# Sensitivity analysis and optimization of the dynamics of multibody systems using analytical gradient based methods

**Álvaro López Varela**

Advisor: Daniel Dopico Dopico

Doctoral thesis

UNIVERSIDADE DA CORUÑA

Programa Oficial de Doutoramento
en Enxeñaría Naval e Industrial

Ferrol, 2022

Dr. Daniel Dopico Dopico, Doctor by the University of A Coruña certifies that this doctoral dissertation, entitled *Sensitivity analysis and optimization of the dynamics of multibody systems using analytical gradient based methods*, has been developed by Álvaro López Varela under his supervision in order to obtain the International Doctor mention by the University of A Coruña.

Dr. Daniel Dopico Dopico, Doctor por la Universidade da Coruña certifica que la presente memoria, titulada *Sensitivity analysis and optimization of the dynamics of multibody systems using analytical gradient based methods*, ha sido desarrollada por Álvaro López Varela bajo su supervisión para optar al grado de Doctor con mención Internacional por la Universidade de Coruña.

Dr. Daniel Dopico Dopico, Doutor pola Universidad de A Coruña certifica que a presente memoria, titulada *Sensitivity analysis and optimization of the dynamics of multibody systems using analytical gradient based methods*, foi desenvolvida por Álvaro López Varela baixo a súa supervisión para optar ó grao de Doutor con mención Internacional pola Universidade da Coruña.

Ferrol, 2022.

| | |
|---|---|
| **Álvaro López Varela** | **Dr. Daniel Dopico Dopico** |
| *PhD student* | *Advisor* |
| *Doctorando* | *Director* |
| *Doutorando* | *Director* |

*Á miña madriña*

# Acknowledgments

The jump from the private sector to research in a public university seems quite difficult nowadays, specially after the hard process to reach an employment in accordance with your qualification. In my case, the decision was very meditated, but I do not regret the selection of the research career, in which I wish I would be involved in coming years. In this sense, I need to thank Alfredo del Caño Gochi for connecting me with the Laboratorio de Ingeniería Mecánica (LIM) and informing me about the opportunity to develop a doctoral thesis in this wonderful group.

Above all, I would like to express my sincere gratitude towards my advisor Daniel Dopico Dopico for his guidance and help during this period. Without his patience and support, this work would have been impossible. Moreover, I should remark that my field of expertise was Electronic Engineering, but I did not have important problems in my introduction to Multibody Dynamics thanks to his thoughtful and detailed explanations.

Additionally, I extend this acknowledgment to all the members of LIM for the extremely good work environment, and especially to Alberto Luaces Fernández for assisting me with the debugging of the code and for sharing with me his programming skills and knowledge.

Moreover, I want to thank professor Olivier Brüls for offering me the opportunity to explore new fields of multibody dynamics during my research stay in the Multibody & Mechatronic Systems Laboratory at the University of Liège. I really value his support and all the material and lessons he have gifted me. I would like to also thank Juliano Todesco for his invaluable collaboration, and to Facundo Cosimo for making my stay more comfortable.

Finally, I would like to thank my family for their unwavering support, particularly to my parents Rosa María and Manuel. They have taught me more than anyone else, and I still have a lot to learn from them. I cannot forget the ones that are now with me, Clara, Moncho, Rubén, abuela Clara..., and those who left us but who are still present in our lives.

# Agradecimientos

El salto del sector privado a la investigación en una universidad pública parece muy difícil actualmente, especiamente después del duro proceso de alcanzar un trabajo acorde con tu cualificación. En mi caso, la decisión fue muy meditada, pero no me arrepiento de la selección de la carrera de investigación, en la cual deseo estar involucrado en los próximos años. En este sentido, necesito agradecer a Alfredo del Caño Gochi que me conectara con el Laboratorio de Ingeniería Mecánica (LIM) y por informarme de la oportunidad de desarrollar una tesis doctoral en este maravilloso grupo.

Por encima de todo, me gustaría expresar mi más sincera gratitud hacia mi director Daniel Dopico Dopico por su ayuda y dirección durante este periodo. Sin su paciencia y apoyo, este trabajo habría sido imposible. Además, debo destacar que mi campo de especialidad era Ingeniería Electrónica, pero gracias a sus razonadas y detalladas explicaciones no tuve demasiados problemas en mi introducción en la dinámica multicuerpo.

Adicionalmente, extiendo este agradecimiento a todos los miembros del LIM por el extremadamente buen ambiente de trabajo, y especialmente a Alberto Luaces Fernández por su apoyo durante el proceso de depuración del código y por compartir conmigo sus habilidades y conocimientos de programación.

Además, quiero agradecer al profesor Oliver Brüls que me haya ofrecido la posibilidad de explorar nuevos campos de la dinámica multicuerpo durante mi estancia en el Multibody & Mechatronic Systems Laboratory de la University of Liège. Realmente valoro su apoyo y todo el material y lecciones que me ha facilitado. También me gustaría agradecer a Juliano Todesco su inestimable colaboración, y a Facundo Cosimo por hacer mi estancia más agradable.

