to simplify the notation:

$$\mathbf{Q}_{ci} = \nabla_{\mathbf{q}_{ei}} \left(V_{di} + \boldsymbol{\Phi}_i^T \boldsymbol{\lambda}_i \right) \tag{A.51}$$

$\mathbf{Q}_{ci} \in \mathbb{R}^{m_i}$ contains the influence of external loads and the geometrical constraints on the beam equilibrium. To compute $\mathbf{Q}_{ci}$, it is convenient first to evaluate the internal beam stress $\boldsymbol{\Gamma} \in \mathbb{R}^3$ that balances the distributed loads $\mathbf{f}_d$ and the beam's tip wrench $\mathbf{w}_{ti} \in \mathbb{R}^3$. This is done by numerically integrating Eq. (2.7) from $s = L_i$ to $s = 0$, here reported for the reader's convenience:

$$\boldsymbol{\Gamma}_i'(s) = \mathbf{ad}_{\boldsymbol{\xi}_i}^T(s) \boldsymbol{\Gamma}_i(s) - \mathbf{w}_d(s) \tag{A.52}$$

with initial values $\boldsymbol{\Gamma}_i(L) = \mathbf{w}_{ti} = \mathbf{C}_i \boldsymbol{\lambda}_i$[3]. In particular, in the planar case, $\mathbf{ad}_{\boldsymbol{\xi}_i}$ simplifies as follows:

$$\mathbf{ad}_{\boldsymbol{\xi}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & u_i & 0 \\ 1 & 0 & u_i \end{bmatrix} \tag{A.53}$$

Then, according to [22], $\mathbf{Q}_{ci}$ is obtained by integrating the following differential equation from $s = L_i$ to $s = 0$:

$$\mathbf{Q}_{ci}'(s) = \left(\mathbf{B}\mathbf{N}(s) \right)^T \boldsymbol{\Gamma}(s) \tag{A.54}$$

with $\mathbf{Q}_{ci}(L) = \mathbf{0}$, $\mathbf{B} = [1, 0, 0] \in \mathbb{R}^{3 \times 1}$. It should be stressed that $\mathbf{w}_d$ is expressed in the local beam frame. For instance, in the case of gravitational loads naturally defined in the global frame, its values must be recovered in the local beam frame at each integration step by considering the cross-section orientations $\mathbf{h}_i$ obtained by the forward integration. Being $\mathbf{w}_d^f$ the distributed load wrench in the global frame, $\mathbf{w}_d(s)$ is obtained as follows:

$$\mathbf{w}_d(s) = \begin{bmatrix} 1 & \mathbf{0}_{1\times 2} \\ \mathbf{0}_{2\times 1} & \mathbf{R}_i(s) \end{bmatrix} \mathbf{w}_d^f \tag{A.55}$$

The, let us compute $\nabla_{\mathbf{q}_p} \mathcal{L}$. To do this, it is convenient to compute the platform equilibrium by considering a generic platform twist variation $\Delta\boldsymbol{\eta} \in \mathbb{R}^3$. Being $\mathbf{w}_p$ the platform wrench in global frame coordinates, and $-\mathbf{w}_{ti}$ the i-th joint wrench applied to the platform, imposing the platform equilibrium results in:

$$-\mathbf{w}_p + \sum_{i=1}^{n_b} \mathbf{Ad}_{t-i} \mathbf{w}_{ti} = -\mathbf{w}_p + \sum_{i=1}^{n_b} \mathbf{Ad}_{t-i} \mathbf{C}_i \boldsymbol{\lambda}_i = \mathbf{0} \tag{A.56}$$

where $\hat{p}_{fi} = [-p_{fiy}, +p_{fix}]$, $\mathbf{Ad}_{t-i} \in \mathbb{R}^{3\times 3}$ is the matrix that converts $\mathbf{w}_{ti}$ in the global frame and translates the wrench to the platform origin, structured as follows:

$$\mathbf{Ad}_{t-i} = \begin{bmatrix} 1 & \widehat{\mathbf{p}_{fi}} \mathbf{R}_i(L) \\ \mathbf{0}_{2\times 1} & \mathbf{R}_i(L) \end{bmatrix} \tag{A.57}$$

[3]Please note that, in this way, the components of $\mathbf{w}_{ti}$ are in the local beam's tip frame by default.

and $\mathbf{p}_{fi} \in \mathbb{R}^2$ is a vector pointing from the i-th joint position to the platform centre, constant w.r.t. $\mathcal{F}_p$. Moreover, it should be noted that imposing the platform equilibrium is equivalent to calculating $\nabla_{\mathbf{q}_p}\mathcal{L} = \mathbf{0}$ [128], in the planar case.

The final expression of the geometrico-static model equations of Eq. (2.23) is obtained by collecting $\nabla_{\mathbf{q}_{ei}}\mathcal{L}, \nabla_{\mathbf{q}_p}\mathcal{L}, \nabla_{\boldsymbol{\lambda}}\mathcal{L}$ together:

$$\begin{cases} \nabla_{\mathbf{q}_{ei}}\mathcal{L} = \mathbf{K}_{ei}\mathbf{q}_{ei} + \mathbf{Q}_{ci} = \mathbf{0}; \quad \forall i = 1, \cdots, n_b \\ \nabla_{\boldsymbol{\lambda}_i}\mathcal{L} = \boldsymbol{\Phi}_i = \mathbf{0}; \quad \forall i = 1, \cdots, n_b \\ \nabla_{\Delta\boldsymbol{\eta}}\mathcal{L} = -\mathbf{w}_p + \sum_{i=1}^{n_b} \mathbf{Ad}_{t-i}\mathbf{C}_i\boldsymbol{\lambda}_i = \mathbf{0} \end{cases} \tag{A.58}$$

A.2.2 Kinemato-static model

The derivation of the kinemato-static model requires the computation of the first derivative of each of Eq. (A.40) w.r.t. $\mathbf{y} = [\mathbf{q}_a, \mathbf{q}_e, \mathbf{q}_p, \boldsymbol{\lambda}]$. In particular, the derivative of Eq.(A.40) w.r.t. $\mathbf{y}$ is structured as follows:

$$\begin{bmatrix} \nabla_{\mathbf{q}_a}\left(\nabla_{\mathbf{q}_e}\mathcal{L}\right) & \nabla_{\mathbf{q}_e}\left(\nabla_{\mathbf{q}_e}\mathcal{L}\right) & \nabla_{\mathbf{q}_e}\left(\nabla_{\mathbf{q}_e}\mathcal{L}\right) & \nabla_{\boldsymbol{\lambda}}\left(\nabla_{\mathbf{q}_e}\mathcal{L}\right) \\ \nabla_{\mathbf{q}_a}\left(\nabla_{\mathbf{q}_p}\mathcal{L}\right) & \nabla_{\mathbf{q}_e}\left(\nabla_{\mathbf{q}_p}\mathcal{L}\right) & \nabla_{\mathbf{q}_e}\left(\nabla_{\mathbf{q}_p}\mathcal{L}\right) & \nabla_{\boldsymbol{\lambda}}\left(\nabla_{\mathbf{q}_p}\mathcal{L}\right) \\ \nabla_{\mathbf{q}_a}\boldsymbol{\Phi} & \nabla_{\mathbf{q}_e}\boldsymbol{\Phi} & \nabla_{\mathbf{q}_e}\boldsymbol{\Phi} & \nabla_{\boldsymbol{\lambda}}\boldsymbol{\Phi} \end{bmatrix} \tag{A.59}$$

- The first of Eq.(A.40) $\mathbf{q}_{ei}, q_{ai}, \boldsymbol{\lambda}_i$ and $\nabla_{\mathbf{q}_p}\left(\nabla_{\mathbf{q}_{ei}}\mathcal{L}\right) = \mathbf{0}$.
- The second of Eq.(A.40) depends on all the variables in $\mathbf{y}$.
- The last of Eq.(A.40) depends on $q_{ai}, \mathbf{q}_e i$ and $\mathbf{q}_p$, and thus $\nabla_{\boldsymbol{\lambda}_i} = \mathbf{0}$.

In the following, the derivatives of $\mathbf{p}_i, \theta_i, \boldsymbol{\Gamma}_i, \mathbf{Q}_{ci}$ are calculated since are the necessary terms for the derivation of the kinemato-static model terms, while the expression of the terms in Eq.(A.59) are not reported here for brevity sake. These derivatives are calculated by using a derivative propagation approach as done in [58]. For instance, let us consider the first of Eq. (A.47), and the goal is to compute $\partial\mathbf{h}_i/\partial\mathbf{q}_{ei}$. Since derivatives w.r.t. s and $\mathbf{q}_{ei}$ are commutative, and θ_i' is governed by Eq. (A.47), the following expression governing the evolution of $\partial\theta_i/\partial\mathbf{q}_{ei}$ over s is obtained:

$$\left(\frac{\partial\theta_i}{\partial\mathbf{q}_{ei}}\right)' = \frac{\partial}{\partial\mathbf{q}_{ei}}\left(\theta_i'\right) = \frac{\partial}{\partial\mathbf{q}_{ei}}\left(\mathbf{N}\mathbf{q}_{ei}\right) = \mathbf{N} \tag{A.60}$$

Eq. (A.60) is integrated from $s = 0$ to $s = L$ to get $\partial\theta_i/\partial\mathbf{q}_{ei}$, with initial value $\partial\theta_i/\partial\mathbf{q}_{ei}(0) = \mathbf{0}$. Proceeding similarly, and considering for simplicity only actuators placed at the robot's base, the following differential equations that govern the derivatives of $\mathbf{p}_i, \theta_i$ can be obtained:

$$\left(\frac{\partial\mathbf{p}_i}{\partial\mathbf{q}_{ei}}\right)' = \mathbf{R}_{\pi/2}\mathbf{e}_2^p\frac{\partial\theta_i}{\partial\mathbf{q}_{ei}} \tag{A.61}$$

$$\left(\frac{\partial\theta_i}{\partial\mathbf{q}_{ei}}\right)' = \mathbf{N} \tag{A.62}$$

$$\left(\frac{\partial\mathbf{p}_i}{\partial q_{ai}}\right)' = \mathbf{R}_{\pi/2}\mathbf{e}_2^p\frac{\partial\theta_i}{\partial q_{ai}} \tag{A.63}$$

$$\left(\frac{\partial\theta_i}{\partial\mathbf{q}_{ei}}\right)' = \mathbf{0}_{1\times m_i} \tag{A.64}$$

with $\mathbf{R}_{\pi/2} = \mathbf{R}_z(\theta_i + \pi/2)$. Eq. (A.64) is integrated from $s = 0$ to $s = L$ with initial values $\partial\mathbf{p}_i/\partial\mathbf{q}_{ei}(0) = \partial\theta_i/\partial\mathbf{q}_{ei}(0) = \mathbf{0}$, and $\partial\mathbf{p}_i/\partial q_{ai}(0)$, $\partial\theta_i/\partial q_{ai}(0)$ can be obtained by

the knowledge of the specific actuator to be employed. Moreover, it should be noted that the derivative of $\mathbf{p}_i, \theta_i$ w.r.t. $\boldsymbol{\lambda}_i$ is null since $\boldsymbol{\lambda}_i$ does not influence the integration of the forward equations.

Proceeding in the same fashion, the following set of differential equations governing the evolution of the derivatives of $\boldsymbol{\Lambda}_i, \mathbf{Q}_{ci}$ can be obtained by using a derivative propagation approach:

$$\left(\frac{\partial \boldsymbol{\Gamma}_i}{\partial \mathbf{q}_{ei}}\right)' = \frac{\partial\left(\mathbf{ad}^T_{\boldsymbol{\xi}_i}\boldsymbol{\Gamma}_i\right)}{\partial \mathbf{q}_{ei}} - \frac{\partial \mathbf{w}_d}{\partial \mathbf{q}_{ei}} \tag{A.65}$$

$$\left(\frac{\partial \mathbf{Q}_{ci}}{\partial \mathbf{q}_{ei}}\right)' = (\mathbf{BN})^T \frac{\partial \boldsymbol{\Gamma}_i}{\partial \mathbf{q}_{ei}} \tag{A.66}$$

$$\left(\frac{\partial \boldsymbol{\Gamma}_i}{\partial q_{ai}}\right)' = \mathbf{ad}^T_{\boldsymbol{\xi}_i}\frac{\partial \boldsymbol{\Gamma}_i}{\partial q_{ai}} - \frac{\partial \mathbf{w}_d}{\partial q_{ai}} \tag{A.67}$$

$$\left(\frac{\partial \mathbf{Q}_{ci}}{\partial q_{ai}}\right)' = (\mathbf{BN})^T \frac{\partial \boldsymbol{\Gamma}_i}{\partial q_{ai}} \tag{A.68}$$

with all the equations integrated from $s = L$ to $s = 0$ with null initial values, and the term $\partial\left(\mathbf{ad}_{\boldsymbol{\xi}}\boldsymbol{\Gamma}\right)/\partial \mathbf{q}_e$ can be obtained as:

$$\frac{\partial\left(\mathbf{ad}^T_{\boldsymbol{\xi}_i}\boldsymbol{\Gamma}_i\right)}{\partial \mathbf{q}_{ei}} = \begin{bmatrix} \mathbf{0}_{1\times N_f} \\ \mathbf{N} \\ \mathbf{N} \end{bmatrix} \boldsymbol{\Gamma} + \mathbf{ad}^T_{\boldsymbol{\xi}_i}\frac{\partial \boldsymbol{\Gamma}_i}{\partial \mathbf{q}_{ei}} \tag{A.69}$$

and the terms $\partial \mathbf{w}_d/\partial \mathbf{q}_e, \partial \mathbf{w}_d/\partial q_{ai}$ can be computed by the knowledge of $\partial\theta/\partial \mathbf{q}_e$,$\partial\theta/\partial q_{ai}$ from the forward integration. Finally, the last necessary term is $\partial \mathbf{Q}_{ci}/\partial \boldsymbol{\lambda}_i$ obtained by first considering the following composite functions rule derivative:

$$\frac{\partial \mathbf{Q}_{ci}}{\partial \boldsymbol{\lambda}_i} = \frac{\partial \mathbf{Q}_{ci}}{\partial \mathbf{w}_{ti}}\frac{\partial \mathbf{w}_{ti}}{\partial \boldsymbol{\lambda}_i} = \frac{\partial \mathbf{Q}_{ci}}{\partial \mathbf{w}_{ti}}\mathbf{C}_i \tag{A.70}$$

and the term $\partial \mathbf{Q}_{ci}/\partial \mathbf{w}_{ti}$ obtained by a derivative propagation approach:

$$\left(\frac{\partial \boldsymbol{\Gamma}_i}{\partial \mathbf{w}_{ti}}\right)' = \mathbf{ad}_{\boldsymbol{\xi}_i}\frac{\partial \boldsymbol{\Gamma}_i}{\partial \mathbf{w}_{ti}} \tag{A.71}$$

$$\left(\frac{\partial \mathbf{Q}_{ci}}{\partial \mathbf{w}_{ti}}\right)' = (\mathbf{BN})^T \frac{\partial \boldsymbol{\Gamma}_i}{\partial \mathbf{w}_{ti}} \tag{A.72}$$

with initial values $\partial \boldsymbol{\Gamma}_i/\partial \mathbf{w}_{ti}(L) = \mathbf{I}_3$, and $\partial \mathbf{Q}_{ci}/\partial \mathbf{w}_{ti}(L) = \mathbf{0}_{m_i\times 3}$.

Appendix B

Geometrico and Kinemato Static Models Derivation: Finite-Differences Approach

This appendix derives the geometrico-static model equations and the necessary steps for the derivation of the kinemato-static model equations when using the finite-difference approach of [56] for the spatial case (Sec. B.1), and for the planar case of [80] (Sec. B.2).

B.1 Spatial Case

This Section derives the geometrico-static and kinemato-static terms when using the finite-differences approach for a spatial *CPR* case. Additional details on this derivation are reported in [180] for the interested reader. The first part (Sec. B.1.1) focuses on the geometrico-static model derivation, while the second (Sec. B.1.2) derives the terms necessary for the kinemato-static model evaluation.

B.1.1 Geometrico-static model

This section derives the geometrico-static model equations of Eq.(2.23) of a *CPR* when a finite-differences modelling approach is used, here reported for clarity:

$$\begin{cases} \nabla_{\mathbf{x}}\mathcal{L} = \mathbf{0} \\ \nabla_{\boldsymbol{\lambda}}\mathcal{L} = \mathbf{0} \end{cases} \rightarrow \begin{cases} \nabla_{\mathbf{q}_e} V_{tot} + \nabla_{\mathbf{q}_e}\left(\boldsymbol{\lambda}^T\boldsymbol{\Phi}\right) = \mathbf{0} \\ \nabla_{\mathbf{q}_p} V_{tot} + \nabla_{\mathbf{q}_p}\left(\boldsymbol{\lambda}^T\boldsymbol{\Phi}\right) = \mathbf{0} \\ \boldsymbol{\Phi} = \mathbf{0} \end{cases} \tag{B.1}$$

To derive the expression of these equations, let us consider the deformation energy of the i-th beam of Eq. (2.12) when shear and extensibility are neglected, here reported for clarity:

$$V_{ei} = \frac{1}{2}\int_0^{L_i} \left(\mathbf{u}_i(s) - \mathbf{u}_i^*(s)\right)^T \mathbf{K}\left(\mathbf{u}_i(s) - \mathbf{u}_i^*(s)\right) ds \tag{B.2}$$

In the case the finite-differences approach is employed, it is convenient to parametrize the orientation of each cross-section by using unit quaternions $\mathbf{h}_i \in \mathbb{R}^4$, $\mathbf{h}_i = h_{i1} + h_{i2}\mathbf{e}_1 + h_{i3}\mathbf{e}_2 + h_{i4}\mathbf{e}_3$, $\mathbf{e}_1 = [1,0,0], \mathbf{e}_2 = [0,1,0], \mathbf{e}_3 = [0,0,1]$. In this case, the i-th beams curvature $\mathbf{u}_i(s) \in \mathbb{R}^3$ is computed as follows:

$$u_{ik}(s) = 2\mathbf{h}_i^T\mathbf{B}_k^T\mathbf{h}_i'; \quad k = 1,2,3 \tag{B.3}$$

with the matrices $\mathbf{B}_k$ structured as follows:

$$\mathbf{B}_1 = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \end{bmatrix}, \mathbf{B}_2 = \begin{bmatrix} 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \mathbf{B}_3 = \begin{bmatrix} 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \tag{B.4}$$

Thus, inserting Eq. (B.3) into Eq. (B.2), and by assuming $\mathbf{u}_i^* = \mathbf{0}$ for simplicity, the following expression of V_{ei} is obtained:

$$V_{ei} = \frac{1}{2} \int_0^{L_i} \left(\sum_{k=1}^{3} K_k \left(2\mathbf{h}_i^T \mathbf{B}_k \mathbf{h}_i' \right)^2 \right) ds \tag{B.5}$$

The finite-difference approximation is introduced by first discretizing the rod into N_{elt} elements of equal length $L_{ei} = L/N_{elt}$, with the orientation of the j-th element of the i-th beam being defined by $\mathbf{h}_{ij}$. The vector $\mathbf{q}_{ei}$ collects the N_{elt} quaternions $\mathbf{h}_{ij}$ of the i-th beam. The expression of V_{ei} becomes:

$$V_{ei} = \sum_{i=1}^{N_{elt}} V_{eij}; \quad V_{eij} = \frac{1}{2} \int_0^{L_{ei}} \left(\sum_{k=1}^{3} K_k \left(2\mathbf{h}_{ij}^T \mathbf{B}_k \mathbf{h}_{ij}' \right)^2 \right) ds \tag{B.6}$$

The expression of $\mathbf{h}_{ij}'$ is then approximated by the use of a first-order backward finite difference approximation:

$$\mathbf{h}_{ij}' \simeq \frac{\mathbf{h}_{ij} - \mathbf{h}_{ij-1}}{L_{ei}} \tag{B.7}$$

Inserting Eq. (B.7) into Eq. (B.6) and integrating in s results in the following expression of V_{eij}:

$$V_{eij} = \sum_{k=1}^{3} \frac{K_k}{L_e} \left(2\mathbf{h}_{ij}^T \mathbf{B}_k \frac{\mathbf{h}_{ij} - \mathbf{h}_{ij-1}}{L_e} \right)^2 \tag{B.8}$$

Once the term V_{ei} is obtained, to compute $\mathcal{L}$, it is necessary to evaluate the influence of distributed loads V_{di} (Eq. (2.13)) and the geometric constraints of the i-th beam $\mathbf{\Phi}_i$. First, the position of each beam's element $\mathbf{p}_{ij}$ is recovered by the use of the following formula:

$$\mathbf{p}_i(s) = \mathbf{p}_{ij} + s\mathbf{R}_{ij}\mathbf{e}_3 \tag{B.9}$$

with $\mathbf{R}_{ij}$ the orientation matrix of the j-th element, recovered by the knowledge of $\mathbf{h}_{ij}$. $\mathbf{p}_0, \mathbf{h}_0$, that are the base position and orientation of the beam, are usually computed from q_{ai}, depending on the employed actuator. To compute V_{di}, let us recall its expression of Eq. (2.13) for clarity:

$$V_{di} = -\int_0^L \mathbf{f}_d^T \mathbf{p}_i ds \tag{B.10}$$

By discretizing the beam into N_{elt} equal elements, inserting Eq. (B.9) into Eq. (B.10), and integrating in s, the following expression is obtained:

$$V_{di} = \sum_{j=1}^{N_{elt}} V_{dij} \quad V_{dij} = -\mathbf{f}_d^T \left(\mathbf{p}_{ij} + L_e \mathbf{R}_{ij} \mathbf{e}_3 \right) \tag{B.11}$$

At this stage, the derivation of the geometrico-static model equations starts. Computing $\mathbf{\Phi}_i$ is possible by inserting the values of $\mathbf{p}_{ij}, \mathbf{R}_{ij}$ of the last beams element computed from Eq. (B.9) in Eq. (2.18), here reported for clarity:

$$\mathbf{\Phi}_i = \mathbf{C}_i \begin{bmatrix} \left(\mathbf{R}_p^T \mathbf{R}_{ij} - \mathbf{R}_{ij} \mathbf{R}_p \right)^\vee \\ \mathbf{p}_{ij} - (\mathbf{p}_p + \mathbf{R}_p \mathbf{p}_{fi}) \end{bmatrix} \; j = N_{elt} \tag{B.12}$$

Let us derive $\nabla_{\mathbf{q}_{ei}}\mathcal{L}$, by considering separately the contribution of the deformation energy, distributed loads energy, and geometrical constraints:

$$\nabla_{\mathbf{q}_{ei}}\mathcal{L} = \nabla_{\mathbf{q}_{ei}}V_{ei} + \nabla_{\mathbf{q}_{ei}}V_{di} + \nabla_{\mathbf{q}_{ei}}\mathbf{\Phi}_i\boldsymbol{\lambda}_i \tag{B.13}$$

For each term, the expression of $\nabla_{\mathbf{h}_{ij}}(.)$, that is the j-th component of $\nabla_{\mathbf{q}_{ei}}(.)$, is derived. For the scope, let us expand the expression of V_{ei} as follows to illustrate where $\mathbf{h}_{ij}$ appears in V_{ei}:

$$V_{ei} = \sum_{j=1}^{N_{elt}}\sum_{k=1}^{3}\frac{K_k}{L_{ei}}\left(\cdots + \left(2\mathbf{h}_{ij}^T\mathbf{B}_k\frac{\mathbf{h}_{ij}-\mathbf{h}_{ij-1}}{L_e}\right)^2 + \left(2\mathbf{h}_{ij+1}^T\mathbf{B}_k\frac{\mathbf{h}_{ij+1}-\mathbf{h}_{ij}}{L_e}\right)^2 + \cdots\right) \tag{B.14}$$

The term $\mathbf{h}_{ij}$, as a cause of the finite-difference approximation, appears in the term j and $j+1$. Thus, $\nabla_{\mathbf{h}_{ij}}V_{ei}$ is structured as the sum of two terms:

$$\nabla_{\mathbf{h}_{ij}}V_{ei} = \mathbf{a}_{ij} + \mathbf{b}_i \tag{B.15}$$

and the terms $\mathbf{a}_{ij}, \mathbf{b}_{ik}, A_{ijk}$ can be computed as follows:

$$\mathbf{a}_{ij} = +2\sum_{k=1}^{3}\frac{K_k}{L_{ei}}\mathbf{B}_k\mathbf{h}_{ij}A_{ijk},\ \mathbf{a}_{ij}\in\mathbb{R}^4 \tag{B.16}$$

$$\mathbf{b}_{ij} = -2\sum_{k=1}^{3}\frac{K_k}{L_{ei}}\mathbf{B}_k\mathbf{h}_{i(j+1)}A_{ijk},\ \mathbf{b}_{ij}\in\mathbb{R}^4 \tag{B.17}$$

$$A_{ijk} = -2\mathbf{h}_{i(j+1)}^T\mathbf{B}_k^T\mathbf{h}_{ij},\ A_{ijk}\in\mathbb{R} \tag{B.18}$$

Then, the term $\nabla_{\mathbf{h}_{ij}}V_{di}$ is calculated by differentiating Eq.(B.11) w.r.t. $\mathbf{h}_{ij}$:

$$\nabla_{\mathbf{h}_{ij}}V_{di} = \sum_{r=1}^{N_{elt}}\mathbf{f}_d^T\left(\nabla_{\mathbf{h}_{ij}}\mathbf{p}_{ir} + L_{ei}\nabla_{\mathbf{h}_{ij}}\left(\mathbf{R}_{ir}\mathbf{e}_3\right)\right) \tag{B.19}$$

with the terms $\nabla_{\mathbf{h}_{ij}}\mathbf{p}_{ir}$ and $\nabla_{\mathbf{h}_{ij}}\left(\mathbf{R}_{ir}\mathbf{e}_3\right)$ compute as follows:

$$\nabla_{\mathbf{h}_{ij}}\mathbf{p}_{ir} = L_{ei}\mathbf{D}_{3ij};\ j\leqslant r;\quad \nabla_{\mathbf{h}_{ij}}\mathbf{p}_{ir} = \mathbf{0};\ j>r \tag{B.20}$$

$$\nabla_{\mathbf{h}_{ij}}\left(\mathbf{R}_{ir}\mathbf{e}_3\right) = \mathbf{D}_{3ij};\ j=r;\quad \nabla_{\mathbf{h}_{ij}}\left(\mathbf{R}_{ir}\mathbf{e}_3\right) = \mathbf{0};\ j\neq r \tag{B.21}$$

The matrix $\mathbf{D}_{3ij}$ (and $\mathbf{D}_{1ij}, \mathbf{D}_{2ij}$ introduced for later convenience), are structured as:

$$\mathbf{D}_{1ij} = 2\begin{bmatrix} +h_{1ij} & +h_{2ij} & -h_{3ij} & -h_{4ij} \\ +h_{4ij} & +h_{3ij} & +h_{2ij} & +h_{1ij} \\ -h_{3ij} & +h_{4ij} & -h_{1ij} & +h_{2ij} \end{bmatrix} \tag{B.22}$$

$$\mathbf{D}_{2ij} = 2\begin{bmatrix} -h_{4ij} & +h_{3ij} & +h_{2ij} & -h_{1ij} \\ +h_{1ij} & -h_{2ij} & +h_{3ij} & -h_{4ij} \\ +h_{2ij} & +h_{1ij} & +h_{4ij} & +h_{3ij} \end{bmatrix} \tag{B.23}$$

$$\mathbf{D}_{3ij} = 2\begin{bmatrix} +h_{3ij} & +h_{4ij} & +h_{1ij} & +h_{2ij} \\ -h_{2ij} & -h_{1ij} & +h_{4ij} & +h_{3ij} \\ +h_{1ij} & -h_{2ij} & -h_{3ij} & +h_{4ij} \end{bmatrix} \tag{B.24}$$

Finally, to compute $\nabla_{\mathbf{q}_{ei}}\left(\mathbf{\Phi}_i^T\boldsymbol{\lambda}_i\right)$, the j-th component $\nabla_{\mathbf{h}_{ij}}\left(\mathbf{\Phi}_i^T\boldsymbol{\lambda}_i\right)$ is calculated. For the following derivation, it is convenient to simplify the expression of the orientation contraints. Being $\mathbf{d}_{p1}, \mathbf{d}_{p2}, \mathbf{d}_{p3}$ the columns of the platform rotation matrix $\mathbf{R}_p$ =

$[\mathbf{d}_{p1}, \mathbf{d}_{p2}, \mathbf{d}_{p3}]$, and $\mathbf{d}_{i1}, \mathbf{d}_{i2}, \mathbf{d}_{i3}$ the columns of the i-th beam rotation matrix at $s = L$ ($\mathbf{R}_{ij} = [\mathbf{d}_{i1}, \mathbf{d}_{i2}, \mathbf{d}_{i3}]$,$j = N_{elt}$), it is possible to obtain the following expression:

$$\left(\mathbf{R}_p^T \mathbf{R}_i(L_i) - \mathbf{R}_i^T(L_i)\mathbf{R}_p\right)^{\smile} = \begin{cases} \mathbf{d}_{p1}^T\mathbf{d}_2 - \mathbf{d}_{p2}^T\mathbf{d}_1 \\ \mathbf{d}_{p1}^T\mathbf{d}_3 - \mathbf{d}_{p3}^T\mathbf{d}_1 \\ \mathbf{d}_{p2}^T\mathbf{d}_3 - \mathbf{d}_{p3}^T\mathbf{d}_2 \end{cases} \tag{B.25}$$

Thus, $\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i$ is explicitly computed as follows:

$$\begin{aligned} \boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i = [\lambda_{4i}, \lambda_{5i}, \lambda_{6i}] \left(\mathbf{p}_i(L_i) - (\mathbf{p}_p + \mathbf{R}_p \mathbf{p}_{fi})\right) + \lambda_{1i}\left(\mathbf{d}_{p1}^T\mathbf{d}_2 - \mathbf{d}_{p2}^T\mathbf{d}_1\right) \\ + \lambda_{2i}\left(\mathbf{d}_{p1}^T\mathbf{d}_3 - \mathbf{d}_{p3}^T\mathbf{d}_1\right) + \lambda_{3i}\left(\mathbf{d}_{p2}^T\mathbf{d}_3 - \mathbf{d}_{p3}^T\mathbf{d}_2\right) \end{aligned} \tag{B.26}$$

Deriving Eq. (B.26) w.r.t. $\mathbf{h}_{ij}$ results in two terms:

$$\nabla_{\mathbf{h}_{ij}}(\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i) = \mathbf{c}_{ij} + \mathbf{d}_{ij} \tag{B.27}$$

where $\mathbf{c}_{ij} \in \mathbb{R}^4$:

$$\mathbf{c}_{ij} = L_{ei}[\lambda_{4i}, \lambda_{5i}, \lambda_{6i}]\mathbf{D}_{3ij} \tag{B.28}$$

Instead, the term $\mathbf{d}_{ij} \in \mathbb{R}^4$ is non-null only for $j = N_{elt}$, and it is structured as follows:

$$\begin{aligned} \mathbf{d}_{ij} = \lambda_{1i}\left(\mathbf{d}_{p1}^T\mathbf{D}_{2ij} - \mathbf{d}_{p2}^T\mathbf{D}_{1ij}\right) + \lambda_{2i}\left(\mathbf{d}_{p1}^T\mathbf{D}_{3ij} - \mathbf{d}_{p3}^T\mathbf{D}_{1ij}\right) + \\ + \lambda_{3i}\left(\mathbf{d}_{p2}^T\mathbf{D}_{3ij} - \mathbf{d}_{p3}^T\mathbf{D}_{2ij}\right),\ j = N_{elt} \end{aligned} \tag{B.29}$$

Finally, the last required term for the geometrico-static model derivation is $\nabla_{\mathbf{q}_p}\mathcal{L}$, that can be computed as:

$$\nabla_{\mathbf{q}_p}\mathcal{L} = \nabla_{\mathbf{q}_p} V_{tot} + \nabla_{\mathbf{q}_p}(\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i) \tag{B.30}$$

where $\nabla_{\mathbf{q}_p} V = -[\mathbf{0}_{3\times1}; \mathbf{I}_3]$ and, being $\mathbf{q}_p = [\mathbf{p}_p, \boldsymbol{\alpha}] \in \mathbb{R}^{n_c}$ with $\mathbf{p}_p \in \mathbb{R}^3$ the platform position and $\boldsymbol{\alpha} \in \mathbb{R}^{n_c-3}$ the platform orientation parameters, the term $\nabla_{\mathbf{q}_p}(\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i)$ is structured as follows:

$$\nabla_{\mathbf{q}_p}(\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i) = -[\lambda_{4i}, \lambda_{5i}, \lambda_{6i}] \begin{bmatrix} \mathbf{I}_3 \\ \mathbf{m}_{p1} \end{bmatrix} + \begin{bmatrix} \mathbf{0}_{3\times1} \\ \mathbf{m}_{p2} \end{bmatrix} \tag{B.31}$$

the vector $\mathbf{m}_{p1} \in \mathbb{R}^{n_c-3}$ is:

$$\mathbf{m}_{p1} = \frac{\partial}{\partial \boldsymbol{\alpha}} \left(\mathbf{R}_p \mathbf{p}_{fi}\right) \tag{B.32}$$

and its expression depends on the specific platform orientation parametrization. The expression of $\mathbf{m}_{p2} \in \mathbb{R}^{n_c-3}$ depends on the platform orientation parametrization as well, and it is structured as:

$$\begin{aligned} \mathbf{m}_{p2} = \lambda_{1i}\left(\frac{\partial \mathbf{d}_{p1}^T}{\partial \boldsymbol{\alpha}}\mathbf{d}_{2ij} - \frac{\partial \mathbf{d}_{p2}^T}{\partial \boldsymbol{\alpha}}\mathbf{d}_{1ij}\right) + \lambda_{2i}\left(\frac{\partial \mathbf{d}_{p1}^T}{\partial \boldsymbol{\alpha}}\mathbf{d}_{3ij} - \frac{\partial \mathbf{d}_{p3}^T}{\partial \boldsymbol{\alpha}}\mathbf{d}_{1ij}\right) + \\ + \lambda_{3i}\left(\frac{\partial \mathbf{d}_{p1}^T}{\partial \boldsymbol{\alpha}}\mathbf{d}_{3ij} - \frac{\partial \mathbf{d}_{p3}^T}{\partial \boldsymbol{\alpha}}\mathbf{d}_{2ij}\right),\ j = N_{elt} \end{aligned} \tag{B.33}$$

with $\partial \mathbf{d}_{pk}^T / \partial \boldsymbol{\alpha}$ to be computed in relation to the selected platform orientation parametrization.

B.1.2 Kinemato-static model

The derivation of the kinemato-static model requires the computation of the first derivative of each of Eq. (B.1) w.r.t. $\mathbf{y} = [\mathbf{q}_a, \mathbf{q}_e, \mathbf{q}_p, \boldsymbol{\lambda}]$. In particular, the derivative of Eq.(B.1) w.r.t. $\mathbf{y}$ is structured as follows:

$$\begin{bmatrix} \nabla_{\mathbf{q}_a}(\nabla_{\mathbf{q}_e}\mathcal{L}) & \nabla_{\mathbf{q}_e}(\nabla_{\mathbf{q}_e}\mathcal{L}) & \nabla_{\mathbf{q}_e}(\nabla_{\mathbf{q}_e}\mathcal{L}) & \nabla_{\boldsymbol{\lambda}}(\nabla_{\mathbf{q}_e}\mathcal{L}) \\ \nabla_{\mathbf{q}_a}(\nabla_{\mathbf{q}_p}\mathcal{L}) & \nabla_{\mathbf{q}_e}(\nabla_{\mathbf{q}_p}\mathcal{L}) & \nabla_{\mathbf{q}_e}(\nabla_{\mathbf{q}_p}\mathcal{L}) & \nabla_{\boldsymbol{\lambda}}(\nabla_{\mathbf{q}_p}\mathcal{L}) \\ \nabla_{\mathbf{q}_a}\boldsymbol{\Phi} & \nabla_{\mathbf{q}_e}\boldsymbol{\Phi} & \nabla_{\mathbf{q}_e}\boldsymbol{\Phi} & \nabla_{\boldsymbol{\lambda}}\boldsymbol{\Phi} \end{bmatrix} \tag{B.34}$$

Let us start by computing the derivatives of $\nabla_{\mathbf{q}_{ei}}\mathcal{L}$: its first derivative w.r.t. $\mathbf{q}_{ei}$ is obtained by considering separately V_{ei}, V_{di} and $\boldsymbol{\lambda}_i^T\boldsymbol{\Phi}_i$. The term $\nabla_{\mathbf{q}_{ei}}V_{ei}$ depends on $\mathbf{q}_{ei}, q_{ai}$ only, and it derivative w.r.t. $\mathbf{q}_{ei}$ is computed according to Eq. (B.27) as a diagonal block matrix:

$$\nabla_{\mathbf{q}_{ei}}(\nabla_{\mathbf{q}_{ei}}V_{ei}) = \begin{bmatrix} \mathbf{E}_{i1} & \mathbf{F}_{i2}^T & \cdots & \mathbf{0} & \mathbf{0} \\ \mathbf{F}_{i2} & \mathbf{E}_{i2} & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{E}_{iN_{elt}-1} & \mathbf{F}_{iN_{elt}}^T \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{F}_{iN_{elt}} & \mathbf{E}_{iN_{elt}} \end{bmatrix} \tag{B.35}$$

in which:

$$\mathbf{E}_{ij} = \mathbf{M}_{ij} + \mathbf{N}_{ij+1} \tag{B.36}$$

$$\mathbf{F}_{ij} = -4\sum_{k=1}^{3}\frac{K_k}{L_{ei}}((\mathbf{B}_k\mathbf{h}_{ij})(\mathbf{B}_k\mathbf{h}_{ij+1})^T - \mathbf{B}_k A_{ijk}/2) \tag{B.37}$$

$$\mathbf{M}_{ij} = -4\sum_{k=1}^{3}\frac{K_k}{L_{ei}}(\mathbf{B}_k\mathbf{h}_{ij})(\mathbf{B}_k^T\mathbf{h}_{ij})^T \tag{B.38}$$

$$\mathbf{N}_{ij} = -4\sum_{k=1}^{3}\frac{K_k}{L_{ei}}(\mathbf{B}_k\mathbf{h}_{ij+1})(\mathbf{B}_k\mathbf{h}_{ij+1})^T \tag{B.39}$$

Concering $\nabla_{q_{ai}}(\nabla_{\mathbf{q}_{ei}}V_{ei})$, this derivative depends on the specific actuators employed. In the case a prismatic actuator is used, $\nabla_{q_{ai}}(\nabla_{\mathbf{q}_{ei}}V_{ei}) = \mathbf{0}$ while, for rotative actuators:

$$\nabla_{q_{ai}}(\nabla_{\mathbf{q}_{ei}}V_{ei}) = \frac{\partial}{\partial\mathbf{h}_0}(\nabla_{\mathbf{q}_{ei}}V_{ei})\frac{\partial\mathbf{h}_0}{\partial q_{ai}} \tag{B.40}$$

with the term $\partial/\partial\mathbf{h}_0(\nabla_{\mathbf{q}_{ei}}V_{ei})$ obtainable from Eq. (B.39), and $\partial\mathbf{h}_0/\partial q_{ai}$ depends on the specific actuator installation. All the other derivatives successive derivatives of $\nabla_{\mathbf{q}_{ei}}V_{ei}$ are null.

Let us consider now $\nabla_{\mathbf{q}_{ei}}V_{di}$, that depends on $\mathbf{q}_{ei}$ only. In particular, only $\mathbf{h}_{ij}$ appears. To compute derivative of $\nabla_{\mathbf{h}_{ij}}V_{di}$ w.r.t. $\mathbf{h}_{ij}$, it is first convenient to insert $\mathbf{f}_d = [f_{dx}, f_{dy}, f_{dz}]$ inside Eq. (B.19), to obtain the following expression:

$$\nabla_{\mathbf{h}_{ij}}V_{di} = \sum_{r=1}^{N_{elt}}\nabla_{\mathbf{h}_{ij}}\left(\mathbf{f}_d^T\mathbf{p}_{ir}\right) + L_{ei}\nabla_{\mathbf{h}_{ij}}\left(\mathbf{f}_d^T\mathbf{R}_{ir}\mathbf{e}_3\right) \tag{B.41}$$

For the derivation, it is convenient to define the following operator that transforms the generic $\mathbf{z} = [z_1, z_2, z_3] \in \mathbb{R}^3$ into a 4×4 matrix structured as follows:

$$\widetilde{\mathbf{z}} = \begin{bmatrix} +z_3 & -z_2 & +z_1 & 0 \\ -z_2 & -z_3 & 0 & +z_1 \\ +z_1 & 0 & -z_3 & +z_2 \\ 0 & +z_1 & +z_2 & +z_3 \end{bmatrix} \tag{B.42}$$

The derivative of $\nabla_{\mathbf{h}_{ij}} V_{di}$ w.r.t. $\mathbf{h}_{ij}$ is expressed as follows:

$$\nabla_{\mathbf{h}_{ij}} \left(\nabla_{\mathbf{h}_{ij}} \left(\mathbf{f}_d^T \mathbf{p}_{ir} \right) \right) = 4\widetilde{\mathbf{f}}_d \tag{B.43}$$

Then, let us consider $\nabla_{\mathbf{q}_{ei}} \boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i$ of Eq.(B.27), that depends on $\mathbf{q}_{ei}, \boldsymbol{\lambda}_i$. First, let us compute the derivative of Eq.(B.27) w.r.t. $\mathbf{h}_{ij}$, that is:

$$\nabla_{\mathbf{h}_{ij}} \left(\nabla_{\mathbf{h}_{ij}} (\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i) \right) = \nabla_{\mathbf{h}_i} \mathbf{c}_i + \nabla_{\mathbf{h}_i} \mathbf{d}_i \tag{B.44}$$

Considering Eq.(B.28), and by introducing $\boldsymbol{\lambda}_{4:6i} = [\lambda_{4i}, \lambda_{5i}, \lambda_{6i}]$, the expression $\nabla_{\mathbf{h}_i} \mathbf{c}_i$ is obtained as follows:

$$\nabla_{\mathbf{h}_i} \mathbf{c}_{ij} = 2L_{ei} \widetilde{\boldsymbol{\lambda}_{4:6i}} \tag{B.45}$$

The term $\nabla_{\mathbf{h}_{ij}} \mathbf{d}_{ij}$ is non-null only for $j = N_{elt}$, and it is structured as follows:

$$\nabla_{\mathbf{h}_{ij}} \mathbf{d}_{ij} = \lambda_{1i} \left(\widetilde{\mathbf{d}_{p1}} - \widetilde{\mathbf{d}_{p2}} \right) + \lambda_{2i} \left(\widetilde{\mathbf{d}_{p1}} - \widetilde{\mathbf{d}_{p3}} \right) + \lambda_{3i} \left(\widetilde{\mathbf{d}_{p2}} - \widetilde{\mathbf{d}_{p3}} \right), \; j = N_{elt} \tag{B.46}$$

Let us calculate $\nabla_{\boldsymbol{\lambda}_i} (\nabla_{\mathbf{q}_{ei}} \boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i)$. Please note that this term also correspond to $\nabla_{\mathbf{q}_{ei}} \boldsymbol{\Phi}_i$. Derivating $\mathbf{c}_{ij}$ w.r.t. $\boldsymbol{\lambda}_i$ results in:

$$\nabla_{\boldsymbol{\lambda}_i} \mathbf{c}_{ij} = L_{ei} \begin{bmatrix} \mathbf{0}_{3\times 4} \\ \mathbf{D}_{3ij} \end{bmatrix} \tag{B.47}$$

while, by differentiating $\mathbf{d}_{ij}$ w.r.t. $\boldsymbol{\lambda}_i$, the following expression is obtained:

$$\nabla_{\boldsymbol{\lambda}_i} \mathbf{d}_{ij} = \begin{bmatrix} \mathbf{d}_{p1}^T \mathbf{D}_{2ij} - \mathbf{d}_{P2}^T \mathbf{D}_{1ij} \\ \mathbf{d}_{p1}^T \mathbf{D}_{3ij} - \mathbf{d}_{P3}^T \mathbf{D}_{1ij} \\ \mathbf{d}_{p2}^T \mathbf{D}_{3ij} - \mathbf{d}_{P3}^T \mathbf{D}_{2ij} \\ \mathbf{0}_{3\times 4} \end{bmatrix}, \; j = N_{elt} \tag{B.48}$$