Finalmente, me gustaría agradecer a mi familia su apoyo incondicional, especialmente a mis padres Rosa María y Manuel. Ellos me han enseñado más que cualquier otra persona, y todavía me queda mucho que aprender de ellos. No me puedo olvidar de aquellos que están conmigo, Clara, Moncho, Rubén, abuela Clara..., y de aquellos que nos dejaron pero que están todavía muy presentes en nuestras vidas.

# Agradecementos

O salto do sector privado á investigación nunha universidade pública parece moi difícil actualmente, especiamente despois do duro proceso de alcadar un traballo acorde coa túa cualificación. No meu caso, a decisión foi moi meditada, pero non me arrepinto da selección da carrera de investigación, na cal desexo estar involucrado nos próximos anos. Neste sentido, necesito agradecer a Alfredo del Caño Gochi que me conectara co Laboratorio de Ingeniería Mecánica (LIM) e por informarme da oportunidade de desenvolver unha tese doutoral neste marabilloso grupo.

Por riba de todo, gustaríame expresar a miña máis sincera gratitude cara ó meu director Daniel Dopico Dopico pola súa axuda e dirección durante este periodo. Sen a súa paciencia e apoio, este traballo sería imposible. Ademais, debo destacar que o meu campo de especialidade era Enxeñaría Electrónica, pero gracias ás súas razonadas e detalladas explicacións non tiven demasiados problemas na miña introducción na dinámica multicorpo.

Adicionalmente, extendo este agradecemento a tódolos membros do LIM polo extremadamente bo ambiente de traballo, e especialmente a Alberto Luaces Fernández polo seu apoio durante o proceso de depuración do código e por compartir comigo as súas habilidades e coñecementos de programación.

Ademais, quero agradecer ó profesor Oliver Brüls que me ofrecese a posibilidade de explorar novos campos da dinámica multicorpo durante a miña estancia no Multibody & Mechatronic Systems Laboratory da University of Liège. Realmente valoro o seu apoio e todo o material e leccións que me facilitou. Tamén me gustaría agradecer a Juliano Todesco a súa inestimable colaboración, e a Facundo Cosimo por facer a miña estancia máis agradable.

Finalmente, gustaríame agradecer á miña familia o seu apoio incondicional, especialmente a meus pais Rosa María e Manuel. Eles ensináronme máis que calquera outra persoa, e aínda me queda moito que aprender deles. Non me podo olvidar de aqueles que están comigo, Clara, Moncho, Rubén, abuela Clara..., e de aqueles que nos deixaron pero que están aínda moi presentes nas nosas vidas.

# Abstract

Sensitivity analysis of the dynamics of multibody systems is an extraordinarily useful tool for the design optimization and optimal control problems. In this thesis document, the sensitivity analysis of joint-coordinate formulations is studied using purely analytical differentiation methods. The recursive foundations of joint coordinate modeling are reviewed and extended to support the definition of recursive kinematic relations and accumulation schemes with an arbitrary selection of reference points.

Two different solutions of the equations of motion of unconstrained open-loop systems are developed, leading to a semi-recursive and a fully-recursive approach. Moreover, the dynamics of constrained multibody systems is described using Matrix R and ALI3-P formulations combined with semi-recursive methods.

The main contribution of this thesis consists on the analytical development of each derivative required in the sensitivity analysis of joint-coordinate formulations. As a result, two sensitivity analyses for unconstrained open-loop systems and six sensitivity formulations for constrained systems are achieved. All dynamic and sensitivity formulations have been implemented in the multibody library MBSLIM as general formulations.

# Resumen

El análisis de sensibilidad de la dinámica de sistemas multicuerpo es una herramienta extraordinariamente útil para problemas de optimización de diseño y control óptimo. En el presente documento de tesis se estudia el análisis de sensibilidad de formulaciones en coordenadas de par usando métodos de diferenciación puramente analíticos. Los fundamentos recursivos de los modelos en coordenadas de par son revisados y extendidos para soportar la definición de relaciones cinemáticas recursivas y esquemas de acumulación para una selección arbitraria de puntos de referencia.

Se desarrollan dos soluciones diferentes de las ecuaciones del movimiento para sistemas de cadena abierta no restringidos que conducen a formulaciones semi-recursivas y totalmente recursivas. Además, se describe la dinámica de sistemas restringidos mediante formulaciones de Matriz R y ALI3-P combinadas con métodos semi-recursivos.

La principal contribución de esta tesis consiste en el desarrollo analítico de cada derivada requerida en el análisis de sensibilidad de formulaciones en coordenadas de par. Como resultado se consiguen dos análisis de sensibilidad para cadenas abiertas sin restricciones y seis formulaciones de sensibilidad de sistemas restringidos. Todas las formulaciones dinámicas y de sensibilidad han sido implementadas en la librería multicuerpo MSBLIM como formulaciones generales.