Let us compute the derivatives of $\nabla_{\mathbf{q}_p} \mathcal{L}$, which depends on $\boldsymbol{\lambda}, \mathbf{q}_e$ and $\mathbf{q}_p$. In particular, $\nabla_{\mathbf{q}_p} V_{tot}$ is constant, and its first derivatives vanishes. Instead, let us consider $\nabla_{\mathbf{q}_p} \boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i$ of Eq.(B.31), where a differentiation w.r.t. $\boldsymbol{\lambda}$ results in:

$$\nabla_{\boldsymbol{\lambda}_i} \left(\nabla_{\mathbf{q}_p} \left(\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i \right) \right) = \begin{bmatrix} \mathbf{0}_{3\times 3} & -\mathbf{I}_{3\times 3} \\ \nabla_{\boldsymbol{\lambda}_{4:6i}} \mathbf{m}_{p2} & \mathbf{0}_{3\times 3} \end{bmatrix} \tag{B.49}$$

where:

$$\nabla_{\boldsymbol{\lambda}_{4:6i}} \mathbf{m}_{p2} = \begin{bmatrix} \frac{\partial \mathbf{d}_{p1}^T}{\partial \boldsymbol{\alpha}} \mathbf{d}_{2ij} - \frac{\partial \mathbf{d}_{p2}^T}{\partial \boldsymbol{\alpha}} \mathbf{d}_{1ij}, & \frac{\partial \mathbf{d}_{p1}^T}{\partial \boldsymbol{\alpha}} \mathbf{d}_{3ij} - \frac{\partial \mathbf{d}_{p3}^T}{\partial \boldsymbol{\alpha}} \mathbf{d}_{1ij}, & \frac{\partial \mathbf{d}_{p1}^T}{\partial \boldsymbol{\alpha}} \mathbf{d}_{3ij} - \frac{\partial \mathbf{d}_{p3}^T}{\partial \boldsymbol{\alpha}} \mathbf{d}_{2ij} \end{bmatrix} \tag{B.50}$$

Please note that $\nabla_{\boldsymbol{\lambda}_i} \nabla_{\mathbf{q}_p} \left(\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i \right)$ also corresponds to $\nabla_{\mathbf{q}_p} \boldsymbol{\Phi}_i$. The derivative of $\nabla_{\mathbf{q}_p} \boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i$ w.r.t. $\mathbf{h}_{ij}$ is non-null only for $j = N_{elt}$, and it computed as follows:

$$\nabla_{\mathbf{h}_{ij}} \left(\nabla_{\mathbf{q}_p} \boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i \right) = \begin{bmatrix} \mathbf{0}_{3\times 4} \\ \nabla_{\mathbf{h}_{ij}} \mathbf{m}_{p2} \end{bmatrix} \tag{B.51}$$

in which:

$$\begin{aligned} \nabla_{\mathbf{h}_{ij}} \mathbf{m}_{p2} = \lambda_{1i} \left(\frac{\partial \mathbf{d}_{p1}^T}{\partial \boldsymbol{\alpha}} \mathbf{D}_{2ij} - \frac{\partial \mathbf{d}_{p2}^T}{\partial \boldsymbol{\alpha}} \mathbf{D}_{1ij} \right) + \lambda_{2i} \left(\frac{\partial \mathbf{d}_{p1}^T}{\partial \boldsymbol{\alpha}} \mathbf{D}_{3ij} - \frac{\partial \mathbf{d}_{p3}^T}{\partial \boldsymbol{\alpha}} \mathbf{D}_{1ij} \right) + \\ + \lambda_{3i} \left(\frac{\partial \mathbf{d}_{p1}^T}{\partial \boldsymbol{\alpha}} \mathbf{D}_{3ij} - \frac{\partial \mathbf{d}_{p3}^T}{\partial \boldsymbol{\alpha}} \mathbf{D}_{2ij} \right), \; j = N_{elt} \end{aligned} \tag{B.52}$$

The derivative of Eq.(B.31) w.r.t. $\mathbf{q}_p$ depends on the specific orientation parametrization, and the derivation of its expression is left to the interested reader.

B.2 Planar Case

This Section derives the geometrico-static and kinemato-static terms when using the finite-differences approach for a planar *CPR* case. The first part (Sec. B.2.1) focuses on the geometrico-static model derivation, while the second (Sec. B.2.2) derives the terms necessary for the kinemato-static model evaluation.

B.2.1 Geometrico-static model

This section derives the geometrico-static model equations of Eq.(2.23) of a *CPR* when the assumed strain mode approach is used in a planar case, here reported for clarity:

$$\begin{cases} \nabla_{\mathbf{x}}\mathcal{L} = \mathbf{0} \\ \nabla_{\boldsymbol{\lambda}}\mathcal{L} = \mathbf{0} \end{cases} \rightarrow \begin{cases} \nabla_{\mathbf{q}_e} V_{tot} + \nabla_{\mathbf{q}_e}\left(\boldsymbol{\lambda}^T\boldsymbol{\Phi}\right) = \mathbf{0} \\ \nabla_{\mathbf{q}_p} V_{tot} + \nabla_{\mathbf{q}_p}\left(\boldsymbol{\lambda}^T\boldsymbol{\Phi}\right) = \mathbf{0} \\ \boldsymbol{\Phi} = \mathbf{0} \end{cases} \tag{B.53}$$

To derive the expression of these equations, let us consider the deformation energy of the i-th beam of Eq. (2.12) when shear and extensibility are neglected, and a planar case is considered, here reported for clarity:

$$V_{ei} = \frac{1}{2}\int_0^{L_i} K_b \left(u_i(s) - u_i^*(s)\right)^T \left(u_i(s) - u_i^*(s)\right) ds \tag{B.54}$$

with K_b being the material stiffness (usually equal to EI). As in a planar case, the pose of each cross-section is described by a position vector $\mathbf{p} \in \mathbb{R}^2$ and the angle $\theta \in \mathbb{R}$. In this case, the i-th beams curvature $u_i(s) \in \mathbb{R}$ is computed as follows:

$$u_i(s) = \theta'(s) \tag{B.55}$$

The finite-difference approximation is introduced by first discretizing the rod into N_{elt} elements of equal length $L_{ei} = L/N_{elt}$, with the orientation of the j-th element of the i-th beam being defined by θ_{ij}. The expression of V_{ei} becomes:

$$V_{ei} = \sum_{i=1}^{N_{elt}} V_{eij}; \quad V_{eij} = \frac{1}{2}\int_0^{L_{ei}} \left(K_b\theta_{ij}^2\right) ds \tag{B.56}$$

The expression of θ'_{ij} is then approximated by the use of a first-order backward finite difference approximation:

$$\theta'_{ij} \simeq \frac{\theta_{ij} - \theta_{ij-1}}{L_{ei}} \tag{B.57}$$

Inserting Eq. (B.57) into Eq. (B.56) and integrating in s results in the following expression of V_{eij}:

$$V_{eij} = \frac{K_b}{L_e}\left(\theta_{ij} - \theta_{ij-1}\right)^2 \tag{B.58}$$

Once the term V_{ei} is obtained, to compute $\mathcal{L}$, it is necessary to evaluate the influence of distributed loads V_{di} (Eq. (2.13)) and the geometric constraints of the i-th beam $\boldsymbol{\Phi}_i$. First, the position of each beam's element $\mathbf{p}_{ij}$ is recovered by the use of the following formula:

$$\mathbf{p}_i(s) = \mathbf{p}_{ij} + s\mathbf{R}_{ij}\mathbf{e}_2^b \tag{B.59}$$

with $\mathbf{R}_{ij}$ the orientation matrix of the j-th element, recovered by the knowledge of θ_{ij} and $\mathbf{e}_2^b = [0, 1]$, with the tangent to the beam's centerline assumed to be parallel to the local y axis. $\mathbf{p}_0, \theta_0$, that are the base position and orientation of the beam, are usually

computed from q_{ai}, depending on the employed actuator. To compute V_{di}, let us recall its expression of Eq. (2.13) for clarity:

$$V_{di} = -\int_0^L \mathbf{f}_d^T \mathbf{p}_i ds \tag{B.60}$$

By discretizing the beam into N_{elt} equal elements, inserting Eq. (B.59) into Eq. (B.60), and integrating in s, the following expression is obtained:

$$V_{di} = \sum_{j=1}^{N_{elt}} V_{dij} \qquad V_{dij} = -\mathbf{f}_d^T \left(\mathbf{p}_{ij} + L_e \mathbf{R}_{ij} \mathbf{e}_3\right) \tag{B.61}$$

At this stage, the derivation of the geometrico-static model equations starts. Let us first consider the geometrico-static model of Eq. (2.20), reported here for clarity:

$$\begin{cases} \nabla_{\mathbf{x}} \mathcal{L} = \mathbf{0} \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L} = \mathbf{0} \end{cases} \rightarrow \begin{cases} \nabla_{\mathbf{q}_{ei}} V_{tot} + \nabla_{\mathbf{q}_{ei}} \boldsymbol{\Phi}^T \boldsymbol{\lambda} = \mathbf{0};\ \forall i \\ \nabla_{\mathbf{q}_p} V_{tot} + \nabla_{\mathbf{q}_p} \boldsymbol{\Phi}^T \boldsymbol{\lambda} = \mathbf{0} \\ \boldsymbol{\Phi}_i = \mathbf{0};\ \forall i \end{cases} \tag{B.62}$$

In particular, $\mathbf{q}_{ei}$ collects the N_{elt} angles θ_{ij} of the i-th beam. Computing $\boldsymbol{\Phi}_i$ is possible by inserting the values of $\mathbf{p}_{ij}, \theta_{ij}$ of the last beams element in Eq. (2.18) computed from Eq. (B.59).

Let us derive $\nabla_{\mathbf{q}_{ei}} \mathcal{L}$, by considering separately the contribution of the deformation energy, distributed loads energy, and geometrical constraints:

$$\nabla_{\mathbf{q}_{ei}} \mathcal{L} = \nabla_{\mathbf{q}_{ei}} V_{ei} + \nabla_{\mathbf{q}_{ei}} V_{di} + \nabla_{\mathbf{q}_{ei}} \boldsymbol{\Phi}_i \boldsymbol{\lambda}_i \tag{B.63}$$

For each term, it is sufficient to calculate $\nabla_{\mathbf{h}_{ij}}(.)$ for each j to obtain $\nabla_{\mathbf{q}_{ei}}(.)$. To derive $\nabla_{\mathbf{h}_{ij}} V_{ei}$, let us expand the expression of V_{ei} as follows:

$$V_{ei} = \sum_{j=1}^{N_{elt}} \frac{K_b}{L_{ei}} \left(\cdots + (\theta_{ij} - \theta_{ij-1})^2 + (\theta_{ij+1} - \theta_{ij})^2 + \cdots \right) \tag{B.64}$$

The term θ_{ij}, as a cause of the finite-difference approximation, appears in the term j and $j+1$. Thus, $\nabla_{\theta_{ij}} V_{ei}$ is structured as follows:

$$\nabla_{\theta_{ij}} V_{ei} = \frac{K_b}{L_{ei}} \left(-\theta_{ij-1} + 2\theta_{ij} - \theta_{ij+1}\right) \tag{B.65}$$

Then, let us compute $\nabla_{\theta_{ij}} V_{di}$:

$$\nabla_{\theta_{ij}} V_{di} = \sum_{r=1}^{N_{elt}} \mathbf{f}_d^T \left(\nabla_{\theta_{ij}} \mathbf{p}_{ir} + L_{ei} \nabla_{\theta_{ij}} \left(\mathbf{R}_{ir} \mathbf{e}_2^b \right) \right) \tag{B.66}$$

with the terms $\nabla_{\theta_{ij}} \mathbf{p}_{ir}$ and $\nabla_{\theta_{ij}} \left(\mathbf{R}_{ir} \mathbf{e}_2^b \right)$ computed as follows:

$$\nabla_{\theta_{ij}} \mathbf{p}_{ir} = L_{ei} \mathbf{R}_{\pi/2} \mathbf{e}_2^p;\ j \leqslant r; \quad \nabla_{\theta_{ij}} \mathbf{p}_{ir} = \mathbf{0};\ j > r \tag{B.67}$$

$$\nabla_{\theta_{ij}} \left(\mathbf{R}_{ir} \mathbf{e}_2^p \right) = \mathbf{R}_{\pi/2} \mathbf{e}_2^p;\ j = r; \quad \nabla_{\theta_{ij}} \left(\mathbf{R}_{ir} \mathbf{e}_3 \right) = \mathbf{0};\ j \neq r \tag{B.68}$$

with $\mathbf{R}_\pi = \mathbf{R}_z(\theta_{ij} + \pi)$.

Finally, to compute $\nabla_{\theta_{ij}} \boldsymbol{\Phi}_i^T \boldsymbol{\lambda}_i$, it is convenient to recall the expression of $\boldsymbol{\Phi}_i$ from Eq. (2.18). In this Appendix, the results are derived for fixed joints only ($\mathbf{C}_i = \mathbf{I}_3$) since this case includes all the others, and thus Eq. (2.18) is written as follows:

$$\boldsymbol{\Phi}_i = \begin{bmatrix} \theta_i(L) - \theta_p \\ \mathbf{p}_i(L_i) - (\mathbf{p}_p + \mathbf{R}_p \mathbf{p}_{fi}) \end{bmatrix} \tag{B.69}$$

with θ_p being the platform orientation angle that defines $\mathbf{R}_p$. Calculating $\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i$ with $\boldsymbol{\lambda}_i = [\lambda_{1i}, \lambda_{2i}, \lambda_{3i}]$ results in:

$$\lambda_{1i}(\theta_i(L) - \theta_p) + [\lambda_{2i}, \lambda_{3i}]^T \left(\mathbf{p}_i(L_i) - (\mathbf{p}_p + \mathbf{R}_p \mathbf{p}_{fi})\right) \tag{B.70}$$

Deriving Eq. (B.70) w.r.t. θ_{ij} results in two terms:

$$\nabla_{\theta_{ij}}(\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i) = c_{ij} + d_{ij} \tag{B.71}$$

where c_{ij}:

$$c_{ij} = L_{ei}[\lambda_{2i}, \lambda_{3i}]^T \left(\mathbf{R}_{\pi/2} \mathbf{e}_2^p\right) \tag{B.72}$$

The term d_{ij} is non-null only if $j = N_{elt}$, where in that case is $d_{ij} = 1$.

Finally, the last required term for the geometrico-static model derivation is $\nabla_{\mathbf{q}_p}\mathcal{L}$, that can be computed as:

$$\nabla_{\mathbf{q}_p}\mathcal{L} = \nabla_{\mathbf{q}_p} V_{tot} + \nabla_{\mathbf{q}_p}(\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i) \tag{B.73}$$

where $\nabla_{\mathbf{q}_p} V = -\mathbf{I}_3$ and, being $\mathbf{q}_p = [\theta_p, \mathbf{p}_p] \in \mathbb{R}^3$ the term $\nabla_{\mathbf{q}_p}(\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i)$ is structured as follows:

$$\nabla_{\mathbf{q}_p}(\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i) = -\boldsymbol{\lambda}_i^T \begin{bmatrix} \mathbf{0}_{1\times 2} & 1 \\ \mathbf{I}_2 & \mathbf{R}_z(\theta_p + \pi/2)\mathbf{p}_{fi} \end{bmatrix} \tag{B.74}$$

B.2.2 Kinemato-static model

The derivation of the kinemato-static model requires the computation of the first derivative of each of model equation w.r.t. $\mathbf{q}_a, \mathbf{q}_e, \mathbf{q}_p$ and $\boldsymbol{\lambda}$.

Let us start with $\nabla_{\mathbf{q}_{ei}}\mathcal{L} = \mathbf{0}$: its first derivative w.r.t. $\mathbf{q}_{ei}$ is obtained by considering separately V_{ei}, V_{di} and $\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i$. The term $\nabla_{\mathbf{q}_{ei}} V_{ei}$ depends on $\mathbf{q}_{ei}, q_{ai}$ only, and it derivative w.r.t. $\mathbf{q}_{ei}$ is computed according to Eq. (B.27) as a three-diagonal matrix:

$$\nabla_{\mathbf{q}_{ei}}\left(\nabla_{\mathbf{q}_{ei}} V_{ei}\right) = \frac{K_b}{L_e} \begin{bmatrix} -1 & 2 & -1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & -1 \end{bmatrix} \tag{B.75}$$

Concering $\nabla_{q_{ai}}(\nabla_{\mathbf{q}_{ei}} V_{ei})$, this derivative depends on the specific actuators employed. In the case a prismatic actuator is used, $\nabla_{q_{ai}}(\nabla_{\mathbf{q}_{ei}} V_{ei}) = \mathbf{0}$ while, for rotative actuators:

$$\nabla_{q_{ai}}(\nabla_{\mathbf{q}_{ei}} V_{ei}) = \frac{\partial}{\partial \theta_0}(\nabla_{\mathbf{q}_{ei}} V_{ei}) \frac{\partial \theta_0}{\partial q_{ai}} \tag{B.76}$$

with the term $\partial/\partial\theta_0(\nabla_{\mathbf{q}_{ei}} V_{ei})$ obtainable from Eq. (B.75), and $\partial\theta_0/\partial q_{ai}$ depends on the specific actuator installation. All the other derivatives successive derivatives of $\nabla_{\mathbf{q}_{ei}} V_{ei}$ are null.

Let us consider now $\nabla_{\mathbf{q}_{ei}} V_{di}$, that depends on $\mathbf{q}_{ei}$ only. In particular, only θ_{ij} appears, and derivating Eq.(B.66) w.r.t. θ_{ij} results in:

$$\nabla_{\theta_{ij}}\left(\nabla_{\theta_{ij}} V_{di}\right) = \sum_{r=1}^{N_{elt}} \mathbf{f}_d^T \left(\nabla_{\theta_{ij}}\left(\nabla_{\theta_{ij}} \mathbf{p}_{ir}\right) + L_{ei} \nabla_{\theta_{ij}}\left(\nabla_{\theta_{ij}}\left(\mathbf{R}_{ir}\mathbf{e}_2^b\right)\right)\right) \tag{B.77}$$

with the terms $\nabla_{\theta_{ij}}\nabla_{\theta_{ij}}\mathbf{p}_{ir}$ and $\nabla_{\theta_{ij}}\nabla_{\theta_{ij}}\left(\mathbf{R}_{ir}\mathbf{e}_2^b\right)$ computed as follows:

$$\nabla_{\theta_{ij}}\left(\nabla_{\theta_{ij}} \mathbf{p}_{ir}\right) = L_{ei}\mathbf{R}_\pi \mathbf{e}_2^p;\ j \leqslant r; \quad \nabla_{\theta_{ij}}\left(\nabla_{\theta_{ij}} \mathbf{p}_{ir}\right) = \mathbf{0};\ j > r \tag{B.78}$$

$$\nabla_{\theta_{ij}}\left(\nabla_{\theta_{ij}}\left(\mathbf{R}_{ir}\mathbf{e}_3\right)\right) = \mathbf{R}_\pi \mathbf{e}_2^p;\ j = r; \quad \nabla_{\theta_{ij}}\left(\nabla_{\theta_{ij}}\left(\mathbf{R}_{ir}\mathbf{e}_2^p\right)\right) = \mathbf{0};\ j \neq r \tag{B.79}$$

with $\mathbf{R}_\pi = \mathbf{R}_z(\theta_{ij} + \pi)$. All the derivatives w.r.t. the other variables are null.

Then, let us consider $\nabla_{\mathbf{q}_{ei}} \boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i$ of Eq.(B.27), that depends on $\mathbf{q}_{ei}, \boldsymbol{\lambda}_i$. First, let us compute the derivative of Eq.(B.27) w.r.t. θ_{ij}, that is:

$$\nabla_{\theta_{ij}} \left(\nabla_{\theta_{ij}} (\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i) \right) = L_{ei} [\lambda_{2i}, \lambda_{3i}]^T \left(\mathbf{R}_\pi \mathbf{e}_2^p \right) \tag{B.80}$$

Let us now calculate $\nabla_{\boldsymbol{\lambda}_i} (\nabla_{\mathbf{q}_{ei}} \boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i)$. Please note that this term also correspond to $\nabla_{\mathbf{q}_{ei}} \boldsymbol{\Phi}_i$. Derivating c_{ij} w.r.t. $\boldsymbol{\lambda}_i$ results in:

$$\nabla_{\boldsymbol{\lambda}_i} c_{ij} = L_{ei} \begin{bmatrix} 0 \\ \mathbf{R}_{\pi/2} \mathbf{e}_2^p \end{bmatrix} \tag{B.81}$$

while, since d_{ij} is constant, its derivatives are null.