# Resumo

A análise de sensibilidade da dinámica de sistemas multicorpo é unha ferramenta extraordinariamente útil para problemas de optimización de deseño e control óptimo. No presente documento de tese estúdase a análise de sensibilidade de formulacións en coordenadas de par usando métodos de diferenciación puramente analíticos. Os fundamentos recursivos dos modelos en coordenadas de par son revisados e estendidos para soportar a definición de relacións cinemáticas recursivas e esquemas de acumulación para unha selección arbitraria de puntos de referencia.

Desenvólvense dúas solucións diferentes das ecuacións do movemento para sistemas de cadea aberta non restrinxidos que conducen a formulacións semi-recursivas e totalmente recursivas. Ademais, descríbese a dinámica de sistemas restrinxidos mediante formulacións de Matriz R y ALI3-P combinadas con métodos semi-recursivos.

A principal contribución desta tese consiste no desenvolvemento analítico de cada derivada requirida na análisee de sensibilidade de formulacións en coordenadas de par. Como resultado, conséguense dúas análises de sensibilidade para cadeas abertas sen restricións e seis formulacións de sensibilidade para sistemas restrinxidos. Todas as formulacións dinámicas e de sensibilidade foron implementadas na librería multicorpo MSBLIM como formulacións xerais.

# Contents

# List of Figures

# List of Tables

# Acronyms

**AD** Automatic (or Algorithmic) Differentiation.

**ALI3-P** Augmented Lagrangian index-3 with Projections.

**AVM** Adjoint Variable Method.

**CAVM** Continuous Adjoint Variable Method.

**CoM** Center of Mass.

**DAE** Differential Algebraic Equation.

**DAVM** Discrete Adjoint Variable Method.

**DDM** Direct Differentiation Method.

**DoF** Degrees of Freedom.

**EoM** Equations of Motion.

**LIM** Laboratorio de Ingeniería Mecánica.

**MBD** Multibody Dynamics.

**MBS** Multibody Systems.

**MBSLIM** Multibody Systems in Laboratorio de Ingeniería Mecánica.

**MCAE** Mechanical Computer Aided Engineering.

**ND** Numerical Differentiation.

**NR** Newton-Raphson.

**ODE** Ordinary Differential Equation.

**SQP** Sequential Quadratic Programming.

**TRR** Trust Region Reflective.

# Chapter 1

# Introduction

## 1.1 Motivation

The steady increase in the capabilities of personal computers during last decades has made it possible to address the solution of complex numerical problems in several areas of knowledge, with a special impact on the dynamic simulation of multibody systems (MBS). Computational improvements have changed the objectives of mechanical engineers in this field, moving the focus from efficient real-time dynamic simulations to the optimization of MBS.

Two main problems can be distinguished in the optimization of MBS, depending on the objective of the optimization: optimal design, where the topology, dimensions, masses and inertia of the mechanism are sought in order to minimize a given objective function; and optimal control, where the aim is to obtain a sequence of controls over time, usually forces, such as a given objective function is minimum. A sensitivity analysis is mandatory if any of those problems are faced with gradient based optimization methods, being the requirements of optimal control especially restrictive in terms of computational time in some particular applications.

Recently, some multibody research works have been devoted to the formulation of new fast and accurate techniques to perform the sensitivity analysis of the dynamic response of a MBS. The result of this sensitivity analysis offers a measure of the variation of the behavior of a model with respect to a set of parameters, which is essential for any gradient-based optimization. The execution of this analysis could involve high complexity and a huge computational effort, being this one of the reasons why this calculation is not generally supported by commercial analysis software.

There are different differentiation techniques which can be used to obtain the sensitivity analysis of a system, from numerical methods based on perturbations, to complex analytical methods based on the differentiation of the dynamic expressions, going through automatic differentiation. In general, analytical approaches are usually the fastest and most accurate, but due to their complexity, they have not been generally implemented in multibody packages nor used in Mechanical Computer Aided Engineering programs (MCAE).

The computational effort dedicated to sensitivity analysis is related to the multibody model that describe the mechanism in study, as well as to the formulation used to obtain the dynamic response. Models based on relative coordinates usually lead to the fastest dynamic simulations, although they involve more complex computations than other models.

During the past few years, the Laboratorio de Ingeniería Mecánica (LIM) of the University of A Coruña has been studying, developing, implementing and testing new analytical computations for the sensitivity analysis of several well-known multibody formulations [1–4], all programmed in the general multibody simulation library MBSLIM [5] for natural coordinates models. The algorithms obtained have been successfully used in the optimization of the design of different multibody models, and into other optimal control problems.

The sensitivity analysis of the dynamics of relative coordinate models have not been generally and systematically studied yet, despite frequently delivering the best dynamic performance in terms of computational cost. The present work intends to shed light on the generality and validity of relative coordinate modeling for both dynamic and sensitivity analyses.

## 1.2 State of the art

### 1.2.1 Multibody modeling

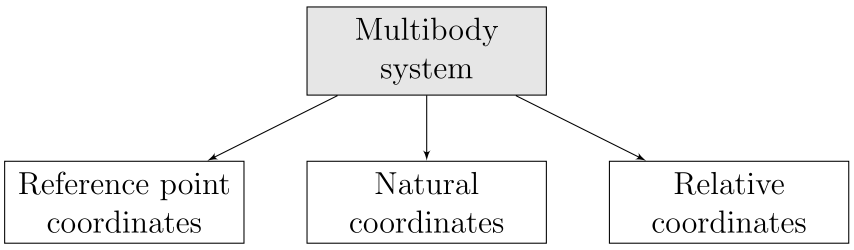


Figure 1.1: Types of coordinates for MBS modeling.

The possibilities in the field of dynamic simulation of MBS are multiple and diverse. A first classification can be done regarding the parameterization of the multibody model. In this sense, three major categories can be identified according to the coordinates which define the motion: reference point coordinates, natural coordinates and relative coordinates.

Reference point coordinates (also referred to as point of reference or Cartesian coordinates) arise from the definition of the configuration of a rigid body by means of the position and orientation of a reference frame attached to the body with respect to an inertial reference frame. Kinematic joints between bodies are imposed through kinematic constraints. This approach is probably the most widely spread in the multibody community and it is the foundation of different commercial multibody packages such as Automatic Dynamic Analysis of Mechanical Systems, commonly known as ADAMS [6, 7].

Implementations of reference point coordinate models differ on how the orientation of the body-attached frames is measured. Some authors use a minimal parameterization with 3 angles along with an additional algorithm to handle singularities, like in ADAMS. Others avoid the singularities by resorting to non-minimal dependent parameters, such as the 9 components of the direction cosine matrix, to 4 dependent parameters or to quaternions [8]. Nowadays, Euler parameters are perhaps the most extended option in this sense. A third possibility is related to the use of Lie Groups as a means to avoid global parameterization of orientations [9].

Garcia de Jalón et al. introduced in [10] a new set of coordinates that has changed the classical perspective of multibody modeling. These new so called natural coordinates stem from the previous works by Garcia de Jalón et al. [11, 12] and Serna et al. [13] in which basic coordinates were developed as an evolution of reference point coordinates. Basic coordinates exploit the kinematic joint conditions, such as the sharing of a point or a vector between bodies, in order to reduce the total number of coordinates and constraints of the system. Natural coordinates widen the advantages of basic coordinates by parameterizing the motion with Cartesian coordinates of points and unitary vectors. The great variety of works based on these coordinates since 1986 evidences the advantages of this modeling technique. In [14], García de Jalón reviews the most important contributions in the field of natural coordinates modeling until 2007. However, the world keeps spinning, and since then, we can remark several new interesting works referred to Real-time simulation [15, 16], performance assessment of different formulations [17] or flexible multibody simulation [18, 19], among others.

Relative coordinates have been traditionally used to describe the kinematic motion of open-loop systems due to the reduced number of coordinates generated and the direct identification of joint coordinates and degrees of freedom. Many authors date the inception of relative coordinate modeling applied to multibody dynamics in the late 70's and early 80's, reaching maturity with the acknowledged work of Featherstone in 1983 [20]. Nevertheless, they had been used beforehand in different multibody programs such as IMP (Integrated Mechanisms Program) [21], DAMN (Dynamic Analysis of Mechanical Networks) [22] or DYMAC (Dynamics of Machinery) [23, 24], to mention a few.

In the middle of the 70's, multibody researchers centered their efforts in the development of efficient techniques to evaluate the dynamics of rigid MBS. Stepanenko and Vukobratovic [25] presented a method for the evaluation of the inverse dynamics applied to anthropomorphic manipulators, which was extended and reformulated in subsequent works [26, 27]. Vukobratovic and collaborators derived the equations of motion (EoM) from D'Alembert's principle and generated a recursive algorithm for the inverse dynamics problem. In 1979, a work from Waters [28] in which the generalized forces vector in open-loop relative coordinate models was calculated as a recursion from the tips to the base body came to light. At the same time, Armstrong [29] developed a recursive method for open-loop systems with spherical joints of $O(n)$ complexity, which means that the number of operations grows linearly with the number of rigid bodies.

In parallel, Luh et al. developed in [30] an efficient method for the recursive solution of the inverse dynamic problem in terms of the Newton-Euler equations. Walker and Orin [31] reviewed some contributions in the field of recursive inverse dynamics and used them to build different methods for the forward solution of the equations of motion. The core of this work relies on method 3, in which a recursive method for the composition of the inertia matrix is outlined. Hollerbach [32] also developed a different $O(n)$ recursive method for the solution of the inverse dynamics problem using the Lagrangian method instead of the Newton-Euler equations.

Those previous works can be gathered into a group of fully-recursive formulations, in which both the accumulation processes and the solution of the equations of motion are recursively executed. A different approach consists in the generation of the set of equations by means of recursive evaluations and accumulations but with a global (instead of recursive) solution of the whole system. Jerkovsky is considered as the pioneer on these semi-recursive methods thanks to his work [33]. Wherein, Jerkovsky presented a detailed method for the generation of the equations of motion in a "primitive" form in terms of reference points and angular magnitudes for each body regarded as free, and then he imposes a velocity transformation to include the constraints required by joints and closed-loops. Moreover, this work delivers a review on the most prominent new studies on the dynamics of MBS up to its publication date.