Finally, let us compute the derivatives of $\nabla_{\mathbf{q}_p} \mathcal{L}$, which depends on $\boldsymbol{\lambda}, \mathbf{q}_e$ and $\mathbf{q}_p$. In particular, $\nabla_{\mathbf{q}_p} V_{tot}$ is constant, and its first derivatives vanishes. Instead, let us consider $\nabla_{\mathbf{q}_p} \boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i$ of Eq.(B.31), where a differentiation w.r.t. $\boldsymbol{\lambda}$ results in:

$$\nabla_{\boldsymbol{\lambda}_i} \left(\nabla_{\mathbf{q}_p} \left(\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i \right) \right) = - \begin{bmatrix} \mathbf{0}_{1\times 2} & 1 \\ \mathbf{I}_2 & \mathbf{R}_z(\theta_p + \pi/2) \mathbf{p}_{fi} \end{bmatrix} \tag{B.82}$$

Please note that $\nabla_{\boldsymbol{\lambda}_i} \left(\nabla_{\mathbf{q}_p} \left(\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i \right) \right)$ also corresponds to $\nabla_{\mathbf{q}_p} \boldsymbol{\Phi}_i$. The derivative of Eq.(B.31) w.r.t. $\mathbf{q}_p$ is structured as follows:

$$\nabla_{\mathbf{q}_p} \left(\nabla_{\mathbf{q}_p} \left(\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i \right) \right) = \begin{bmatrix} \mathbf{0}_{2\times 3} \\ [\lambda_{2i}, \lambda_{3i}]^T \mathbf{R}_z(\theta_p + \pi) \mathbf{p}_{fi} & \mathbf{0}_{1\times 2} \end{bmatrix} \tag{B.83}$$

Appendix C

Derivatives of the kinemato-static model terms with respect to f

This appendix derives the derivatives of the kinemato-static model terms w.r.t. a parameter f useful for the *DCLI* computation of Chapter 6. In particular, the goal is to compute the following terms when a finite-differences approximation is used:

$$\frac{\partial \mathbf{H}}{\partial f} = \frac{\partial}{\partial f}\begin{bmatrix} \nabla_{\mathbf{q}_e}\left(\nabla_{\mathbf{q}_e}\mathcal{L}\right) & \nabla_{\mathbf{q}_p}\left(\nabla_{\mathbf{q}_e}\mathcal{L}\right) \\ \nabla_{\mathbf{q}_e}\left(\nabla_{\mathbf{q}_p}\mathcal{L}\right) & \nabla_{\mathbf{q}_p}\left(\nabla_{\mathbf{q}_p}\mathcal{L}\right) \end{bmatrix} \tag{C.1}$$

$$\frac{\partial \mathbf{\Lambda}}{\partial f} = \frac{\partial}{\partial f}\begin{bmatrix} \nabla_{\boldsymbol{\lambda}}\left(\nabla_{\mathbf{q}_e}\mathcal{L}\right) & \nabla_{\boldsymbol{\lambda}}\left(\nabla_{\mathbf{q}_p}\mathcal{L}\right) \end{bmatrix}^T \tag{C.2}$$

Sec. C.1 derives the expressions of these terms for the spatial case, while Sec. C.2 for the planar case.

C.1 Spatial Case

Let us first consider $\nabla_{\mathbf{q}_{ei}}\left(\nabla_{\mathbf{q}_{ei}}\mathcal{L}\right)$ which is structured as:

$$\nabla_{\mathbf{q}_{ei}}\left(\nabla_{\mathbf{q}_{ei}}\mathcal{L}\right) = \nabla_{\mathbf{q}_{ei}}\left(\nabla_{\mathbf{q}_{ei}}V_{ei}\right) + \left(\nabla_{\mathbf{q}_{ei}}\left(\nabla_{\mathbf{q}_{ei}}V_{di}\right)\right) + \left(\nabla_{\mathbf{q}_{ei}}\left(\nabla_{\mathbf{q}_{ei}}\left(\boldsymbol{\lambda}_i^T\mathbf{\Phi}\right)\right)\right) \tag{C.3}$$

The derivative of $\nabla_{\mathbf{q}_{ei}}\left(\nabla_{\mathbf{q}_{ei}}V_{ei}\right)$ w.r.t. f is obtained by derivating Eq. (B.35):

$$\frac{\partial}{\partial f}\left(\nabla_{\mathbf{q}_{ei}}\left(\nabla_{\mathbf{q}_{ei}}V_{ei}\right)\right) = \begin{bmatrix} \frac{\partial}{\partial f}\left(\mathbf{E}_{i1}\right) & \frac{\partial}{\partial f}\left(\mathbf{F}_{i2}^T\right) & \cdots & \mathbf{0} & \mathbf{0} \\ \frac{\partial}{\partial f}\left(\mathbf{F}_{i2}\right) & \frac{\partial}{\partial f}\left(\mathbf{E}_{i2}\right) & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \frac{\partial}{\partial f}\left(\mathbf{E}_{iN_{elt}-1}\right) & \frac{\partial}{\partial f}\left(\mathbf{F}_{iN_{elt}}^T\right) \\ \mathbf{0} & \mathbf{0} & \cdots & \frac{\partial}{\partial f}\left(\mathbf{F}_{iN_{elt}}\right) & \frac{\partial}{\partial f}\left(\mathbf{E}_{iN_{elt}}\right) \end{bmatrix} \tag{C.4}$$

in which:

$$\frac{\partial \mathbf{E}_{ij}}{\partial f} = \frac{\partial \mathbf{M}_{ij}}{\partial f} + \frac{\partial \mathbf{N}_{ij+1}}{\partial f} \tag{C.5}$$

$$\frac{\partial \mathbf{F}_{ij}}{\partial f} = -4 \sum_{k=1}^{3} \frac{K_k}{L_{ei}} \left((\mathbf{B}_k \frac{\partial \mathbf{h}_{ij}}{\partial f})(\mathbf{B}_k \mathbf{h}_{ij+1})^T + (\mathbf{B}_k \mathbf{h}_{ij})(\mathbf{B}_k \frac{\partial \mathbf{h}_{ij+}}{\partial f}))^T - \mathbf{B}_k \frac{1}{2} \frac{\partial A_{ijk}}{\partial f} \right) \tag{C.6}$$

$$\frac{\partial \mathbf{M}_{ij}}{\partial f} = -4 \sum_{k=1}^{3} \frac{K_k}{L_{ei}} \left((\mathbf{B}_k \mathbf{h}_{ij})(\mathbf{B}_k^T \frac{\partial \mathbf{h}_{ij}}{\partial f})^T + (\mathbf{B}_k \mathbf{h}_{ij})(\mathbf{B}_k^T \frac{\partial \mathbf{h}_{ij}}{\partial f})^T \right) \tag{C.7}$$

$$\frac{\partial \mathbf{N}_{ij}}{\partial f} = -4 \sum_{k=1}^{3} \frac{K_k}{L_{ei}} \left((\mathbf{B}_k \frac{\partial \mathbf{h}_{ij+1}}{\partial f})(\mathbf{B}_k \mathbf{h}_{ij+1})^T + (\mathbf{B}_k \mathbf{h}_{ij+1})(\mathbf{B}_k \frac{\partial \mathbf{h}_{ij+1}}{\partial f})^T \right) \tag{C.8}$$

$$\frac{\partial A_{ijk}}{\partial f} = -2 \left(\frac{\partial \mathbf{h}_{i(j+1)}}{\partial f}^T \mathbf{B}_k^T \mathbf{h}_{ij} + \mathbf{h}_{i(j+1)}^T \mathbf{B}_k^T \frac{\partial \mathbf{h}_{ij}}{\partial f} \right) \tag{C.9}$$

Let us consider the j-th term of $\nabla_{\mathbf{q}_{ei}} (\nabla_{\mathbf{q}_{ei}} V_{di})$. Since $\nabla_{\mathbf{h}_{ij}} \left(\nabla_{\mathbf{h}_{ij}} V_{di} \right)$ of Eq.(B.43) is constant, if derivative w.r.t. f is null:

$$\frac{\partial}{\partial f} \nabla_{\mathbf{h}_{ij}} \left(\nabla_{\mathbf{h}_{ij}} V_{di} \right) = \mathbf{0} \tag{C.10}$$

To compute the j-th term of $\nabla_{\mathbf{q}_{ei}} \left(\nabla_{\mathbf{q}_{ei}} \left(\boldsymbol{\lambda}_i^T \boldsymbol{\Phi} \right) \right)$, let us calculate the derivative w.r.t. f of Eq.(B.44):

$$\frac{\partial}{\partial f} \nabla_{\mathbf{h}_{ij}} \left(\nabla_{\mathbf{h}_{ij}} (\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i) \right) = \frac{\partial}{\partial f} \left(\nabla_{\mathbf{h}_i} \mathbf{c}_{ij} \right) + \frac{\partial}{\partial f} \left(\nabla_{\mathbf{h}_i} \mathbf{d}_{ij} \right) \tag{C.11}$$

in which:

$$\frac{\partial}{\partial f} \left(\nabla_{\mathbf{h}_i} \mathbf{c}_{ij} \right) = 2 L_{ei} \widetilde{\left(\frac{\partial \boldsymbol{\lambda}_{4:6i}}{\partial f} \right)} \tag{C.12}$$

and:

$$\begin{aligned} \frac{\partial}{\partial f} \left(\nabla_{\mathbf{h}_{ij}} \mathbf{d}_{ij} \right) = {} & \frac{\partial \lambda_{1i}}{\partial f} \left(\widetilde{\mathbf{d}_{p1}} - \widetilde{\mathbf{d}_{p2}} \right) + \frac{\partial \lambda_{2i}}{\partial f} \left(\widetilde{\mathbf{d}_{p1}} - \widetilde{\mathbf{d}_{p3}} \right) + \frac{\partial \lambda_{3i}}{\partial f} \left(\widetilde{\mathbf{d}_{p2}} - \widetilde{\mathbf{d}_{p3}} \right) + \\ & + \lambda_{1i} \left(\widetilde{\frac{\partial \mathbf{d}_{p1}}{\partial f}} - \widetilde{\frac{\partial \mathbf{d}_{p2}}{\partial f}} \right) + \lambda_{2i} \left(\widetilde{\frac{\partial \mathbf{d}_{p1}}{\partial f}} - \widetilde{\frac{\partial \mathbf{d}_{p3}}{\partial f}} \right) + \lambda_{3i} \left(\widetilde{\frac{\partial \mathbf{d}_{p2}}{\partial f}} - \widetilde{\frac{\partial \mathbf{d}_{p3}}{\partial f}} \right), \; j = N_{elt} \end{aligned} \tag{C.13}$$

In particular, the derivatives of the rotation matrix columns $\mathbf{d}_{pi}$ are computed by first calculating $\partial \mathbf{R}_p / \partial f$ and then extracting the columns:

$$\frac{\partial \mathbf{R}_p}{\partial f} = \sum_{i=1}^{n_c - 3} \frac{\partial \mathbf{R}_p}{\partial \alpha_i} \frac{\partial \alpha_i}{\partial f} \tag{C.14}$$

where $\alpha_i \; i = 1, \cdots, n_c - 3$ is the i-th parameter of the platform orientation. Since the term $\nabla_{\mathbf{q}_p} \left(\nabla_{\mathbf{q}_p} \left(\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i \right) \right)$ depends on the specific orientation parametrization, its derivative w.r.t. f is not reported here for brevity.

Let us know compute the derivative of $\nabla_{\boldsymbol{\lambda}_i} \left(\nabla_{\mathbf{q}_{ei}} \left(\boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i \right) \right)$ w.r.t. f. This term is structured as follows:

$$\frac{\partial}{\partial f} \nabla_{\boldsymbol{\lambda}_i} (\nabla_{\mathbf{q}_{ei}} \boldsymbol{\lambda}_i^T \boldsymbol{\Phi}_i) = \frac{\partial}{\partial f} \left(\nabla_{\boldsymbol{\lambda}_i} \mathbf{c}_{ij} \right) + \frac{\partial}{\partial f} \left(\nabla_{\boldsymbol{\lambda}_i} \mathbf{d}_{ij} \right) \tag{C.15}$$

where:

$$\nabla_{\boldsymbol{\lambda}_i} \mathbf{c}_{ij} = L_{ei} \begin{bmatrix} \mathbf{0}_{3 \times 4} \\ \frac{\partial \mathbf{D}_{3ij}}{\partial f} \end{bmatrix} \tag{C.16}$$

and:

$$\frac{\partial}{\partial f}\nabla_{\boldsymbol{\lambda}_i}\mathbf{d}_{ij} = \begin{bmatrix} \frac{\partial}{\partial f}\left(\mathbf{d}_{p1}^T\right)\mathbf{D}_{2ij} - \frac{\partial}{\partial f}\left(\mathbf{d}_{P2}^T\right)\mathbf{D}_{1ij} \\ \frac{\partial}{\partial f}\left(\mathbf{d}_{p1}^T\right)\mathbf{D}_{3ij} - \frac{\partial}{\partial f}\left(\mathbf{d}_{P3}^T\right)\mathbf{D}_{1ij} \\ \frac{\partial}{\partial f}\left(\mathbf{d}_{p2}^T\right)\mathbf{D}_{3ij} - \frac{\partial}{\partial f}\left(\mathbf{d}_{P3}^T\right)\mathbf{D}_{2ij} \\ \mathbf{0}_{3\times 4} \end{bmatrix} + \begin{bmatrix} \mathbf{d}_{p1}^T\frac{\partial}{\partial f}\left(\mathbf{D}_{2ij}\right) - \mathbf{d}_{P2}^T\frac{\partial}{\partial f}\left(\mathbf{D}_{1ij}\right) \\ \mathbf{d}_{p1}^T\frac{\partial}{\partial f}\left(\mathbf{D}_{3ij}\right) - \mathbf{d}_{P3}^T\frac{\partial}{\partial f}\left(\mathbf{D}_{1ij}\right) \\ \mathbf{d}_{p2}^T\frac{\partial}{\partial f}\left(\mathbf{D}_{3ij}\right) - \mathbf{d}_{P3}^T\frac{\partial}{\partial f}\left(\mathbf{D}_{2ij}\right) \\ \mathbf{0}_{3\times 4} \end{bmatrix},\ j = N_{elt} \tag{C.17}$$

where $\partial\mathbf{D}_{kij}/\partial f, k =, 1, 2, 3$ are computed according to Eq.(B.24) by using $\partial\mathbf{h}_{ij}/\partial f$.

Then, the derivative of $\nabla_{\boldsymbol{\lambda}_i}\left(\nabla_{\mathbf{q}_p}\left(\boldsymbol{\lambda}_i^T\boldsymbol{\Phi}_i\right)\right)$ w.r.t. f is calculated by derivating Eq.(B.49) w.r.t. f:

$$\frac{\partial}{\partial f}\left(\nabla_{\boldsymbol{\lambda}_i}\left(\nabla_{\mathbf{q}_p}\left(\boldsymbol{\lambda}_i^T\boldsymbol{\Phi}_i\right)\right)\right) = + \begin{bmatrix} \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 3} \\ \frac{\partial}{\partial f}\left(\nabla_{\boldsymbol{\lambda}_{4:6i}}\mathbf{m}_{p2}\right) & \mathbf{0}_{3\times 3} \end{bmatrix} \tag{C.18}$$

where:

$$\frac{\partial}{\partial f}\left(\nabla_{\boldsymbol{\lambda}_{4:6i}}\mathbf{m}_{p2}\right) = \left[\ \frac{\partial^2\mathbf{d}_{p1}^T}{\partial\boldsymbol{\alpha}\partial f}\mathbf{d}_{2ij} - \frac{\partial^2\mathbf{d}_{p2}^T}{\partial\boldsymbol{\alpha}\partial f}\mathbf{d}_{1ij},\ \ \frac{\partial^2\mathbf{d}_{p1}^T}{\partial\boldsymbol{\alpha}\partial f}\mathbf{d}_{3ij} - \frac{\partial^2\mathbf{d}_{p3}^T}{\partial\boldsymbol{\alpha}\partial f}\mathbf{d}_{1ij},\ \ \frac{\partial^2\mathbf{d}_{p1}^T}{\partial\boldsymbol{\alpha}\partial f}\mathbf{d}_{3ij} - \frac{\partial^2\mathbf{d}_{p3}^T}{\partial\boldsymbol{\alpha}\partial f}\mathbf{d}_{2ij}\right] +$$
$$+\left[\ \frac{\partial\mathbf{d}_{p1}^T}{\partial\boldsymbol{\alpha}}\frac{\partial\mathbf{d}_{2ij}}{\partial f} - \frac{\partial\mathbf{d}_{p2}^T}{\partial\boldsymbol{\alpha}}\frac{\partial\mathbf{d}_{1ij}}{\partial f},\ \ \frac{\partial\mathbf{d}_{p1}^T}{\partial\boldsymbol{\alpha}}\frac{\partial\mathbf{d}_{3ij}}{\partial f} - \frac{\partial\mathbf{d}_{p3}^T}{\partial\boldsymbol{\alpha}}\frac{\partial\mathbf{d}_{1ij}}{\partial f},\ \ \frac{\partial\mathbf{d}_{p1}^T}{\partial\boldsymbol{\alpha}}\frac{\partial\mathbf{d}_{3ij}}{\partial f} - \frac{\partial\mathbf{d}_{p3}^T}{\partial\boldsymbol{\alpha}}\frac{\partial\mathbf{d}_{2ij}}{\partial f}\right] \tag{C.19}$$

where the term $\partial^2\mathbf{d}_{pk}^T/\partial\boldsymbol{\alpha}\partial f$ depends on the specific platform orientation parametrization, and $\partial\mathbf{d}_{kij}/\partial f$ is obtained as follows:

$$\frac{\partial\mathbf{d}_{kij}}{\partial f} = \mathbf{D}_{kij}\frac{\partial\mathbf{h}_{ij}}{\partial f} \qquad k = 1, 2, 3 \tag{C.20}$$

C.2 Planar Case

Let us first consider $\nabla_{\mathbf{q}_{ei}}\left(\nabla_{\mathbf{q}_{ei}}\mathcal{L}\right)$. Since $\nabla_{\mathbf{q}_{ei}}\left(\nabla_{\mathbf{q}_{ei}}V_{ei}\right)$ of Eq.(B.75) is constant, its derivative w.r.t. f is null. Then, let us consider $\nabla_{\mathbf{q}_{ei}}\left(\nabla_{\mathbf{q}_{ei}}V_{di}\right)$ of Eq.(B.77). The derivative of its j-th components w.r.t. f is structured as follows:

$$\frac{\partial}{\partial f}\left(\nabla_{\theta_{ij}}\left(\nabla_{\theta_{ij}}V_{di}\right)\right) = \sum_{r=1}^{N_{elt}}\mathbf{f}_d^T\left(\frac{\partial}{\partial f}\left(\nabla_{\theta_{ij}}\left(\nabla_{\theta_{ij}}\mathbf{p}_{ir}\right)\right) + L_{ei}\frac{\partial}{\partial f}\left(\nabla_{\theta_{ij}}\left(\nabla_{\theta_{ij}}\left(\mathbf{R}_{ir}\mathbf{e}_2^b\right)\right)\right)\right) \tag{C.21}$$

where:

$$\frac{\partial}{\partial f}\left(\nabla_{\theta_{ij}}\left(\nabla_{\theta_{ij}}\mathbf{p}_{ir}\right)\right) = L_{ei}\mathbf{R}_{3\pi/2}\mathbf{e}_2^p\frac{\partial\theta_{ij}}{\partial f};\ j \leqslant r; \quad \frac{\partial}{\partial f}\left(\nabla_{\theta_{ij}}\left(\nabla_{\theta_{ij}}\mathbf{p}_{ir}\right)\right) = \mathbf{0};\ j > r \tag{C.22}$$

$$\frac{\partial}{\partial f}\left(\nabla_{\theta_{ij}}\left(\nabla_{\theta_{ij}}\left(\mathbf{R}_{ir}\mathbf{e}_3\right)\right)\right) = \mathbf{R}_{3\pi/2}\mathbf{e}_2^p\frac{\partial\theta_{ij}}{\partial f};\ j = r; \quad \frac{\partial}{\partial f}\left(\nabla_{\theta_{ij}}\left(\nabla_{\theta_{ij}}\right)\right)\left(\mathbf{R}_{ir}\mathbf{e}_2^p\right) = \mathbf{0};\ j \neq r \tag{C.23}$$

with $\mathbf{R}_{3\pi/2} = \mathbf{R}_z(\theta_{ij} + 3\pi/2)$.

The next considered term is $\nabla_{\theta_{ij}}\left(\nabla_{\theta_{ij}}(\boldsymbol{\lambda}_i^T\boldsymbol{\Phi}_i)\right)$, where its derivative w.r.t. f is structured as follows:

$$\nabla_{\theta_{ij}}\left(\nabla_{\theta_{ij}}(\boldsymbol{\lambda}_i^T\boldsymbol{\Phi}_i)\right) = L_{ei}\left[\frac{\partial\lambda_{2i}}{\partial f}, \frac{\partial\lambda_{3i}}{\partial f}\right]^T\left(\mathbf{R}_\pi\mathbf{e}_2^p\right) + L_{ei}[\lambda_{2i}, \lambda_{3i}]^T\left(\mathbf{R}_{3\pi/2}\mathbf{e}_2^p\frac{\partial\theta_{ij}}{\partial f}\right) \tag{C.24}$$

Let us now calculate the derivative of $\nabla_{\boldsymbol{\lambda}_i}(\nabla_{\mathbf{q}_{ei}}\boldsymbol{\lambda}_i^T\boldsymbol{\Phi}_i)$ w.r.t. f. Its expression is obtained by derivating Eq.(B.81) w.r.t. f:

$$\frac{\partial}{\partial f}\left(\nabla_{\boldsymbol{\lambda}_i}c_{ij}\right) = L_{ei}\begin{bmatrix} 0 \\ \mathbf{R}_\pi\frac{\partial\theta_{ij}}{\partial f}\mathbf{e}_2^p \end{bmatrix} \tag{C.25}$$

The derivative of $\nabla_{\boldsymbol{\lambda}_i}\left(\nabla_{\mathbf{q}_p}\left(\boldsymbol{\lambda}_i^T\boldsymbol{\Phi}_i\right)\right)$ w.r.t. f is computed as:

$$\frac{\partial}{\partial f}\left(\nabla_{\boldsymbol{\lambda}_i}\left(\nabla_{\mathbf{q}_p}\left(\boldsymbol{\lambda}_i^T\boldsymbol{\Phi}_i\right)\right)\right) = -\begin{bmatrix}\mathbf{0}_{1\times 2} & 0\\ \mathbf{0}_2 & \mathbf{R}_z(\theta_p+\pi)\mathbf{p}_{fi}\frac{\partial\theta_p}{\partial f}\end{bmatrix} \tag{C.26}$$

Finally, the derivative of the last required term $\nabla_{\mathbf{q}_p}\left(\nabla_{\mathbf{q}_p}\left(\boldsymbol{\lambda}_i^T\boldsymbol{\Phi}_i\right)\right)$ w.r.t. f is calculated as follows:

$$\frac{\partial}{\partial f}\left(\nabla_{\mathbf{q}_p}\left(\nabla_{\mathbf{q}_p}\left(\boldsymbol{\lambda}_i^T\boldsymbol{\Phi}_i\right)\right)\right) = \begin{bmatrix}\mathbf{0}_{2\times 3}\\ \left[\frac{\partial\lambda_{2i}}{\partial f},\frac{\partial\lambda_{3i}}{\partial f}\right]^T\mathbf{R}_z(\theta_p+\pi)\mathbf{p}_{fi}\end{bmatrix}+\begin{bmatrix}\mathbf{0}_{2\times 3}\\ [\lambda_{2i},\lambda_{3i}]^T\mathbf{R}_z(\theta_p+3\pi/2)\mathbf{p}_{fi}\frac{\partial\theta_p}{\partial f}\end{bmatrix} \tag{C.27}$$

Appendix D

Kantorovich's Constant Derivation

This appendix describes the detailed derivation of the Kantorovich constant γ, of Chapter 3. In this appendix no distributed loads are considered, and only passive revolute joints are included in the geometrico-static model derivation. These assumptions are in accordance with the necessary derivation of Chapter 3.

D.1 Computation of the First and Second IGSP Derivatives

The scope of this section is to compute the second derivatives of the geometrico-static model equations necessary for the computation of the Kantorovich's constant γ, here reported for clarity:

$$\gamma \geqslant \max_{\mathbf{y}\in\mathcal{B}} \left(\max_{h\in N_{eq}} \left(\sum_{i,j} \left| \frac{\partial^2 F_h(\mathbf{y})}{\partial y_i \partial y_j} \right| \right) \right) = \max_{\mathbf{y}\in\mathcal{B}} \left(\max \left(\mathbf{U}_\gamma(\mathbf{y}) \right) \right) \tag{D.1}$$

with

$$\mathbf{U}_\gamma(\mathbf{y}) = \left[\sum_{i,j} \left| \frac{\partial^2 F_1(\mathbf{y})}{\partial y_i \partial y_j} \right| , \cdots , \sum_{i,j} \left| \frac{\partial^2 F_h(\mathbf{y})}{\partial y_i \partial y_j} \right| , \cdots , \sum_{i,j} \left| \frac{\partial^2 F_{N_{eq}}(\mathbf{y})}{\partial y_i \partial y_j} \right| \right]^T \tag{D.2}$$

being the vector stacking the sum in absolute value of all the non-null second derivatives for each equation. For later convenience, we recall some important properties of absolute values ([145], Appendix A). Given two generic scalars $a, b \in \mathbb{R}$, we have:

$$|ab| = |a|\,|b| \tag{D.3}$$

$$|a + b| \leqslant |a| + |b| \tag{D.4}$$

In the next subsections, all the terms of $\mathbf{U}_\gamma$ are approximated in a worst-case scenario (considering the Kantorovich's certification) and their expressions are explicitly computed.

Let us first consider $\nabla_{\mathbf{q}_{ei}} (\nabla_{\mathbf{q}_{ei}} \mathcal{L})$: by considering Eq.(B.75), no distributed loads, and passive joints in Eq.B.80, the following expression of $\nabla_{\theta_{ij}} \nabla_{\theta_{ij}} \mathcal{L}$ is obtained:

$$\frac{\partial}{\partial \theta_{ij}} \left(\frac{\partial \mathcal{L}}{\partial \theta_{ij}} \right) = 2k_{eq} - L_{ei} \left(\lambda_{2i} \cos\theta_{ij} + \lambda_{3i} \sin\theta_{ij} \right) \tag{D.5}$$

where $k_{eq} = K_b / L_{ei}$. Eq. (D.5) depends on $\theta_{ij}, \lambda_{2i}, \lambda_{3i}$. By derivating Eq. (D.5) w.r.t. θ_{ij} we obtain:

$$\frac{\partial^2}{\partial \theta_{ij}^2} \left(\frac{\partial \mathcal{L}}{\partial \theta_{ij}} \right) = -L_{ei} \left(-\lambda_{2i} \sin\theta_{ij} + \lambda_{3i} \cos\theta_{ij} \right) \tag{D.6}$$

By employing properties of Eqs. (D.3), (D.4), the absolute value of Eq. (D.6) can be obtained as:

$$\left|\frac{\partial^2}{\partial\theta_{ij}^2}\left(\frac{\partial\mathcal{L}}{\partial\theta_{ij}}\right)\right| = L_{ei}\left|(-\lambda_{2i}\sin\theta_{ij} + \lambda_{3i}\cos\theta_{ij})\right| \leqslant \tag{D.7}$$

$$L_e\left(|\lambda_{2i}\sin\theta_{ij}| + |\lambda_{3i}\cos\theta_{ij}|\right) = L_{ei}\left(|\lambda_{2i}|\,|\sin\theta_{ij}| + |\lambda_{3i}|\,|\cos\theta_{ij}|\right) \tag{D.8}$$

Instead, differentiating Eq.(D.5) w.r.t. λ_{2i} and considering its absolute value results is:

$$\left|\frac{\partial^2}{\partial\lambda_{2i}\partial\theta_{ij}}\left(\frac{\partial\mathcal{L}}{\partial\theta_{ij}}\right)\right| = L_{ei}\left|\cos\theta_{ij}\right| \tag{D.9}$$

In the same fashion, differentiating Eq.(D.5) w.r.t. λ_{3i} and considering its absolute value:

$$\frac{\partial^2}{\partial\lambda_{3i}\partial\theta_{ij}}\left(\frac{\partial\mathcal{L}}{\partial\theta_{ij}}\right) = L_{ei}\left|\sin\theta_{ij}\right| \tag{D.10}$$

By considering Eqs. (D.8), (D.9), (D.10), the sum in absolute value of all the non-null derivatives of Eq. (D.5) can be expressed as:

$$\sum_{i,j}\left|\frac{\partial^2}{\partial y_i\partial y_j}\left(\frac{\partial\mathcal{L}}{\partial\theta_{ij}}\right)\right| = \left|\frac{\partial^2}{\partial\theta_{ij}^2}\left(\frac{\partial\mathcal{L}}{\partial\theta_{ij}}\right)\right| + 2\left|\frac{\partial^2}{\partial\lambda_{2i}\partial\theta_{ij}}\left(\frac{\partial\mathcal{L}}{\partial\theta_{ij}}\right)\right| + 2\left|\frac{\partial^2}{\partial\lambda_{3i}\partial\theta_{ij}}\left(\frac{\partial\mathcal{L}}{\partial\theta_{ij}}\right)\right| =$$
$$L_{ei}\left((|\lambda_{2i}| + 2)\,|\sin\theta_{ij}| + (|\lambda_{3i}| + 2)\,|\cos\theta_{ij}|\right) \tag{D.11}$$

The terms in Eq. (D.11) are not constant w.r.t. the robot variables, because it depends on $\theta_{ij}, \lambda_{2i}, \lambda_{3i}$.

Let us consider Eq. (B.83), where the only non-null term is simplified as follows:

$$\frac{\partial}{\partial\theta_p}\left(\frac{\partial\mathcal{L}}{\partial\theta_p}\right) = -\sum_{i=1}^{n}\left(-p_{pfix}\cos\theta_p + p_{pfiy}\sin\theta_p\right)\lambda_{2i} + \left(-p_{pfix}\sin\theta_p - p_{pfiy}\cos\theta_p\right)\lambda_{3i} \tag{D.12}$$

Eq. (D.12) depends on θ_p and $\boldsymbol{\lambda}$. A successive derivative of Eq. (D.12) w.r.t. θ_p results in:

$$\frac{\partial^2}{\partial\theta_p^2}\left(\frac{\partial\mathcal{L}}{\partial\theta_p}\right) = -\sum_{i=1}^{n}\left(+p_{fix}\sin\theta_p + p_{fiy}\cos\theta_p\right)\lambda_{2i} + \left(-p_{fix}\cos\theta_p + p_{fiy}\sin\theta_p\right)\lambda_{3i} \tag{D.13}$$

and its absolute value, by employing properties of Eqs. (D.3), (D.4), can be obtained as:

$$\left|\frac{\partial^2}{\partial\theta_p^2}\left(\frac{\partial\mathcal{L}}{\partial\theta_p}\right)\right| = \left|\sum_{i=1}^{n}\left(\left(-p_{fix}\sin\theta_p + p_{fiy}\cos\theta_p\right)\lambda_{2i} + \left(-p_{fix}\cos\theta_p + p_{fiy}\sin\theta_p\right)\lambda_{3i}\right)\right| \leqslant \tag{D.14}$$

$$\sum_{i=1}^{n}\left|\left(\left(-p_{fix}\sin\theta_p + p_{fiy}\cos\theta_p\right)\lambda_{2i} + \left(-p_{fix}\cos\theta_p + p_{fiy}\sin\theta_p\right)\lambda_{3i}\right)\right| \leqslant \tag{D.15}$$

$$\sum_{i=1}^{n}\left(\left|\left(-p_{fix}\sin\theta_p + p_{fiy}\cos\theta_p\right)\lambda_{2i}\right| + \left|\left(-p_{fix}\cos\theta_p + p_{fiy}\sin\theta_p\right)\lambda_{3i}\right|\right) = \tag{D.16}$$

$$\sum_{i=1}^{n}\left(\left|\left(-p_{fix}\sin\theta_p + p_{fiy}\cos\theta_p\right)\right|\left|\lambda_{2i}\right| + \left|\left(-p_{fix}\cos\theta_p + p_{fiy}\sin\theta_p\right)\right|\left|\lambda_{3i}\right|\right) \leqslant \tag{D.17}$$

$$\sum_{i=1}^{n}\left(\left(\left|p_{fix}\sin\theta_p\right| + \left|p_{fiy}\cos\theta_p\right|\right)\left|\lambda_{2i}\right| + \left(\left|p_{fix}\cos\theta_p\right| + \left|p_{fiy}\sin\theta_p\right|\right)\left|\lambda_{3i}\right|\right) = \tag{D.18}$$

$$\sum_{i=1}^{n}\left(\left(\left|p_{fix}\right|\left|\sin\theta_p\right| + \left|p_{fiy}\right|\left|\cos\theta_p\right|\right)\left|\lambda_{2i}\right| + \left(\left|p_{fix}\right|\left|\cos\theta_p\right| + \left|p_{fiy}\right|\left|\sin\theta_p\right|\right)\left|\lambda_{3i}\right|\right) \tag{D.19}$$

The derivative of Eq.(D.12) w.r.t. λ_{2i} can be obtained as follows:

$$\frac{\partial^2}{\partial\lambda_{2i}\partial\theta_p}\left(\frac{\partial\mathcal{L}}{\partial\theta_p}\right) = -\left(-p_{fix}\cos\theta_p + p_{fiy}\sin\theta_p\right)\lambda_{2i} \tag{D.20}$$

and, as before, its absolute value is computed:

$$\left|\frac{\partial^2}{\partial\lambda_{2i}\partial\theta_p}\left(\frac{\partial\mathcal{L}}{\partial\theta_p}\right)\right| = \left|\left(-p_{fix}\cos\theta_p + p_{fiy}\sin\theta_p\right)\lambda_{2i}\right| = \tag{D.21}$$

$$\left|\left(-p_{fix}\cos\theta_p + p_{fiy}\sin\theta_p\right)\right|\left|\lambda_{2i}\right| \leqslant \tag{D.22}$$

$$\left(\left|p_{fix}\cos\theta_p\right| + \left|p_{fiy}\sin\theta_p\right|\right)\left|\lambda_{2i}\right| = \tag{D.23}$$

$$\left(\left|p_{fix}\right|\left|\cos\theta_p\right| + \left|p_{fiy}\right|\left|\sin\theta_p\right|\right)\left|\lambda_{2i}\right| \tag{D.24}$$

In the same fashion, the derivative of Eq.(D.12) w.r.t. λ_{3i} can be obtained as;

$$\frac{\partial^2}{\partial\lambda_{2i}\partial\theta_p}\left(\frac{\partial\mathcal{L}}{\partial\theta_p}\right) = \left(-p_{fix}\sin\theta_p - p_{fiy}\cos\theta_p\right)\lambda_{3i} \tag{D.25}$$

and its absolute value:

$$\left|\frac{\partial^2}{\partial\lambda_{2i}\partial\theta_p}\left(\frac{\partial\mathcal{L}}{\partial\theta_p}\right)\right| = \left|\left(-p_{fix}\sin\theta_p - p_{fiy}\cos\theta_p\right)\lambda_{3i}\right| = \tag{D.26}$$

$$\left|\left(-p_{fix}\sin\theta_p - p_{fiy}\cos\theta_p\right)\right|\left|\lambda_{3i}\right| \leqslant \tag{D.27}$$

$$\left(\left|p_{fix}\right|\left|\sin\theta_p\right| + \left|p_{fiy}\right|\left|\cos\theta_p\right|\right)\left|\lambda_{3i}\right| \tag{D.28}$$

The sum in absolute value of all non-null derivatives of Eq.(D.12), by considering Eqs. (D.19),

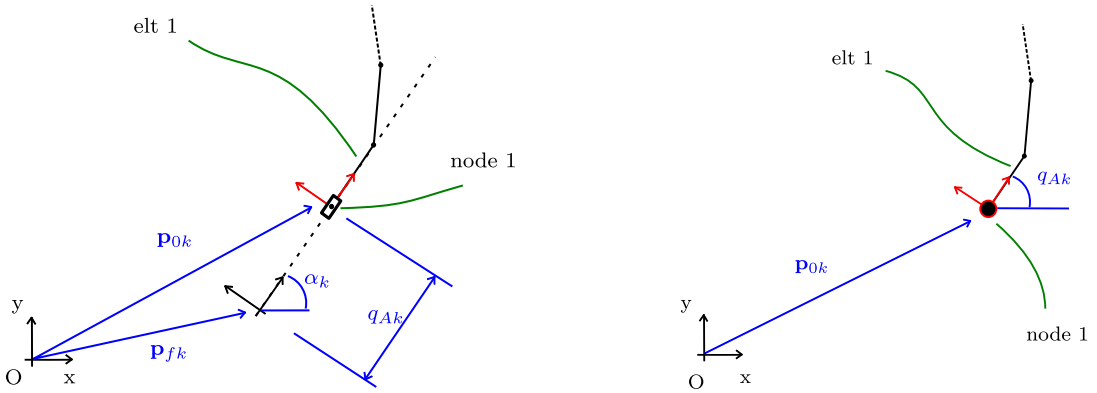


Figure D.1: (a) Prismatic and (b) revolute actuators.

(D.24), (D.28), is:

$$\sum_{i,j}\left|\frac{\partial^2}{\partial y_i \partial y_j}\left(\frac{\partial \mathcal{L}}{\partial \theta_p}\right)\right| = \left|\frac{\partial^2}{\partial \theta_p^2}\left(\frac{\partial \mathcal{L}}{\partial \theta_p}\right)\right| + 2\sum_{i=1}^{n}\left(\left|\frac{\partial^2}{\partial \lambda_{2i}\partial \theta_p}\left(\frac{\partial \mathcal{L}}{\partial \theta_p}\right)\right| + \left|\frac{\partial^2}{\partial \lambda_{3i}\partial \theta_p}\left(\frac{\partial \mathcal{L}}{\partial \theta_p}\right)\right|\right) = \tag{D.29}$$

$$\sum_{i=1}^{n}\left(|p_{fix}| + |p_{fiy}|\right)\left(|\cos\theta_p| + |\sin\theta_p|\right)\left(|\lambda_{2i}| + |\lambda_{3i}|\right) \tag{D.30}$$

Again, the sum of the non-null second derivative is not constant, and it depends on θ_p and all the multipliers $\boldsymbol{\lambda}$.

Then, let us consider the generic closure equation: in the case only revolute joints are considered:

$$\boldsymbol{\Phi}_i = \mathbf{p}_{0i} + L_{ei}\sum_{i=1}^{n}\begin{bmatrix}\cos\theta_{ij}\\ \sin\theta_{ij}\end{bmatrix} - (\mathbf{p}_P + \mathbf{R}_z(\theta_p)\mathbf{p}_{fi}) \tag{D.31}$$

which depends on all the $\theta_{ij}, q_{ai}, \mathbf{p}_P$ and θ_p. In this report two kinds of actuators are considered: revolute and prismatic. For the prismatic actuator (Fig.D.1a), the position of the connection point $\mathbf{p}_{0i}$ is defined by the actuated variable q_{ai}:

$$\mathbf{p}_{0i} = \mathbf{p}_{fk} + \mathbf{R}_z(\alpha_i)\begin{bmatrix}q_{ai}\\ 0\end{bmatrix} \tag{D.32}$$

where $\alpha_i k$ is a constant orientation of the prismatic joint, $\mathbf{p}_{fi}$ the fixed position of the origin of the prismatic joint. In this case, since Eq. (D.31) depends linearly on q_{ai}, the second derivative w.r.t. q_{ai} two times is null. In the case of a revolute motor (Fig.D.1b), $\theta_{i1} = q_{ai}$ and $\mathbf{p}_{0i}$ a constant vector indicating the position of the revolute joint. Taking the first derivative of Eq. (D.31) w.r.t. q_{ai} results in:

$$\frac{\partial}{\partial q_{ai}}\boldsymbol{\Phi}_i = L_e\begin{bmatrix}-\sin q_{ai}\\ \cos q_{ai}\end{bmatrix} \tag{D.33}$$

and the absolute value of the second derivatives w.r.t. q_{ai} two times is:

$$\left|\frac{\partial^2}{\partial q_{ai}^2}\boldsymbol{\Phi}_i\right| = L_e\begin{bmatrix}|\cos q_{ai}|\\ |\sin q_{ai}|\end{bmatrix} \tag{D.34}$$

Then, consider the dependence of the closure loop equations w.r.t. intermediate angles θ_{ij}. Taking the first derivative of Eq. (D.31) w.r.t. θ_{ij}:

$$\frac{\partial}{\partial \theta_{ij}}\boldsymbol{\Phi}_i = L_e\begin{bmatrix}-\sin\theta_{ij}\\ \cos\theta_{ij}\end{bmatrix} \tag{D.35}$$

and the absolute value of the second derivative w.r.t. θ_{ij} two times:

$$\left|\frac{\partial^2}{\partial\theta_{ij}^2}\mathbf{\Phi}_i\right| = L_e\begin{bmatrix}|\cos\theta_{ij}|\\|\sin\theta_{ij}|\end{bmatrix} \tag{D.36}$$

Since Eq. (D.31) depends linearly on $\mathbf{p}_P$, its second derivatives are null. Considering the dependence on θ_p, by taking the first derivative of Eq. (D.31) w.r.t. θ_p, we obtain:

$$\frac{\partial}{\partial\theta_p}\mathbf{\Phi}_i = -\begin{bmatrix}-p_{fix}\sin\theta_p - p_{fiy}\cos\theta_p\\ p_{fix}\cos\theta_p - p_{fiy}\sin\theta_p\end{bmatrix} \tag{D.37}$$

and the second derivative w.r.t. θ_p two times:

$$\frac{\partial^2}{\partial\theta_p^2}\mathbf{\Phi}_i = -\begin{bmatrix}-p_{fix}\cos\theta_p + p_{fiy}\sin\theta_p\\ -p_{fix}\sin\theta_p - p_{fiy}\cos\theta_p\end{bmatrix} \tag{D.38}$$

The absolute value of each Eq. (D.38) is:

$$\left|\frac{\partial^2}{\partial\theta_p^2}\mathbf{\Phi}_i\right| = \begin{bmatrix}|-p_{fix}\cos\theta_p + p_{fiy}\sin\theta_p|\\|-p_{fix}\sin\theta_p - p_{fiy}\cos\theta_p|\end{bmatrix} \leqslant \begin{bmatrix}|p_{fix}\cos\theta_p| + |p_{fiy}\sin\theta_p|\\|p_{fix}\sin\theta_p| + |p_{fiy}\cos\theta_p|\end{bmatrix} = \begin{bmatrix}|p_{fix}|\,|\cos\theta_p| + |p_{fiy}|\,|\sin\theta_p|\\|p_{fix}|\,|\sin\theta_p| + |p_{fiy}|\,|\cos\theta_p|\end{bmatrix} \tag{D.39}$$

Proceeding as before, the sum of all absolute values of the second derivatives of Eq.(D.31) is:

$$\sum_{i,j}\left|\frac{\partial^2}{\partial y_i\partial y_j}\left(\mathbf{\Phi}_i\right)\right| = \left|\frac{\partial^2}{\partial\theta_p^2}\mathbf{\Phi}_i\right| + \left|\frac{\partial^2}{\partial q_{ai}^2}\mathbf{\Phi}_i\right| + \sum_{i=2}^{N}\left|\frac{\partial^2}{\partial\theta_{ij}^2}\mathbf{\Phi}_i\right| \tag{D.40}$$

In the case of a prismatic actuator the term w.r.t. q_{ai} is null:

$$\sum_{i,j}\left|\frac{\partial^2}{\partial y_i\partial y_j}\left(\mathbf{\Phi}_i\right)\right| = \left|\frac{\partial^2}{\partial\theta_p^2}\mathbf{\Phi}_i\right| + \sum_{i=2}^{N}\left|\frac{\partial^2}{\partial\theta_{ij}^2}\mathbf{\Phi}_i\right| = \begin{bmatrix}|p_{fix}|\,|\cos\theta_p| + |p_{fiy}|\,|\sin\theta_p|\\|p_{fix}|\,|\sin\theta_p| + |p_{fiy}|\,|\cos\theta_p|\end{bmatrix} + \sum_{i=2}^{N} L_e\begin{bmatrix}|\cos\theta_{ij}|\\|\sin\theta_{ij}|\end{bmatrix} \tag{D.41}$$

Whereas, in the case of a revolute actuator:

$$\sum_{i,j}\left|\frac{\partial^2}{\partial y_i\partial y_j}\left(\mathbf{\Phi}_i\right)\right| = \left|\frac{\partial^2}{\partial\theta_p^2}\mathbf{\Phi}_i\right| + \left|\frac{\partial^2}{\partial q_{ai}^2}\mathbf{\Phi}_i\right| + \sum_{i=2}^{N}\left|\frac{\partial^2}{\partial\theta_{ij}^2}\mathbf{\Phi}_i\right| = \tag{D.42}$$

$$\begin{bmatrix}|p_{fix}|\,|\cos\theta_p| + |p_{fiy}|\,|\sin\theta_p|\\|p_{fix}|\,|\sin\theta_p| + |p_{fiy}|\,|\cos\theta_p|\end{bmatrix} + L_e\left(\begin{bmatrix}|\cos q_{ai}|\\|\sin q_{ai}|\end{bmatrix} + \sum_{i=2}^{N}\begin{bmatrix}|\cos\theta_{ij}|\\|\sin\theta_{ij}|\end{bmatrix}\right) \tag{D.43}$$

The sum of the second derivative depends on θ_p all the θ_{ij} and, in the case of a revolute motor, also on q_{ai}.