Those and other papers established the foundations for the method of articulated-body inertias, firstly introduced by Featherstone in 1983 [20]. The articulated-body inertia method sets forth a technique to accumulate masses and inertia of groups of bodies connected through kinematic joints. Although the method was originally developed for revolute and prismatic joints, it has been demonstrated to be applicable to joints with more than one degree of freedom (DoF) [34]. The relevance of Featherstone's work relies on the systematic procedure presented, on the simplicity of the accumulation procedures with straightforward matrix expressions and on the general application to any rigid multibody model. This publication constitutes one of the basis of the present thesis document.

The interest on relative coordinate formulations and recursive methods does not decay in the following years. Bae and Haug presented the trilogy of papers devoted to the fully-recursive solution of the dynamics of any MBS including closed loops in a general fashion [35–37]. This formalism was extended also to flexible multibody systems in posterior works [38–40]. Kim and Vanderploeg [41] explored the relation between reference point coordinates (therein called Cartesian coordinates) and relative coordinates in the framework of the semi-recursive method based on velocity transformations. In 1993, Nikravesh published a work [42] on the semi-recursive generation and solution of a general closed-loop system using a second velocity transformation related to cut-joint constraints. One peculiarity of this paper is the use of the term"natural coordinates" as equivalent to "relative coordinates".

In 1991, Jain devoted his work [43] to review and unify the different recursive algorithms published until that moment for open-loop systems. In this publication, Jain classifies recursive algorithms into $O(n)$, $O(n^2)$ and $O(n^3)$ methods depending on the number of elemental operations required by each one.

Later on, Avello et al. [44] presented a semi-recursive method based on velocity projections for Real-time simulation applying parallelization techniques.

Jiménez developed in his PhD thesis [45] a deep study on relative coordinate formulations, referring two different compositions of the equations of motion based on a semi-recursive approach using two sets of reference points, and a third formulation referred to a fully-recursive approach. The two sets of reference points considered in this thesis were the center of mass (CoM) of each body, which delivers the so called formulation RTdyn0, and the global origin of the inertial frame, which leads to the formulation that Jiménez identified as RTdyn1. This notation has been used by other authors [46], and will be maintained in the present thesis document. Moreover, Jiménez introduced two methods to solve the constrained dynamics of closed-loop systems, which are an independent coordinates formulation based on velocity transformations and a penalty approach.

In 1997, Negrut et al. [47] studied sparsity in relative coordinate formulations paying special attention to the effect of the spanning tree structure into the dynamic expressions, to the factorization of the EoM and to parallelization opportunities in semi-recursive algorithms. Bae et al. explored in [48] the composition of kinematic and dynamic magnitudes in recursive methods based on velocity transformations, and the results obtained in their work were later expanded to flexible bodies in [49]. Other authors continued the pursuit of Real-time simulation by combining recursive methods with new techniques such as the decomposition of complex multibody models into simpler subsystems [50], by the reformulation of loop-closure constraints including virtual bodies in order to obtain $O(n+m)$ methods [51, 52] or by fully-recursive formulations with a penalized enforcement of constraints [53].

The listed works prove that recursive methods offer a general solution for any multibody system by reducing it to an open-loop system (or spanning tree) subjected to a series of kinematic constraints. The solution of the constrained system can be reached using one of the multiple schemes applied to natural and reference point coordinates. In that regard, semi-recursive methods have been combined with the Matrix R formulation (also referred to as coordinate partitioning method) in [45, 46, 54–56] and more recently in [57], with the penalty approach in [45, 46] or with the Augmented Lagrangian index-3 formulation with velocity and acceleration projections (ALI3-P) in [58–61]. This last formulation has shown to outperform the previous ones in different numerical problems [17], but as it was stated in [53], the performance of each formulation and coordinate modeling is case dependent.

### 1.2.2 Sensitivity analysis

The sensitivity analysis of a multibody system dynamics (MBD) entails the determination of the effect of a set of parameters, such as masses, inertia, coefficients of forces or geometrical specifications, on the dynamic response. The sensitivity analysis of MBS can be categorized according to the differentiation procedure used to differentiate each term involved in the equations of motion.

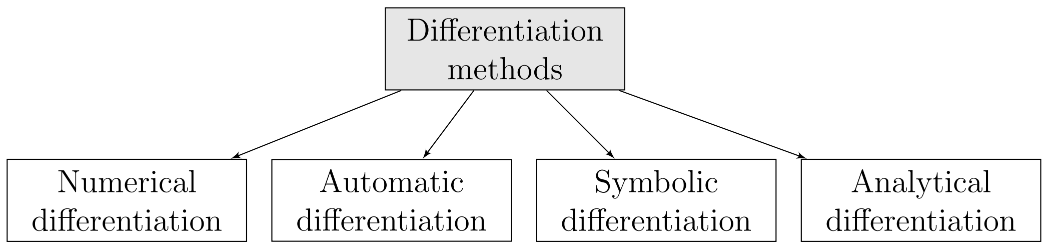

Differentiation methods
Numerical differentiation
Automatic differentiation
Symbolic differentiation
Analytical differentiation

Figure 1.2: Differentiation techniques for the sensitivity analysis of MBS.