D.2 Kantorovich's Constants Computation

In this Section, we demonstrate how to obtain the Kantorovich constant λ. For the computation of λ accordingly to Eq.(3.5), we have to identify where:

$$\max_{h\in[1,N_{eq}]}\left(\sum_{i,j}\left|\frac{\partial^2 F_h(\mathbf{y})}{\partial y_i\partial y_j}\right|\right) = \max\left(\mathbf{U}_\gamma(\mathbf{y})\right) \tag{D.44}$$

assumes the maximum value inside a ball $\mathcal{B}$ centered in $\bar{\mathbf{y}}$ of radius 2δ. We decide to employ infinite norm, the vector $\mathbf{y}$ can be expressed as:

$$\mathbf{y} = \bar{\mathbf{y}} \pm \mathbf{\Delta} \tag{D.45}$$

where the superscript $\bar{(.)}$ indicate the values at the center of the ball. Equations (D.45) implies that:

$$\|\mathbf{y} - \bar{\mathbf{y}}\|_\infty = \|\Delta\|_\infty \leqslant 2\delta \tag{D.46}$$

Because of Eq. (D.46), each term of $\mathbf{\Delta}$ can assume values $\in [0, 2\delta]$ in the case of infinite norm is employed.

In order to obtain the value of λ, we have to determine which term of $\mathbf{U}_\gamma$ assume the highest value inside the ball. To do that, we consider the case of revolute actuator, since the prismatic-actuator case is included. The set of the non null terms of $\mathbf{U}_\gamma$ can be written as:

$$\mathbf{U}_\gamma(\mathbf{y}) = \begin{cases} L_e\left(\left(|\lambda_{2i}| + 2\right)|\sin\theta_{ij}| + \left(+|\lambda_{3i}| + 2\right)|\cos\theta_{ij}|\right) \ \forall i = 2, \cdots, N \ , \ \forall k = 1, \cdots n \\ |p_{fix}|\,|\cos\theta_p| + |p_{fiy}|\,|\sin\theta_p| + L_e\left(|\cos q_{ai}| + \sum_{i=2}^{N}|\cos\theta_{ij}|\right) \ \forall k = 1, \cdots, n \\ |p_{fix}|\,|\sin\theta_p| + |p_{fiy}|\,|\cos\theta_p| + L_e\left(|\sin q_{ai}| + \sum_{i=2}^{N}|\sin\theta_{ij}|\right) \ \forall k = 1, \cdots, n \\ \sum_{i=1}^{n} + \left(|p_{fix}| + |p_{fiy}|\right)\left(|\cos\theta_p| + |\sin\theta_p|\right)\left(|\lambda_{2i}| + |\lambda_{3i}|\right) \end{cases} \tag{D.47}$$

We note that $\mathbf{U}_\gamma$ of Eq. (D.47) depends on some components of $\mathbf{y}$. Therefore, we have to determine for which value of $\mathbf{y} \in \mathcal{B}(\mathbf{y}_0, 2\delta)$ the highest value in $\mathbf{U}_\gamma$ is obtained. It is convenient to separate the value at the center of the ball $\bar{\mathbf{y}}$ from the increment $\mathbf{\Delta}$ as follows. For the multipliers:

$$|\lambda_{2i}| = |\bar{\lambda}_{1k} \pm \Delta_{\lambda_{2i}}| \leqslant |\bar{\lambda}_{1k}| + |\Delta_{\lambda_{2i}}| \tag{D.48}$$

$$|\lambda_{3i}| = |\bar{\lambda}_{2k} \pm \Delta_{\lambda_{3i}}| \leqslant |\bar{\lambda}_{2k}| + |\Delta_{\lambda_{3i}}| \tag{D.49}$$

For the angles $(q_{ai}, \theta_{ij}, \theta_p)$, since they are embedded in trigonometric functions, we employ basic trigonometric expressions. As an example, for the angle θ_{ij}:

$$|\sin\theta_{ij}| = |\sin\left(\bar{\theta}_{ik} \pm \Delta_{\theta_{ij}}\right)| = |\sin\bar{\theta}_{ik}\cos\Delta_{\theta_{ij}} \pm \cos\bar{\theta}_{ik}\sin\Delta_{\theta_{ij}}| \leqslant \tag{D.50}$$

$$|\sin\bar{\theta}_{ik}||\cos\Delta_{\theta_{ij}}| + |\cos\bar{\theta}_{ik}||\sin\Delta_{\theta_{ij}}| \tag{D.51}$$

$$|\cos\theta_{ij}| = |\cos\left(\bar{\theta}_{ik} \pm \Delta_{\theta_{ij}}\right)| = |\cos\bar{\theta}_{ik}\cos\Delta_{\theta_{ij}} \mp \sin\bar{\theta}_{ik}\sin\Delta_{\theta_{ij}}| \leqslant \tag{D.52}$$

$$|\cos\bar{\theta}_{ik}||\cos\Delta_{\theta_{ij}}| + |\sin\bar{\theta}_{ik}||\sin\Delta_{\theta_{ij}}| \tag{D.53}$$

and the same strategy is used for q_{ai}, θ_p Then, we study each equation of $\mathbf{U}_\gamma$ separately. Considering the first of Eq. (D.47), by substituting Eqs. (D.48),(D.49),(D.51),(D.53), we obtain:

$$L_e\left(|\bar{\lambda_{2i}}| + |\Delta_{\lambda_{2i}}| + 2\right)\left(|\sin\bar{\theta_{ij}}||\cos\Delta_{\theta_{ij}}| + |\cos\bar{\theta_{ij}}||\sin\Delta_{\theta_{ij}}|\right) + L_e\left(|\bar{\lambda_{2i}}| + |\Delta_{\lambda_{3i}}| + 2\right)\left(|\cos\bar{\theta_{ij}}||\cos\Delta_{\theta_{ij}}| + |\sin\bar{\theta_{ij}}||\sin\Delta_{\theta_{ij}}|\right) \tag{D.54}$$

At this point, we cannot define the exact increments $\Delta_{\theta_{ij}}, \Delta_{\lambda_{2i}}, \Delta_{\lambda_{3i}}$ inside the ball for which the function is maximum. This is caused by the presence of trigonometric functions. The maximum may be found by solving a constrained optimization problem, to identify the maximum inside the ball, but it is time consuming and not adequate for being iterated several times (as in the case of a workspace computation algorithm). Alternative, an upper bound that ensure safety in the quantification of the maximum value is obtained as follows. A majorizer for the maximum value that Eq. (D.54) is given by:

$$L_e\left(+|\bar{\lambda_{2i}}| + 2\delta + 2\right)\left(|\sin\bar{\theta}_{ik}| + 2\delta|\cos\bar{\theta}_{ik}|\right) + L_e\left(+|\bar{\lambda_{2i}}| + 2\delta + 2\right)\left(|\cos\bar{\theta}_{ik}| + 2\delta|\sin\bar{\theta}_{ik}|\right) \tag{D.55}$$

where we selected:

$$\Delta_{\lambda_{2i}} = 2\delta \tag{D.56}$$
$$\Delta_{\lambda_{3i}} = 2\delta \tag{D.57}$$
$$\cos\Delta_{\theta_k} = 1 \tag{D.58}$$
$$\sin\Delta_{\theta_k} = 2\delta \tag{D.59}$$

Proceeding in the same way for the second of Eq. (D.47) we obtain:

$$\begin{aligned}&|p_{fix}|\left(|\cos\bar{\theta_p}||\cos\Delta_{\theta_p}| + |\sin\bar{\theta_p}||\sin\Delta_{\theta_p}|\right) + |p_{fiy}|\left(|\sin\bar{\theta_p}||\cos\Delta_{\theta_p}| + |\cos\bar{\theta_p}||\sin\Delta_{\theta_p}|\right)\\ &+ L_e\left(|\cos\bar{q_{ai}}||\cos\Delta_{q_{ai}}| + |\sin\bar{q_{ai}}||\sin\Delta_{q_{ai}}| + \sum_{i=2}^{N}|\cos\bar{\theta_{ij}}||\cos\Delta_{\theta_{ij}}| + |\sin\bar{\theta_{ij}}||\sin\Delta_{\theta_{ij}}|\right)\end{aligned} \tag{D.60}$$

As before, a majorizer for the maximum value of .Eq. (D.60) is:

$$\begin{aligned}&|p_{fix}|\left(|\cos\bar{\theta_p}| + 2\delta|\sin\bar{\theta_p}|+\right) + |p_{fiy}|\left(|\sin\bar{\theta_p}| + 2\delta|\cos\bar{\theta_p}|\right)\\ &+ L_e\left(|\cos\bar{q_{ai}}| + 2\delta|\sin\bar{q_{ai}}| + \sum_{i=2}^{N}|\cos\bar{\theta_{ij}}| + 2\delta|\sin\bar{\theta_{ij}}|\right)\end{aligned} \tag{D.61}$$

where we selected:

$$\cos\Delta_{q_{ai}} = 1 \tag{D.62}$$
$$\sin\Delta_{q_{ai}} = 2\delta \tag{D.63}$$
$$\cos\Delta_{\theta_k} = 1 \tag{D.64}$$
$$\sin\Delta_{\theta_k} = 2\delta \tag{D.65}$$
$$\cos\Delta_{\theta_p} = 1 \tag{D.66}$$
$$\sin\Delta_{\theta_p} = 2\delta \tag{D.67}$$

With the same procedure, the majorizer for the maximum of the third of Eq. (D.47) is given by:

$$\begin{aligned}&|p_{fix}|\left(|\sin\bar{\theta_p}| + 2\delta|\cos\bar{\theta_p}|+\right) + |p_{fiy}|\left(|\cos\bar{\theta_p}| + 2\delta|\sin\bar{\theta_p}|\right)\\ &+ L_e\left(|\sin\bar{q_{ai}}| + 2\delta|\cos\bar{q_{ai}}| + \sum_{i=2}^{N}|\sin\bar{\theta_{ij}}| + 2\delta|\cos\bar{\theta_{ij}}|\right)\end{aligned} \tag{D.68}$$

And finally, for the last of Eq. (D.47), the majorizer can be written as:

$$\sum_{i=1}^{n}\left(|p_{fix}| + |p_{fiy}|\right)\left(|\cos\bar{\theta_p}| + \sin\bar{\theta_p}|\right)(2\delta+1)\left(|\bar{\lambda_{2i}}| + |\bar{\lambda_{12}}| + 4\delta\right) \tag{D.69}$$

Then, the value of the constant λ can be obtained by evaluating the maximum between Eq.(D.55) $\forall i,k$, Eqs.(D.61),(D.68) $\forall k$ and Eq.(D.69).

Appendix E

Concentric tube robot geometrico-static modelling

In order to derive a geometrico-static model formulation analogous to the one presented in Chapter 6, this appendix derives the geometrico-static model of a concentric tube robot (*CTR*) based on energetic considerations.

E.1 Tubes energy

Let us consider a *CTR* made by n concentric tubes (Fig. E.1a). A fixed frame $\mathcal{F}_0$ is attached to the robot base, and the coordinate s is used to parametrize the robot centerline. We use i as the index representing the i-th tube, numbered from the innermost to the outermost. The length of the i-th tube, measured from the arc length $s = 0$, is L_i. The tubes are actuated at the coordinate $s = -\beta_i$ in translation and rotation: we name θ_{0i} the rotation of the tube's base, and β_i is called transmission length.

As done in [102], a frame $\mathcal{F}_b$ is attached to the robot centerline at each s, and $\mathbf{p}(s)$ defines the position of $\mathcal{F}_b$ with respect to (w.r.t.) $\mathcal{F}_0$. Then, the orientation of $\mathcal{F}_b$ is described by $\mathbf{R}_b$, that is a rotation matrix obtained by sliding the base frame orientation matrix along the robot centerline without any rotation about the local z axis, assuming the z axis to be aligned with the centerline tangent vector. Since the tubes are concentric, the position of the i-th tube is $\mathbf{p}_i(s) = \mathbf{p}(s)$. However, the concentric tube arrangement let each tube to be free to twist around the local z axis of an angle $\theta_i(s)$ (see Fig. E.1b). Thus, we can recover the i-th tube orientation w.r.t. $\mathcal{F}_0$ as $\mathbf{R}_i(s) = \mathbf{R}_z(\theta_i(s))\mathbf{R}_b(s)$, where $\mathbf{R}_z$ is an elementar rotation around z. Then, we consider $\mathbf{u}_b(s)$, which represents the angular rate of change of $\mathbf{R}_b(s)$ w.r.t. s. We can recover the i-th tube curvature $\mathbf{u}_i(s)$ as [102]:

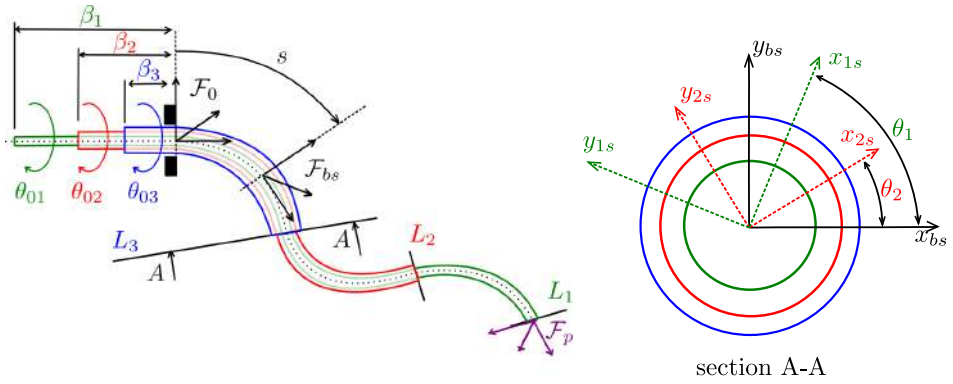
$$\mathbf{u}_i(s) = \mathbf{R}_z^T(\theta_i(s))\mathbf{u}_b(s) + \theta_i'(s)\mathbf{e}_z \tag{E.1}$$

where $(.)' = d/ds$, and $\mathbf{e}_z = [0; 0; 1]$. In the following, the variables explicit dependence from s is dropped for brevity sake.

In order to derive the *CTR* geometrico-static model, let us consider the deformation energy of the tubes V_t. By assuming linear elasticity, and by neglecting shear and extensibility on the tubes, we can compute V_t as [74]:

$$V_t = \sum_{i=1}^{n} V_i; \quad V_i = \frac{1}{2}\int_{L_{i-1}}^{L_i} \left(\sum_{j=1}^{m} \left(\mathbf{u}_j - \mathbf{u}_j^*\right)^T \mathbf{K}_t \left(\mathbf{u}_j - \mathbf{u}_j^*\right)^T \right) ds \tag{E.2}$$

where $L_0 = 0$, $m = n + 1 - i$, $\mathbf{u}_i^*$ is the initial tube precurvature, $\mathbf{K}_i = \text{diag}(k_{bi}, k_{bi}, k_{ti})$, is the local stiffness matrix, k_{bi} the flexural stiffness, and k_{ti} the torsional stiffness of the



(a) *CTR* representation (b) Tubes twist angles

Figure E.1: Representation of a *CTR*: relevant configuration variables are displayed in (a), and the tubes twist angles are illustrated in (b).

i-th tube. By introducing Eq. (E.1), into Eq. (E.2) we have:

$$V_i = \frac{1}{2}\int_{L_{i-1}}^{L_i} \sum_{j=1}^{m} \left(\mathbf{R}_z^T(\theta_j)\mathbf{u}_b + \theta_j'\mathbf{e}_z - \mathbf{u}_j^*\right)^T \mathbf{K}_j \left(\mathbf{R}_z^T(\theta_j)\mathbf{u}_b + \theta_j'\mathbf{e}_z - \mathbf{u}_j^*\right)^T ds \tag{E.3}$$

Thus, the tubes energy depends on $\mathbf{u}_b$, all the θ_i of the tubes and the actuated lengths L_i.

E.2 *CTR* energy

In order to deduce *CTR* energy, some configuration variables classification is necessary. First, we introduce the vector $\mathbf{q}_a \in \mathbb{R}^{2n}$ to collect the actuated variables (base rotation θ_{i0} and length L_i of each tube, two motors for each tube). Then, we introduce $\mathbf{q}_p = [\mathbf{p}_p, \boldsymbol{\alpha}_p] \in \mathbb{R}^6$ to represent the *CTR* tip pose: $\mathbf{p}_p \in \mathbb{R}^3$ is the tip position, and $\boldsymbol{\alpha}_p \in \mathbb{R}^3$ are three orientation angles of the tip that define $\mathbf{R}_p = \mathbf{R}_p(\boldsymbol{\alpha}_p)$. For later convenience, we introduce $\mathbf{q}_c$ the set of controlled variables: we assume to have the same number of controlled and actuated variables, and thus $\mathbf{q}_c \in \mathbb{R}^{2n}$. In general, $\mathbf{q}_c$ is a subset of $\mathbf{q}_p$, and the remaining tip variables are stacked into $\mathbf{q}_u \in \mathbb{R}^{6-2n}$. To evaluate the *CTR* energy, we consider the energy associated with a tip force $\mathbf{f}_p$, constant w.r.t. $\mathcal{F}_0$, that is[1].

$$V_{tip} = -\mathbf{f}_p^T \mathbf{p}_p \tag{E.4}$$

and the total *CTR* potential energy is obtained as:

$$V_{tot} = V_{tip} + V_t \tag{E.5}$$

Although we introduced $\mathbf{q}_p$ to represent the *CTR* tip pose, we can also recover the *CTR* tip pose as inner tube distal section pose:

$$\mathbf{g}_{tip} = \mathbf{g}_0 + \int_0^{L_1} \mathbf{g}\tilde{\boldsymbol{\xi}} ds; \quad \tilde{\boldsymbol{\xi}} = \begin{bmatrix} \hat{\mathbf{u}}_1 & \mathbf{e}_z \\ \mathbf{0} & 0 \end{bmatrix} \tag{E.6}$$

with $\hat{\mathbf{u}}_1 \in so(3)$ is the skew-symmetric matrix obtained from $\mathbf{u}_1$, and $\mathbf{g}, \mathbf{g}_0, \mathbf{g}_{tip} \in SE(3)$ are the matrices expressing the pose the first tube at s, the pose of the base frame, and the

[1]Generic concentrated loads or distributed load are neglected for simplicity, but their inclusion is possible. Moments which are not conservative are not considered.

tip pose, respectively. Thus, since $\mathbf{q}_p$ must represent the same pose as the one obtained in Eq. (E.6), the following constraint is introduced:

$$\mathbf{\Phi} = \begin{cases} \mathbf{p}_p - \mathbf{p}_{tip} = \mathbf{0} \\ \check{(}\mathbf{R}_p^T\mathbf{R}_{tip} - \mathbf{R}_{tip}^T\mathbf{R}_p\check{)} = \mathbf{0} \end{cases}, \ \mathbf{\Phi} \in \mathbb{R}^6 \tag{E.7}$$

where $\mathbf{p}_{tip}, \mathbf{R}_{tip}$ are the tip position and orientation obtained from Eq.(E.6), and the operator $\check{(.)}$ extracts the three independent components from its argument.

E.3 Geometrico-static model

CTRs equilibrium configurations are associated with critical points of the robot energy V_{tot} [74]. The function V_{tot} depends on $\mathbf{q}_a, \mathbf{q}_p$, and the continuous functions $\mathbf{u}_b$ and θ_i. Moreover, constraints $\mathbf{\Phi}$ must be enforced. Discretization techniques offers a straightforward way to numerically identify critical points of V_{tot} [106], and a finite set of discretization coordinates $\mathbf{q}_e \in \mathbb{R}^m$ is introduced to parametrize the tubes elastic deformations [106]. For convenience, we define $\mathbf{q}_d = [\mathbf{q}_u, \mathbf{q}_e] \in \mathbb{R}^{m+6-2n}$ to collect the uncontrolled variables after the discretization of the tubes, and $\mathbf{x} = [\mathbf{q}_d, \mathbf{q}_c] \in \mathbb{R}^{m+6}$ the non actuated variables. Please note that this way of parametrizing the *CTR* is true regardeless of the discretization technique employed.

After the discretization process, $V_{tot} = V_{tot}(\mathbf{q}_a, \mathbf{x})$ and $\mathbf{\Phi} = \mathbf{\Phi}(\mathbf{q}_a, \mathbf{x})$. Due to the presence of constraints, critical points of V_{tot} are characterized by Lagrange conditions [145]: assuming $\nabla_{\mathbf{x}}\mathbf{\Phi}$ is full row rank, $\mathbf{x}$ is a critical point of V_{tot} if there exists a vector of Lagrange multipliers $\boldsymbol{\lambda} \in \mathbb{R}^6$ such as:

$$\begin{cases} \nabla_{\mathbf{x}} V_{tot} + \nabla_{\mathbf{x}}\mathbf{\Phi}^T\boldsymbol{\lambda} = \mathbf{0} \\ \mathbf{\Phi} = \mathbf{0} \end{cases} \tag{E.8}$$

Equation (E.8) represents the geometrico-static model of a *CTR*, which is an undetermined system of $m + 6 + 6$ system of equations in $2n + m + 6 + 6$ unknowns. As in [56], forward and inverse problems are stated in a unified formulation:

$$\mathbf{F} = \begin{cases} \nabla_{\mathbf{x}} V_{tot} + \nabla_{\mathbf{x}}\mathbf{\Phi}_p^T\boldsymbol{\lambda} = \mathbf{0} \\ \mathbf{\Phi} = \mathbf{0} \\ \mathbf{e} = \mathbf{0} \end{cases} \tag{E.9}$$

where $\mathbf{e} = \mathbf{q}_a - \mathbf{q}_a^d$ for the forward problem, $\mathbf{e} = \mathbf{q}_c - \mathbf{q}_c^d$ for the inverse problem, and the superscript $(.)^d$ indicates a desired value. Eqs. (E.9) is a system of $2n + m + 6 + 6$ non-linear equations that can be solved with appropriate root-finding techniques [145] (e.g the Newton-Raphson method). It should be noted that Eqs. (E.9) are formulated in the same fashion as Eq. (2.23) of Chapter 2

Author's Publications

The following publications are author's publications related on the topics of this thesis, and realized during the time of this Ph.D:

International Journals:

- F. Zaccaria, E. Ida', and S. Briot, "A Boundary Computation Algorithm for the Workspace Evaluation of Continuum Parallel Robots," Journal of Mechanisms and Robotics, vol. 16, no. 4, p. 041 010, 2023.

- F. Zaccaria, E. Idá, S. Briot, and M. Carricato, "Workspace computation of planar continuum parallel robots," IEEE Robotics and Automation Letters, vol. 7, no. 2, pp. 2700–2707, 2022

- F. Zaccaria, E. Idà, S. Briot. "Design and Experimental Equilibrium Stability Assessment of a RFRFR Continuum Parallel Robot", Mechatronics, 95 (2023): 103064, Elsevier

International Conferences and Congress:

- A. Gotelli, F. Zaccaria, O. Kermorgant, and S. Briot, "A gazebo simulator for continuum parallel robots," in Altuzarra, O., Kecskeméthy, A. (eds) Advances in Robot Kinematics 2022. ARK 2022. Springer Proceedings in Advanced Robotics, vol. 24, Springer, Cham, 2022, pp. 248–256

- F. Zaccaria, S. Briot, M. T. Chikhaoui, E. Idà, and M. Carricato, "An analytical formulation for the geometrico-static problem of continuum planar parallel robots," in Venture, G., Solis, J., Takeda, Y., Konno, A. (eds) ROMANSY 23 - Robot Design, Dynamics and Control. ROMANSY 2020. CISM International Centre for Mechanical Sciences, vol. 601, 2021, pp. 512–520

Other Publications (submitted, accepted, or not related to the scope of this thesis):

- F. Zaccaria F, E. Idà , S. Briot "Singularity Conditions of Concentric Tube Robots". Accepted for publication at the 16th Iftomm world congress (WC23), 5-10th Nov 2023, Tokio, Japan

- M. Duchi, F. Zaccaria, S. Briot, E. Idà, "Total Least Squares in-field indetification for MEMS-based Triaxial Accelerometers". Accepted for publication at the 16th Iftomm world congress (WC23), 5-10th Nov 2023, Tokio, Japan.

- F. Zaccaria, E. Ida, and S. Briot. "Directional Critical Load Index: a Distance-to-Instability Metric for Continuum Robots". Under Review to the IEEE Transaction on Robotics (2024).

- F. Zaccaria, E. Quarta, S. Badini, and M. Carricato, "Optimal design for vibration mitigation of a planar parallel mechanism for a fast automatic machine," Machines, vol. 10, no. 9, p. 770, 2022.

- J. Aleotti, A. Baldassarri, M. Bonfè, et al., "Toward future automatic warehouses: An autonomous depalletizing system based on mobile manipulation and 3d perception," Applied Sciences, vol. 11, no. 13, p. 5959, 2021

- F. Zaccaria, A. Baldassarri, G. Palli, and M. Carricato, "A mobile robotized system for depalletizing applications: Design and experimentation," IEEE Access, vol. 9, pp. 96 682–96 691, 2021.

Communications:

- F.Zaccaria, E. Idá, S. Briot, "Design of a Planar Continuum Parallel Robot with Large Workspace Capabilities" (Extended Abstract) Summer school on the topic of deformation in robotics, Lille, France, 2022;

- F.Zaccaria, E. Idá, S. Briot, M.Carricato, "Workspace Computation Algorithms for Continuum Parallel Robots: state-of-the-art and perspectives" (Extended Abstract), New Frontiers in Parallel Robotics (Second Edition), Philadelphia, PA, USA. Workshop at the ICRA2022 Conference.

- F.Zaccaria, E. Idá, S. Briot, M.Carricato, "Challenges on Workspace Evaluation of Continuum Parallel Robots" (Extended Abstract), Parallel Robots or not parallel robots? New frontiers in parallel robotics, ONLINE, Workshop at the ICRA2021 Conference.

Bibliography

[1] G. Robinson and J. B. C. Davies. "Continuum robots-a state of the art". In: *Proceedings 1999 IEEE international conference on robotics and automation (Cat. No. 99CH36288C),, Detroit, MI, USA*. Vol. 4. IEEE. 1999, pp. 2849–2854.

[2] G. S. Chirikjian and J. W. Burdick. "The kinematics of hyper-redundant robot locomotion". In: *IEEE transactions on robotics and automation* 11.6 (1995), pp. 781–793.

[3] A. Vysocky and P. Novak. "Human-robot collaboration in industry". In: *MM Science Journal* 9.2 (2016), pp. 903–906.

[4] C. Wright et al. "Design and architecture of the unified modular snake robot". In: *2012 IEEE international conference on robotics and automation, Saint Paul, MN, USA*. IEEE. 2012, pp. 4347–4354.

[5] A. Das and M. Nabi. "A review on soft robotics: Modeling, control and applications in human-robot interaction". In: *2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), Greater Noida, India*. IEEE. 2019, pp. 306–311.

[6] I. D. Walker. "Continuous backbone "continuum" robot manipulators". In: *International Scholarly Research Notices* 2013 (2013), p. 726506.

[7] R. J. Webster and B. A. Jones. "Design and Kinematic Modeling of Constant Curvature Continuum Robots: A Review". In: *The International Journal of Robotics Research* 29.13 (2010).

[8] J. Burgner-Kahrs, D. C. Rucker, and H. Choset. "Continuum Robots for Medical Applications: A Survey". In: *IEEE Transactions on Robotics* 31.6 (2015), pp. 1261–1280.

[9] E. Amanov, T.-D. Nguyen, and J. Burgner-Kahrs. "Tendon-driven continuum robots with extensible sections—A model-based evaluation of path-following motions". In: *The International Journal of Robotics Research* 40.1 (2021), pp. 7–23.

[10] R. Hendrick et al. "Concentric tube robots for transurethral prostate surgery: Matching the workspace to the endoscopic field of view". In: *The Hamlyn symposium on medical robotics*. 2014, p. 23.

[11] Q. Peyron et al. "Kinematic analysis of magnetic continuum robots using continuation method and bifurcation analysis". In: *IEEE Robotics and Automation Letters* 3.4 (2018), pp. 3646–3653.

[12] S. Kolachalama and S. Lakshmanan. "Continuum Robots for Manipulation Applications: A Survey". In: *Journal of Robotics* 2020 (2020), pp. 1–19.

[13] C. E. Bryson and D. C. Rucker. "Toward Parallel Continuum Manipulators". In: *2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China*. 2014, pp. 778–785.

[14] B. Mauzé et al. "Nanometer precision with a planar parallel continuum robot". In: *IEEE Robotics and Automation Letters* 5.3 (2020), pp. 3806–3813.

[15] C. Nwafor, G. J. Laurent, and K. Rabenorosoa. "Miniature Parallel Continuum Robot Made of Glass: Analysis, Design, and Proof-of-Concept". In: *IEEE/ASME Transactions on Mechatronics* 28.4 (2023), pp. 2038–2046.

[16] A. L. Orekhov et al. "A surgical parallel continuum manipulator with a cable-driven grasper". In: *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Milan, Italy*. IEEE. 2015, pp. 5264–5267.

[17] F. Campa et al. “A 2 *DoFs* continuum parallel robot for pick & place collaborative tasks”. In: *Uhl, T. (eds) Advances in Mechanism and Machine Science. IFToMM WC 2019*. Vol. 73. Springer, 2019, pp. 1979–1988.

[18] C. B. Black, J. Till, and D. C. Rucker. “Parallel continuum robots: Modeling, analysis, and actuation-based force sensing”. In: *IEEE Transactions on Robotics* 34.1 (2017), pp. 29–47.

[19] J. Simo and L. Vu-Quoc. “A three-dimensional finite-strain rod model. Part II: Computational aspects”. In: *Computer methods in applied mechanics and engineering* 58.1 (1986), pp. 79–116.

[20] V. Aloi, C. Black, and C. Rucker. “Stiffness control of parallel continuum robots”. In: *Proceedings of the ASME 2018 Dynamic Systems and Control Conference, Atlanta, Georgia, USA*. Vol. 1. American Society of Mechanical Engineers. 2018, V001T04A012.

[21] K. Leibrandt, C. Bergeles, and G.-Z. Yang. “Concentric Tube Robots: Rapid, Stable Path-Planning and Guidance for Surgical Use”. In: *IEEE Robotics & Automation Magazine* 24.2 (2017), pp. 42–53.

[22] F. Boyer et al. “Dynamics of Continuum and Soft Robots: A Strain Parameterization Based Approach”. In: *IEEE Transactions on Robotics* 37.3 (2021), pp. 847–863.

[23] M. Ghafoori and A. K. Khalaji. “Modeling and experimental analysis of a multi-rod parallel continuum robot using the Cosserat theory”. In: *Robotics and Autonomous Systems* 134 (2020), p. 103650.

[24] E. M. Young and K. J. Kuchenbecker. “Implementation of a 6-DOF parallel continuum manipulator for delivering fingertip tactile cues”. In: *IEEE transactions on haptics* 12.3 (2019), pp. 295–306.

[25] Festo. *Bionic Tripod 3.0*. 2010. URL: `https://www.festo.com/it/it/e/informazioni-su-festo/ricerca-e-sviluppo/bionic-learning-network/highlights-2010-2012/cinematica-tripod-bionica-3-0-id_33747/` (visited on 07/25/2023).

[26] Cod.Act. *Nyloid, performative installation*. 2013. URL: `https://codact.ch/works/nyloid-2/` (visited on 07/25/2023).

[27] A. L. Orekhov, V. A. Aloi, and D. C. Rucker. “Modeling parallel continuum robots with general intermediate constraints”. In: *2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore*. IEEE. 2017, pp. 6142–6149.

[28] G. Wu and G. Shi. “Experimental statics calibration of a multi-constraint parallel continuum robot”. In: *Mechanism and Machine Theory* 136 (2019), pp. 72–85.

[29] H. Pan et al. “Design and Kinematic Analysis of a Flexible-Link Parallel Mechanism With a Spatially Quasi-Translational End Effector”. In: *Journal of Mechanisms and Robotics* 13.1 (2021).

[30] G. Wu and G. Shi. “Design, modeling, and workspace analysis of an extensible rod-driven Parallel Continuum Robot”. In: *Mechanism and Machine Theory* 172 (2022), p. 104798.

[31] O. F. Gallardo et al. “Turning an Articulated 3-PPSR Manipulator into a Parallel Continuum Robot”. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Prague, Czech Republic*. IEEE. 2021, pp. 4955–4960.

[32] B. Mauze et al. “Micrometer positioning accuracy with a planar parallel continuum robot”. In: *Frontiers in Robotics and AI* 8 (2021), p. 196.

[33] A. W. Mahoney et al. “Reconfigurable parallel continuum robots for incisionless surgery”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea (South)*. IEEE. 2016, pp. 4330–4336.

[34] O. Altuzarra et al. “Forward and inverse kinematics in 2-dof planar parallel continuum manipulators”. In: *EuCoMeS 2018: Proceedings of the 7th European Conference on Mechanism Science 7*. Springer. 2019, pp. 231–238.

[35] G. Chen et al. “Kinetostatics modeling and analysis of parallel continuum manipulators”. In: *Mechanism and Machine Theory* 163 (2021), p. 104380.

[36] X. Duan et al. “Analysis and validation of a planar parallel continuum manipulator with variable Cartesian stiffness”. In: *Mechanism and Machine Theory* 177 (2022), p. 105030.

[37] S. Lilge et al. “Tendon Actuated Continuous Structures in Planar Parallel Robots: A Kinematic Analysis”. In: *Journal of Mechanisms and Robotics* 13 (1 2020), p. 011025.

[38] K. Nuelle et al. “Modeling, Calibration, and Evaluation of a Tendon-Actuated Planar Parallel Continuum Robot”. In: *IEEE Robotics and Automation Letters* 5.4 (2020), pp. 5811–5818.

[39] G. Böttcher, S. Lilge, and J. Burgner-Kahrs. “Design of a reconfigurable parallel continuum robot with tendon-actuated kinematic chains”. In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 1272–1279.

[40] J. A. Childs and C. Rucker. “Leveraging geometry to enable high-strength continuum robots”. In: *Frontiers in Robotics and AI* 8 (2021).

[41] C. Troeung and C. Chen. “A Translational Parallel Continuum Robot Reinforced by Origami and Cross-Routing Tendons”. In: *021 IEEE International Conference on Robotics and Automation (ICRA), Xi’an, China*. IEEE. 2021, pp. 7039–7045.

[42] S. Patel and T. Sobh. “Manipulator performance measures-a comprehensive literature survey”. In: *Journal of Intelligent & Robotic Systems* 77 (2015), pp. 547–570.

[43] J.-P. Merlet. *Parallel robots*. Vol. 128. Springer Science & Business Media, 2005.

[44] J.-P. Merlet. “Jacobian, Manipulability, Condition Number, and Accuracy of Parallel Robots”. In: *Journal of Mechanical Design* 128.1 (2005), pp. 199–206.

[45] C. Gosselin and J. Angeles. “A global performance index for the kinematic optimization of robotic manipulators”. In: *ASME. Journal Mechanical Design* 113.3 (1991), pp. 220–226.

[46] T. Yoshikawa. “Manipulability of robotic mechanisms”. In: *The international journal of Robotics Research* 4.2 (1985), pp. 3–9.

[47] K. Leibrandt, L. da Cruz, and C. Bergeles. “Designing Robots for Reachability and Dexterity: Continuum Surgical Robots as a Pretext Application”. In: *IEEE Transactions on Robotics* 39.4 (2023), pp. 2989–3007.

[48] M. T. Chikhaoui, K. Rabenorosoa, and N. Andreff. “Kinematics and performance analysis of a novel concentric tube robotic structure with embedded soft micro-actuation”. In: *Mechanism and Machine Theory* 104 (2016), pp. 234–254.

[49] I. A. Gravagne and I. D. Walker. “Manipulability, force, and compliance analysis for planar continuum manipulators”. In: *IEEE Transactions on Robotics and Automation* 18.3 (2002), pp. 263–273.

[50] D. C. Rucker and R. J. Webster. “Computing jacobians and compliance matrices for externally loaded continuum robots”. In: *2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 2011*. IEEE. 2011, pp. 945–950.

[51] G. Chen et al. “Analysis and validation of a flexible planar two degrees-of-freedom parallel manipulator with structural passive compliance”. In: *Journal of Mechanisms and Robotics* 12.1 (2020), p. 011011.

[52] Z. Du, W. Yang, and W. Dong. “Kinematics modeling and performance optimization of a kinematic-mechanics coupled continuum manipulator”. In: *Mechatronics* 31 (2015), pp. 196–204.

[53] K. Wen and J. Burgner-Kahrs. “Modeling and analysis of tendon-driven parallel continuum robots under constant curvature and pseudo-rigid-body assumptions”. In: *Journal of Mechanisms and Robotics* 15.4 (2023), p. 041003.

[54] S. Lilge, K. Wen, and J. Burgner-Kahrs. “Singularity analysis of 3-DOF planar parallel continuum robots with constant curvature links”. In: *Frontiers in Robotics and AI* 9 (2023), p. 1082185.

[55] A. Mayer and O. Sawodny. “Singularity and Workspace Analysis for Modular Continuum Robots”. In: *2018 IEEE Conference on Control Technology and Applications (CCTA), Copenhagen, Denmark*. 2018, pp. 280–285.

[56] S. Briot and A. Goldsztejn. “Singularity Conditions for Continuum Parallel Robots”. In: *IEEE Transactions on Robotics* 38.1 (2022), pp. 507–525.

[57] J. Ha, F. C. Park, and P. E. Dupont. “Elastic stability of concentric tube robots subject to external loads”. In: *IEEE Transactions on Biomedical Engineering* 63.6 (2015), pp. 1116–1128.

[58] J. Till and D. C. Rucker. “Elastic Stability of Cosserat Rods and Parallel Continuum Robots”. In: *IEEE Transactions on Robotics* 33.3 (2017), pp. 718–733.

[59] K. Leibrandt, C. Bergeles, and G.-Z. Yang. “On-line collision-free inverse kinematics with frictional active constraints for effective control of unstable concentric tube robots”. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany*. IEEE. 2015, pp. 3797–3804.

[60] M. Khadem, L. Da Cruz, and C. Bergeles. “Force/velocity manipulability analysis for 3d continuum robots”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain*. IEEE. 2018, pp. 4920–4926.

[61] L. Wu, R. Crawford, and J. Roberts. “Dexterity analysis of three 6-DOF continuum robots combining concentric tube mechanisms and cable-driven mechanisms”. In: *IEEE Robotics and Automation Letters* 2.2 (2016), pp. 514–521.

[62] C. Wang et al. “Task space-based orientability analysis and optimization of a wire-driven continuum robot”. In: *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 233.23-24 (2019), pp. 7658–7668.

[63] J. Wang and H. Y. Lau. “Dexterity analysis based on Jacobian and performance optimization for multi-segment continuum robots”. In: *Journal of Mechanisms and Robotics* 13.6 (2021), p. 061012.

[64] Y. Chen et al. “Kinematic optimization of a continuum surgical manipulator”. In: *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO), Kuala Lumpur, Malaysia*. IEEE. 2018, pp. 2069–2074.

[65] K. Xu, J. Zhao, and X. Zheng. “Configuration comparison among kinematically optimized continuum manipulators for robotic surgeries through a single access port”. In: *Robotica* 33.10 (2015), pp. 2025–2044.

[66] C. Baykal, C. Bowen, and R. Alterovitz. “Asymptotically optimal kinematic design of robots using motion planning”. In: *Autonomous robots* 43.2 (2019), pp. 345–357.

[67] C. Bergeles et al. “Concentric tube robot design and optimization based on task and anatomical constraints”. In: *IEEE Transactions on Robotics* 31.1 (2015), pp. 67–84.

[68] I. A. Bonev and J. Ryu. “A geometrical method for computing the constant-orientation workspace of 6-PRRS parallel manipulators”. In: *Mechanism and machine theory* 36.1 (2001), pp. 1–13.

[69] W. Amehri, G. Zheng, and A. Kruszewski. “Fem based workspace estimation for soft robots: A forward-backward interval analysis approach”. In: *2020 3rd IEEE International Conference on Soft Robotics (RoboSoft), Yale, USA*. 2020, pp. 170–175.

[70] D. Trivedi, D. Lesutis, and C. D. Rahn. “Dexterity and workspace analysis of two soft robotic manipulators”. In: *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Quebec, Canada*. 2010, pp. 1389–1398.

[71] J. Burgner-Kahrs et al. “Workspace characterization for concentric tube continuum robots”. In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, USA*. 2014, pp. 1269–1275.

[72] A. L. Orekhov et al. “Analysis and validation of a teleoperated surgical parallel continuum manipulator”. In: *IEEE Robotics and Automation Letters* 1.2 (2016), pp. 828–835.

[73] I. Singh et al. “Optimal work space of parallel continuum manipulator consisting of compact bionic handling arms”. In: *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO), Macau, Macao*. IEEE. 2017, pp. 258–263.

[74] D. C. Rucker et al. "Equilibrium conformations of concentric-tube continuum robots". In: *The International journal of robotics research* 29.10 (2010), pp. 1263–1280.

[75] J. Till et al. "Efficient computation of multiple coupled Cosserat rod models for real-time simulation and control of parallel continuum manipulators". In: *2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA*. IEEE. 2015, pp. 5067–5074.

[76] C. Escande et al. "Kinematic calibration of a multisection bionic manipulator". In: *IEEE/ASME transactions on mechatronics* 20.2 (2014), pp. 663–674.

[77] D. Oetomo et al. "Certified workspace analysis of 3RRR planar parallel flexure mechanism". In: *2008 IEEE International Conference on Robotics and Automation, Pasadena, CA, USA*. IEEE. 2008, pp. 3838–3843.

[78] O. Altuzarra et al. "Position analysis in planar parallel continuum mechanisms". In: *Mechanism and Machine Theory* 132 (Feb. 2019), pp. 13–29.

[79] O. Altuzarra and J. P. Merlet. "Certified Kinematics Solution of 2-DOF Planar Parallel Continuum Mechanisms". In: *Advances in Mechanism and Machine Science, IFToMM WC 2019. Mechanisms and Machine Science*. Ed. by T. Uhl. Vol. 73. 2019, pp. 197–208.

[80] F. Zaccaria et al. "An Analytical Formulation for the Geometrico-Static Problem of Continuum Planar Parallel Robots". In: *ROMANSY 23 - Robot Design, Dynamics and Control CISM. ROMANSY 2020. International Centre for Mechanical Sciences*. Ed. by G. Venture et al. Vol. 601. Cham: Springer International Publishing, 2021, pp. 512–520.

[81] Y. Wang et al. "Inverse Kinematics and Dexterous Workspace Formulation for 2-Segment Continuum Robots With Inextensible Segments". In: *IEEE Robotics and Automation Letters* 7.1 (2022), pp. 510–517.

[82] W. Amehri, G. Zheng, and A. Kruszewski. "Workspace Boundary Estimation for Soft Manipulators Using a Continuation Approach". In: *IEEE Robotics and Automation Letters* 6.4 (2021), pp. 7169–7176.

[83] F. Renda et al. "A Geometric Variable-Strain Approach for Static Modeling of Soft Manipulators With Tendon and Fluidic Actuation". In: *IEEE Robotics and Automation Letters* 5.3 (2020), pp. 4006–4013.

[84] W. Amehri et al. "Discrete Cosserat Method for Soft Manipulators Workspace Estimation: an Optimization-Based Approach". In: *Journal of Mechanisms and Robotics* (2021), pp. 1–12.

[85] F. Renda et al. "Discrete cosserat approach for multisection soft manipulator dynamics". In: *IEEE Transactions on Robotics* 34.6 (2018), pp. 1518–1533.

[86] W. Amehri, G. Zheng, and A. Kruszewski. "FEM-Based Exterior Workspace Boundary Estimation for Soft Robots via Optimization". In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 3672–3678.

[87] K. Abdel-Malek and H.-J. Yeh. "Analytical boundary of the workspace for general 3-dof mechanisms". In: *The International Journal of Robotics Research* 16.2 (1997), pp. 198–213.

[88] E. J. Haug et al. "Numerical Algorithms for Mapping Boundaries of Manipulator Workspaces". In: *Journal of Mechanical Design* 118.2 (1996), pp. 228–234.