The simplest differentiation method consists in numerical differentiation (ND), which is based on the numerical perturbation of the parameters in order to approximate the derivative of the objective function by means of its variation using a finite difference scheme or similar. This method is perhaps the most widely extended, not only in multibody dynamics but in several other different fields, since it offers a result of an objective function derivative with a minimum implementation effort and with a scheme applicable to any numerical problem. However, this technique suffers from two issues: first, it is extremely dependent on the magnitude of the perturbations used which yields the so called “step-size dilemma” [62], i.e. high perturbations lead to bad results and low perturbations to the incorporation of numerical errors; and second, the computational effort is directly proportional to the number of parameters. Despite its problems, this method has been used combined with semi-analytical differentiation in works such as [63] or [64], and it is commonly used as test method for analytical implementations.

The problem of selecting the most appropriate perturbation value minimizing both truncation and subtraction cancellation errors can be dodged thanks to the use of complex variables [62]. The use of complex algebra allows very small perturbations without incurring into errors related to the numerical ill conditioning of subtractions. Despite these properties, this method has not been generally applied to the differentiation of MBS due to the need to extend the definition of any kinematic or dynamic magnitude to complex variables.

A second technique used in the sensitivity analysis of MBS, whose presence in sensitivity problems has exploded in the past few years thanks to the development and expansion of new math packages, is the automatic differentiation method (AD) (also called algorithmic differentiation method). Automatic differentiation is founded on the decomposition of complex computations into elemental operations with known direct analytical expressions for their derivatives, hence the derivatives of the original complex function can be computed automatically from the elemental ones using derivative procedures as the chain rule. This method offers high accuracy with low computational expense. Even though its implementation is substantially more complicated than numerical methods, the fact that the derivatives of elemental operations are not managed directly, avoiding possible errors, make it a suitable method for obtaining accurate derivatives with a low programming effort [65].

Several automatic differentiation libraries have been developed in the last decades for different languages, such as ADIFOR [66] for Fortran 77 or ADOL-C [67] and ADIC2 [68] for C and C++. It is also worth to mention the automatic differentiation capabilities of the Eigen library, such as the AutoDiff package based on a data type replacement. The list of automatic differentiation tools is much longer and nowadays is still increasing. The listed libraries have been successfully used in the sensitivity analysis of the dynamics of MBS in different cases. Dürrbaum et al. compared the performance of ADOL-C against a symbolic differentiation software in [69], giving as result a better performance of the symbolic program. Callejo and collaborators explored the automatic differentiation for the dynamics of MBS in different works [65, 70, 71]. Ambrosio et al. applied ADIFOR for the optimization of flexible MBS in [72]. The list of works comprising automatic differentiation is steady growing, specially in those problems where analytical differentiation is unmanageable.

The third possibility consists in symbolic differentiation, supported by symbolic calculus. Nowadays, this technique is more common for comparison and validation purposes of analytical derivatives [69], but it still is a frequently used method in the multibody community. Symbolic differentiation is usually a computationally expensive calculation, therefore it is commonly harnessed as a preprocessing tool whose results are implemented as analytical functions.

The fourth differentiation path used in sensitivity analysis consists in the analytical differentiation of the EoM describing the dynamic response of the system. Analytical approaches are usually the best option to achieve fast and accurate solutions, since they use the exact analytical derivatives of the original expressions. The method requires a huge implementation effort owing to that the derivative of every term included in the dynamics have to be developed, programmed and tested. Nevertheless, the enormous relevance of sensitivity accuracy and efficiency in optimization problems justifies the implementation effort. In fact, the recent increase of publications related to analytical sensitivity analyses proves this. Our approach is to implement analytical for the core but we do not close the possibility of using other techniques for some non-critical parts of the software.

The sensitivity analysis of the EoM of a multibody system can be accomplished following two different approaches, which are the direct differentiation method (DDM) and the adjoint variable method (AVM). The application of the DDM to the EoM of a MBS delivers a system of equations in which the unknowns are the sensitivities of the variables of the original dynamic problem. The simplicity of the DDM has made many authors to resort to this method in problems involving a sensitivity analysis, which converts it in the most spread option in the multibody community.

An increase in the number of sensitivity parameters in a sensitivity analysis entails a loss of efficiency of the DDM because the number of systems of equations to be solved is $p$ times the number of systems solved in the dynamics, being $p$ the number of parameters. The AVM solves this problem by the reformulation of the sensitivity analysis through the definition of a new Lagrangian function composed of the objective function and the dynamic EoM pre-multiplied by a set of new variables, namely the adjoint variables. Thanks to this transformation, the number of systems of equations is independent of the number of parameters, which makes it the ideal method for highly parameterized sensitivity problems.