[89] J. Snyman, L. Du Plessis, and J. Duffy. "An optimization approach to the determination of the boundaries of manipulator workspaces". In: *J. Mech. Des.* 122.4 (2000), pp. 447–456.

[90] R. Xu, S. F. Atashzar, and R. V. Patel. "Kinematic instability in concentric-tube robots: Modeling and analysis". In: *5th IEEE RAS/EMBS International Conference on Biomedical Robotics and Biomechatronics, Sao Paulo, Brazil*. IEEE. 2014, pp. 163–168.

[91] J. Ha, F. C. Park, and P. E. Dupont. "Optimizing tube precurvature to enhance the elastic stability of concentric tube robots". In: *IEEE Transactions on Robotics* 33.1 (2016), pp. 22–37.

[92] C. Rucker et al. "Transverse anisotropy stabilizes concentric tube robots". In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 2407–2414.

[93] R. J. Hendrick, H. B. Gilbert, and R. J. Webster. "Designing snap-free concentric tube robots: A local bifurcation approach". In: *2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA*. 2015, pp. 2256–2263.

[94] L. Euler. *Methodus inveniendi lineas curvas maximi minimive proprietate gaudentes sive solutio problematis isoperimetrici latissimo sensu accepti*. Vol. 1. Springer Science & Business Media, 1952.

[95] J. Maddocks. "Stability of nonlinearly elastic rods". In: *Archive for rational mechanics and analysis* 85.4 (1984), pp. 311–354.

[96] A. H. Nayfeh and S. A. Emam. "Exact solution and stability of postbuckling configurations of beams". In: *Nonlinear Dynamics* 54 (2008), pp. 395–408.

[97] A. Lazarus, J. Miller, and P. M. Reis. "Continuation of equilibria and stability of slender elastic rods using an asymptotic numerical method". In: *Journal of the Mechanics and Physics of Solids* 61.8 (2013), pp. 1712–1736.

[98] H. B. Gilbert, R. J. Hendrick, and R. J. Webster III. "Elastic stability of concentric tube robots: A stability measure and design test". In: *IEEE Transactions on Robotics* 32.1 (2015), pp. 20–35.

[99] O. Altuzarra and F. J. Campa. "On Singularity and Instability in a Planar Parallel Continuum Mechanism". In: *Lenarčič, J., Siciliano, B. (eds) Advances in Robot Kinematics 2020. ARK 2020. Springer Proceedings in Advanced Robotics*, pp. 327–334.

[100] O. Altuzarra et al. "Kinematic Analysis of three degrees of freedom planar parallel continuum mechanisms". In: *Mechanism and Machine Theory* 185 (2023), p. 105311.

[101] S. Briot and J.-P. Merlet. "Direct kinematic singularities and stability analysis of sagging cable-driven parallel robots". In: *IEEE Transactions on Robotics* 39.3 (2023), pp. 2240–2254.

[102] Q. Peyron et al. "A numerical framework for the stability and cardinality analysis of concentric tube robots: Introduction and application to the follow-the-leader deployment". In: *Mechanism and Machine Theory* 132 (2019), pp. 176–192.

[103] M. Carricato and J.-P. Merlet. "Stability analysis of underconstrained cable-driven parallel robots". In: *IEEE Transactions on Robotics* 29.1 (2012), pp. 288–296.

[104] K. Leibrandt, C. Bergeles, and G.-Z. Yang. "Implicit active constraints for safe and effective guidance of unstable concentric tube robots". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea (South)*. 2016, pp. 1157–1163.

[105] W. Yan et al. "Design of a reconfigurable planar parallel continuum manipulator with variable stiffness". In: *Intelligent Robotics and Applications: 14th International Conference, ICIRA 2021, Yantai, China*. 2021, pp. 803–813.

[106] C. Armanini et al. "Soft Robots Modeling: A Structured Overview". In: *IEEE Transactions on Robotics* 39.3 (2023), pp. 1728–1748.

[107] G. Kirchhoff. "Ueber das Gleichgewicht und die Bewegung eines unendlich dünnen elastischen Stabes." In: *Journal für die reine und angewandte Mathematik* 1859.56 (1859), pp. 285–313.

[108] A. E. H. Love. *A treatise on the mathematical theory of elasticity*. Cambridge university press, New york, 1892.

[109] I. Vardoulakis. *Cosserat Continuum Mechanics*. Springer International Publishing, 2019.

[110] E. Reissner. "On finite deformations of space-curved beams". In: *Zeitschrift für angewandte Mathematik und Physik ZAMP* 32.6 (1981), pp. 734–744.

[111] J. Simo. "A finite strain beam formulation. The three-dimensional dynamic problem. I". In: *Computer methods in applied mechanics and engineering* 49.1 (1985), pp. 55–70.

[112] R. M. Murray et al. *A mathematical introduction to robotic manipulation*. CRC press, Boca Raton, 1994.

[113] S. Lilge and J. Burgner-Kahrs. "Kinetostatic Modeling of Tendon-Driven Parallel Continuum Robots". In: *IEEE Transactions on Robotics* 39.2 (2022), pp. 1563–1579.

[114] A. Zhang and G. Chen. "A comprehensive elliptic integral solution to the large deflection problems of thin beams in compliant mechanisms". In: *Journal of Mechanisms and Robotics* 5.2 (2013).

[115] O. Altuzarra et al. "Kinematic Analysis of a Continuum Parallel Robot". In: *Wenger, P., Flores, P. (eds) New Trends in Mechanism and Machine Science. Mechanisms and Machine Science*. Vol. 43. 2016, pp. 173–180.

[116] O. Altuzarra et al. "Forward and Inverse Kinematics in 2-DOF Planar Parallel Continuum Manipulators". In: *Corves, B., Wenger, P., Hüsing, M. (eds) EuCoMeS 2018 . EuCoMeS 2018. Mechanisms and Machine Science*. Vol. 59. Springer, Cham., 2018.

[117] O. Altuzarra et al. "Kinematic Characteristics of Parallel Continuum Mechanisms". In: *Lenarcic, J., Parenti-Castelli, V. (eds) Advances in Robot Kinematics 2018. ARK 2018. Springer Proceedings in Advanced Robotics*. 2018, pp. 293–301.

[118] L. Li et al. "Shape Modeling of a Parallel Soft Panel Continuum Robot". In: *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO), Kuala Lumpur, Malaysia)*. 2018, pp. 367–372.

[119] B. A. Jones and I. D. Walker. "Kinematics for multisection continuum robots". In: *IEEE Transactions on Robotics* 22.1 (2006), pp. 43–55.

[120] I. A. Gravagne, C. D. Rahn, and I. D. Walker. "Large deflection dynamics and control for planar continuum robots". In: *IEEE/ASME transactions on mechatronics* 8.2 (2003), pp. 299–307.

[121] C. Li and C. D. Rahn. "Design of continuous backbone, cable-driven robots". In: *J. Mech. Des.* 124.2 (2002), pp. 265–271.

[122] I. S. Godage et al. "Dynamics for variable length multisection continuum arms". In: *The International Journal of Robotics Research* 35.6 (2016), pp. 695–722.

[123] W. Khalil, G. Gallot, and F. Boyer. "Dynamic modeling and simulation of a 3-D serial eel-like robot". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 37.6 (2007), pp. 1259–1268.

[124] T. Zheng et al. "Dynamic continuum arm model for use with underwater robotic manipulators inspired by octopus vulgaris". In: *2012 IEEE International Conference on Robotics and Automation, Saint Paul, MN, USA*. 2012, pp. 5289–5294.

[125] A. Quarteroni, R. Sacco, and F. Saleri. *Numerical mathematics*. Vol. 37. Springer Science & Business Media, London, 2010.

[126] Q. Peyron et al. "Evaluation of Dynamic Relaxation to Solve Kinematics of Concentric Tube Robots". In: *Advances in Robot Kinematics 2018. ARK 2018. Springer Proceedings in Advanced Robotics*. Ed. by J. Lenarcic and V. Parenti-Castelli. Vol. 8. Springer International Publishing, 2018, pp. 100–107.

[127] J. Spillmann and M. Teschner. "CoRdE: Cosserat rod elements for the dynamic simulation of one-dimensional elastic objects". In: *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation, San Diego, California, USA*. 2007, pp. 63–72.

[128] S. Briot and F. Boyer. "A Geometrically Exact Assumed Strain Modes Approach for the Geometrico- and Kinemato-Static Modelings of Continuum Parallel Robots". In: *IEEE Transactions on Robotics* 39.3 (2022), pp. 2240–2254.

[129] P. K. Masjedi and H. Ovesy. "Chebyshev collocation method for static intrinsic equations of geometrically exact beams". In: *International Journal of Solids and Structures* 54 (2015), pp. 183–191.

[130] L. Greco and M. Cuomo. "B-Spline interpolation of Kirchhoff-Love space rods". In: *Computer Methods in Applied Mechanics and Engineering* 256 (2013), pp. 251–269.

[131] A. L. Orekhov and N. Simaan. "Solving cosserat rod models via collocation and the magnus expansion". In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA*. IEEE, 2020, pp. 8653–8660.

[132] F. Auricchio et al. “Isogeometric collocation methods”. In: *Mathematical Models and Methods in Applied Sciences* 20.11 (2010), pp. 2075–2107.

[133] J. Allard et al. “Sofa-an open source framework for medical simulation”. In: *MMVR 15-Medicine Meets Virtual Reality*. Vol. 125. IOP Press. 2007, pp. 13–18.

[134] F. Faure et al. “Sofa: A multi-model framework for interactive physical simulation”. In: *Payan, Y. (eds) Soft Tissue Biomechanical Modeling for Computer Assisted Surgery. Studies in Mechanobiology, Tissue Engineering and Biomaterials*. Springer, 2012, pp. 283–321.

[135] T. M. Bieze et al. “Finite element method-based kinematics and closed-loop control of soft, continuum manipulators”. In: *Soft robotics* 5.3 (2018), pp. 348–364.

[136] M. Koehler et al. “Modeling and Control of a 5-DOF Parallel Continuum Haptic Device”. In: *IEEE Transactions on Robotics* (2023). DOI: 10.1109/TRO.2023.3277068.

[137] A. T. Mathew et al. “SoRoSim: A MATLAB Toolbox for Hybrid Rigid-Soft Robots Based on the Geometric Variable-Strain Approach”. In: *IEEE Robotics and Automation Magazine* (2022). DOI: 10.1109/MRA.2022.3202488.

[138] F. Renda et al. “Discrete Cosserat approach for soft robot dynamics: A new piece-wise constant strain model with torsion and shears”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea (South)*. IEEE. 2016, pp. 5495–5502.

[139] H. Li et al. “Discrete Cosserat Static Model-Based Control of Soft Manipulator”. In: *IEEE Robotics and Automation Letters* 8.3 (2023), pp. 1739–1746.

[140] F. Boyer et al. “Statics and Dynamics of Continuum Robots Based on Cosserat Rods and Optimal Control Theories”. In: *IEEE Transactions on Robotics* 39.2 (2023), pp. 1544–1562.

[141] F. Boyer et al. “Extended Hamilton’s principle applied to geometrically exact Kirchhoff sliding rods”. In: *Journal of Sound and Vibration* 516 (2022), p. 116511.

[142] J. Till, V. Aloi, and C. Rucker. “Real-time dynamics of soft and continuum robots based on Cosserat rod models”. In: *The International Journal of Robotics Research* 38.6 (2019), pp. 723–746.

[143] S. Antman. *Nonlinear Problems of Elasticity*. Vol. 107. Springer Verlag New York, 1995.

[144] H. Ziegler. *Principles of structural stability*. Vol. 35. Birkhäuser, 2013.

[145] J. Nocedal and S. Wright. *Numerical Optimization*. 2nd. Springer, 2006.

[146] F. W. Olver. *NIST handbook of mathematical functions hardback and CD-ROM*. Cambridge university press, 2010.

[147] E. Hairer et al. “Geometric numerical integration”. In: *Oberwolfach Reports* 3.1 (2006), pp. 805–882.

[148] F. Zaccaria et al. “Workspace Computation of Planar Continuum Parallel Robots”. In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 2700–2707.

[149] N. Tan, X. Gu, and H. Ren. “Pose characterization and analysis of soft continuum robots with modeling uncertainties based on interval arithmetic”. In: *IEEE Transactions on Automation Science and Engineering* 16.2 (2018), pp. 570–584.

[150] S. Iqbal, S. Mohammed, and Y. Amirat. “A guaranteed approach for kinematic analysis of continuum robot based catheter”. In: *2009 IEEE International Conference on Robotics and Biomimetics (ROBIO), Guilin, Chins*. IEEE. 2009, pp. 1573–1578.

[151] L. Kantorovich. “On Newton’s method for functional equations”. In: *Dokl. Akad. Nauk SSSR*. Vol. 59. 7. 1948, pp. 1237–1240.

[152] A. Goldsztejn, S. Caro, and G. Chabert. “A three-step methodology for dimensional tolerance synthesis of parallel manipulators”. In: *Mechanism and Machine Theory* 105 (2016), pp. 213–234.

[153] L. Kantorovich. “On Newton’s method for functional equations”. In: *Dokl. Akad. Nauk SSSR*. Vol. 59. 7. 1948, pp. 1237–1240.

[154] I. A. Bonev, D. Chablat, and P. Wenger. "Working and assembly modes of the Agile Eye". In: *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA, Orlando, Florida, USA*. 2006, pp. 2317–2322.

[155] P. Wenger. "Cuspidal and noncuspidal robot manipulators". In: *Robotica* 25.6 (2007), pp. 677–689.

[156] F. Zaccaria, E. Idá, and S. Briot. "A Boundary Computation Algorithm for the Workspace Evaluation of Continuum Parallel Robots". In: *Journal of Mechanisms and Robotics* 16.4 (2023).

[157] I. A. Bonev and J. Ryu. "A new approach to orientation workspace analysis of 6-*DoFs* parallel manipulators". In: *Mechanism and machine theory* 36.1 (2001), pp. 15–28.

[158] F. Zaccaria, E. Idá, and S. Briot. "Design and Experimental Equilibrium Stability Assessment of a RFRFR Continuum Parallel Robot". In: *Mechatronics* 95 (2023), p. 103064.

[159] A. Chawla, C. Frazelle, and I. Walker. "A comparison of constant curvature forward kinematics for multisection continuum manipulators". In: *2018 Second IEEE International Conference on Robotic Computing (IRC)*. IEEE. 2018, pp. 217–223.

[160] S. M. H. Sadati et al. "Control Space Reduction and Real-Time Accurate Modeling of Continuum Manipulators Using Ritz and Ritz–Galerkin Methods". In: *IEEE Robotics and Automation Letters* 3.1 (2018), pp. 328–335.

[161] M. T. Chikhaoui et al. "Comparison of modeling approaches for a tendon actuated continuum robot with three extensible segments". In: *IEEE Robotics and Automation Letters* 4.2 (2019), pp. 989–996.

[162] J. Jung, R. S. Penning, and M. R. Zinn. "A modeling approach for robotic catheters: effects of nonlinear internal device friction". In: *Advanced Robotics* 28.8 (2014), pp. 557–572.

[163] I. S. Godage et al. "Accurate and efficient dynamics for variable-length continuum arms: A center of gravity approach". In: *Soft Robotics* 2.3 (2015), pp. 96–106.

[164] A. Jalali and F. Janabi-Sharifi. "Dynamic Modeling of Tendon-Driven Co-Manipulative Continuum Robots". In: *IEEE Robotics and Automation Letters* 7.2 (2021), pp. 1643–1650.

[165] J.-P. Merlet and D. Daney. "Legs interference checking of parallel robots over a given workspace or trajectory". In: *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* IEEE. 2006, pp. 757–762.

[166] L. Campos et al. "Development of a five-bar parallel robot with large workspace". In: *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. Vol. 44106. 2010, pp. 917–922.

[167] B. S. Institution. *BS EN 13706–2:2002: Reinforced Plastic Composites - Specification for Pultruded Profiles - Part 2: Methods of Test and General Requirements*. London, United Kingdom: BSI Standards Limited, 2002.

[168] B. A. Jones, R. L. Gray, and K. Turlapati. "Three dimensional statics for continuum robotics". In: *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2009, pp. 2659–2664.

[169] S. M. Kay. *Fundamentals of statistical signal processing: estimation theory*. Prentice-Hall, Inc., 1993.

[170] G. Bradski. "The openCV library." In: *Dr. Dobb's Journal: Software Tools for the Professional Programmer* 25.11 (2000), pp. 120–123.

[171] D. Zlatanov, I. A. Bonev, and C. M. Gosselin. "Constraint singularities of parallel mechanisms". In: *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*. Vol. 1. IEEE. 2002, pp. 496–502.

[172] A. Gotelli et al. "A Gazebo Simulator for Continuum Parallel Robots". In: *Altuzarra, O., Kecskeméthy, A. (eds) Advances in Robot Kinematics 2022. ARK 2022. Springer Proceedings in Advanced Robotics*. Vol. 24. Springer, 2022, pp. 248–256.

[173] F. Zaccaria, E. Idá, and S. Briot. "Directional Critical Load Index: a Distance-to-Instability Metric for Continuum Robots". In: *submitted to the IEEE Transaction on Robotics* (2024).

[174] J. R. Magnus and H. Neudecker. *Matrix differential calculus with applications in statistics and econometrics*. John Wiley & Sons, 2019.

[175] R. H. Byrd and R. B. Schnabel. "Continuity of the null space basis and constrained optimization". In: *Mathematical Programming* 35.1 (1986), pp. 32–41.

[176] T. F. Coleman and A. Pothen. "The null space problem II. Algorithms". In: *SIAM Journal on Algebraic Discrete Methods* 8.4 (1987), pp. 544–563.

[177] E. Idà, S. Briot, and M. Carricato. "Natural oscillations of underactuated cable-driven parallel robots". In: *IEEE Access* 9 (2021), pp. 71660–71672.

[178] S. Briot and F. Boyer. "Technical report associated with the paper:'A geometrically-exact assumed strain modes approach for the geometrico-and kinemato-static modellings of continuum parallel robots'". In: *Laboratoire des Sciences du Numérique de Nantes (LS2N), Tech. Rep* 3836288 (2022).

[179] F. Renda et al. "A sliding-rod variable-strain model for concentric tube robots". In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 3451–3458.

[180] S. Briot and A. Goldsztejn. "Technical report associated with the paper 'singularity conditions for continuum parallel robots'". In: *CNRS, Laboratoire des Sciences du Numérique de Nantes* (2020).

COLLEGE SCIENCES
DOCTORAL DE L'INGENIERIE
PAYS DE LA LOIRE ET DES SYSTEMES

CENTRALE NANTES

Titre : Contributions à l'analyse de la performance des robots parallèles continus.

Mots clés: Robots parallèles continus, analyse de performances, espace de travail, stabilité des equilibres.

Résumé: Les robots parallèles continus (RPCs) sont des manipulateurs utilisant plusieurs tiges flexibles disposées en parallèle et connectées à une plateforme rigide. Les RCPs promettent des capacités de charge et une précision plus élevée que les robots sériels continus, tout en gardant une grande flexibilité. Puisque le risque de blessure lors d'un contact accidentel entre un humain et un CPR devrait être réduit, les RCPs peuvent être utilisés dans des tâches collaboratives à grande échelle ou dans de tâches de chirurgie robotique assistée. Différentes architectures de RCP existent, mais la conception du prototype est rarement basée sur des considérations de performance, et la réalisation de RCPs est principalement à partir d'intuitions en utilisant d'architectures de manipulateurs parallèles rigides. Les thèmes de recherche de cette thèse portent sur l'analyse des performances des RCPs, et sur les outils nécessaires à une telle évaluation, ainsi que sur les algorithmes de calcul de leur espace de travail. En particulier, les stratégies de calcul de l'espace de travail pour les RCPs sont essentielles pour l'´évaluation de la performance, car l'espace de travail peut être utilisé comment un indice de performance, par exemple pour des outils de conception optimale. Deux nouveaux algorithmes de calcul de l'espace de travail sont proposés dans ce manuscrit, le premier se concentrant sur le calcul du volume de l'espace de travail et la certification de ses résultats numériques, et le second sur le calcul des bords de l'espace de travail uniquement. En raison de la nature ´élastique des RCPs, un indicateur de performance essentiel pour ces robots est la stabilité de leurs configurations d'équilibre. Cette thèse propose la validation expérimentale de l'´évaluation de la stabilité des équilibres sur un prototype réel, démontrant les limites de certaines hypothèses couramment utilisées. De plus, un indice de performance mesurant la distance à l'instabilité est proposé dans ce manuscrit. Contrairement à la majorité des approches existantes, l'avantage évident de l'indice proposé est une signification physique bien défini

Title: Contributions to the Performance Analysis of Continuum Parallel Robots

Keywords: Continuum parallel robots, performance analysis, workspace, equilibrium stability.

Abstract: Continuum parallel robots (CPRs) are manipulators employing multiple flexible beams arranged in parallel and connected to a rigid end-effector. CPRs promise higher payload and accuracy than serial CRs while keeping great flexibility. As the risk of injury during accidental contact between a human and a CPR should be reduced, CPRs may be used in large-scale collaborative tasks or assisted robotic surgery. There exist various CPR designs, but the prototype conception is rarely based on performance considerations, and the CPRs realization in mainly based on intuitions or rigid-link parallel manipulators architectures. This thesis focuses on the performance analysis of CPRs, and the tools needed for such evaluation, such as workspace computation algorithms. In particular, workspace computation strategies for CPRs are essential for the performance assessment, since the CPRs workspace may be used as a performance index or it can serve for optimal-design tools. Two new workspace computation algorithms are proposed in this manuscript, the former focusing on the workspace volume computation and the certification of its numerical results, while the latter aims at computing the workspace boundary only. Due to the elastic nature of CPRs, a key performance indicator for these robots is the stability of their equilibrium configurations. This thesis proposes the experimental validation of the equilibrium stability assessment on a real prototype, demonstrating limitations of some commonly used assumptions. Additionally, a performance index measuring the distance to instability is originally proposed in this manuscript. Differently from the majority of the existing approaches, the clear advantage of the proposed index is a sound physical meaning; accordingly, the index can be used for a more straightforward performance quantification, and to derive robot specifications.