The theory behind sensitivity analysis was mature previous to the beginning of its application to MBS, which justifies that both DDM and AVM have been applied from the very first dynamic sensitivity analysis. In the late 70's and early 80's, Haug, Arora and Rousselet made public some works related to the sensitivity analysis of structural problems [73–77]. Despite the gap between structural and multibody dynamics, these publications are regarded as precursors and have paved the way to the sensitivity analysis of the dynamics of rigid and flexible MBS.

Later on, several sensitivity analyses of the dynamics of MBS came to light. Some of the first analytic works on the sensitivity analysis of the dynamics of MBS were developed by Haug's group in the middle 80's [78, 79]. The main difference among those works is referred to the type of mechanisms considered, but the theoretical basis is equivalent. In those works, the authors consider the sensitivity analysis of a set of first order differential equations as an introductory step to the sensitivity analysis of the classical form of the index-3 Lagrange EoM, corresponding to a second order DAE system. However, both approaches (first and second order differential-equation systems) are equally important in multibody dynamics, since several multibody formulations can be categorized in one of these two groups. This work, therefore, founded the base for the sensitivity analysis of any dynamic formulation encompassing all the spectrum of continuous sensitivity analysis, i.e. from the DDM to the AVM.

A novel approach for the solution of the sensitivity problem based on the coordinate partitioning method presented in [80] was developed by Mani and Haug in [81]. In this work, the coordinate partitioning method is applied to the direct and adjoint sensitivity equations previously obtained from an index-1 Lagrange formulation by means of the singular value decomposition method (SVD). The option of developing a different adjoint formulation from the basis of the coordinate partitioning method applied to the dynamics is unexplored there.

In 1990, Ashrafiuon and Mani developed a study [82] that is closely related to the subject of the current thesis, which is the sensitivity analysis of a semi-recursive formulation. In this case, because "a general purpose computer program for design sensitivity analysis of spatial systems is extremely difficult to develop, too complicated, and computationally inefficient" [82], the authors computed the sensitivity analysis of the index-1 Lagrange formulation studied through symbolic differentiation. Despite the author's arguments, the present document thesis will prove that this general purpose computer program for the sensitivity analysis of the dynamics of relative coordinate models, though complex, is achievable and could be highly efficient.

In 1992, Bestle ans Seybold [83] published a work on the adjoint sensitivity analysis of an index-1 Lagrange formulation considering an integral objective function whose upper time limit depends on the final states. The DDM is omitted in this case as the authors regarded the AVM more computational efficient for the applications they were interested in. In 1997, Dias and Pereira explored the direct sensitivity analysis of an index-1 formulation applied to flexible multibody systems in the sense of the component mode synthesis method [84]. In this work, the authors resorted to symbolic and numerical differentiation [85].

In the meantime, Pagalday and Avello investigated in [86] the sensitivity analysis of the EoM of a MBS in the framework of design optimization. They start from three different solutions for the constrained dynamics: the Lagrangian formulation combined with Baumgarte stabilization; the penalty formulation, which emerges from the substitution of the Lagrange multipliers of the former formulation by a penalized term; and the so called Augmented Lagrangian penalty formulations, which combine the solution of the two previous formulations by means of the inclusion of an update loop of the Lagrange multipliers with a penalized term. The DDM applied to these three formulations is presented in this publication along with a series of numerical examples executed with symbolic differentiation tools.

One of the most challenging stages in the generation of an analytical sensitivity analysis is referred to the assessment of the derivatives of each kinematic and dynamic term with respect the states and the parameters. The performance of this process will strongly determine the global accuracy and efficiency of the method. The number of works treating this subject in multibody dynamics is relatively reduced. We can highlight the work [87], in which Serban and Haug studied the differentiation of the dynamics of reference point coordinate models in which orientation has been parameterized through Euler parameters. In the same line, Wang et al. presented in [88] a detailed description of the required terms for the direct sensitivity analysis of MBS modeled with reference point coordinates and with dynamics described by a set of EoM reduced to a “state-space” ODE system in the framework of the assessment of an implicit numerical integrator on stiff vehicle models..

In 2000, Schulz and Brauchli set forth two sensitivity-analysis schemes for the dynamics of MBS with non-smooth collisions [89]. The main contribution of this work is the description of a method to account for discontinuities in continuous by definition differentiation schemes such as the DDM and AVM (considering the MBD as continuous expressions). Other works [90, 91] zeroed in on the implementation of the sensitivity calculations [90], developing new algorithms (DASPK3.0 and DASPKADJOINT) for the efficient sensitivity analysis of any up to index-2 DAE (using the DDM and AVM respectively).

In [92], the AVM applied to implicit, semi-explicit and explicit ODE along with index-1 and index-2 DAE is studied in terms of its derivation procedure and the stability of the resulting adjoint equations. The stability of the adjoint equations has been also studied by Schaffer for index-3 DAE in [93, 94], who also devoted the work [95] to the deep study of the AVM applied to the Hiller–Anantharaman stabilized index-1.

Sensitivity analysis methods can be further classified according to the order between differentiation and discretization. Frequently, differentiate-then-discretize techniques are denoted as continuous methods, while discretize-then-differentiate approaches are identified as discrete methods. In DDM, continuous and discrete methods normally yield equivalent expressions, but in AVM, the composition of the adjoint equations could be significantly different. Studies on the discrete adjoint variable method (DAVM) can be found in [96], for Runge-Kutta integrators; [97] for the implicit Hilber–Hughes–Taylor (HHT) numerical integrator; or [98] for the generalized-alpha integrator.

Despite the great advantages of relative coordinate modeling, the number of works related to the sensitivity analysis of relative coordinate dynamic formulations is really reduced. In addition to the commented work by Ashrafiuon [82], Anderson and Hsu explored this path with the sensitivity analysis of fully-recursive formulations considering open-loop chains in [99, 100]. Those developments were later used in the recursive direct sensitivity analysis of flexible MBS [101]. Later, Gutiérrez et al. explored in [102] the DDM of semi-recursive formulations using the independent-coordinate Maggi's formulation. In that work, analytical, numerical and automatic differentiation are compared for two numerical experiments modeled through a semi-recursive approach based on the selection of the global origin of coordinates as reference point for each body.

Nowadays, researches on the sensitivity analysis of the dynamics of MBS are more frequent and they study topics related to recent developments in the field of multibody dynamics, such as the use of Lie-group algebra, non-smooth contact, flexible multibody simulation or Real-time dynamics, but also other issues as second-order sensitivities or parallel computation. Sonneville and Brüls recently published a work [103] referred to the sensitivity analysis of rigid MBS modeled with reference-point coordinates under a Lie group formalism. Other authors zeroed in on solving classical issues of continuous analytical approaches, such as the inclusion of discontinuities in the continuous AVM. In this regard, Corner et al. [104] consistently included inequality constraints emerging from non-smooth contact in a continuous adjoint variable formalism though jump sensitivity matrices. Serban studied in [105] second order sensitivities of multibody dynamics described by an ODE system though parallel computing combining analytical and automatic differentiation methods.

Recently reviewed formulations for sensitivity analysis are the Hamiltonian index-2 DAE [106], independent-coordinate Matrix R formulations [1, 107], the ALI3-P formulation [3, 4], the index-3 Lagrange formulation, the index-1 Lagrange approach and the penalty formulation [2].

Due to the complexity stemmed from the differentiation of dynamic magnitudes required in any analytical sensitivity analysis, relative coordinate models are usually dodged so as to reduce the implementation effort. For the best of the author's knowledge, the purely analytical sensitivity analysis of general semi-recursive methods for constrained multibody systems modeled in joint coordinates has not been addressed yet. In fact, the analytical differentiation complexity of this type of problems has been circumvented by several authors by combining analytical and automatic differentiation procedures. In the present thesis document, the sensitivity analysis equations of semi-recursive methods will be described using dependent and independent coordinate formulations in a general fashion for a wide range of kinematic joints and for any topology of the mechanism using the analytical differentiation method.

### 1.2.3 Optimization

"Optimization is an important tool in decision science and in the analysis of physical systems", [108]. It has attracted the attention of researchers for years in diverse fields of knowledge, and it has been gaining relevance in the field of MBS since the appearance of the first multibody numerical formulations.

The first works involving the optimization of MBS were focused in the kinematic synthesis of mechanisms in order to carry out specific tasks or maneuvers. In [109], a review of the most prominent works on kinematic synthesis is presented. These first works paved the way for the application of optimization techniques to the dynamics of MBS.

The optimization of the dynamic response of a MBS seeks to obtain an optimal value of a group of parameters which minimizes a given objective function dependent on this dynamic performance. In this sense, dynamic optimization is significantly more complicated than kinematic synthesis since the type of parameters affecting the system is greater and the dynamic expressions are more demanding.

Optimization problems are usually approached using the derivatives of the objective function as a means to compute the optimal increments in the set of parameters by the optimization algorithm. This is commonly known as gradient-based optimization. The evaluation of the gradient of a given function dependent on the dynamics of a MBS requires a sensitivity analysis. In fact, the inception of most of the sensitivity studies commented in the previous section was due to optimization problems.

Regarding the target of the optimization, two optimization problems can be distinguished: design optimization and optimal control. Design optimization is referred to the problem of discovering what are the optimal values of some constant design magnitudes that minimize a given objective function. The seek of optimal control is as old as design optimization, but in this case the aim of the problem is to determine the set of controls that deliver an optimal dynamic performance of a MBS.

The number of works referred to optimal control and design is unfathomable and still growing nowadays, since they are still open problems and highly case-dependent.

One of the most resorted books in optimal design is [110] (or one of its editions). It constitutes an educational approach to design optimization and establishes the fundamentals to transform any design problem into an optimization problem. It also reviews the solution of constrained and unconstrained optimizations through numerical examples. Out of the field of mechanical systems, several optimization books can be pointed out, such as the monographs from Nocedal and Wright [108], Fletcher [111] or Betts [112].

## 1.3 Objectives

The aim of the present thesis is the study of recursive formulations and the development of their analytical sensitivity analyses. This general objective can be decomposed into a series of intermediate objectives listed below:

- Review and generalization of recursive multibody dynamic formulations, including fully-recursive and semi-recursive formulations.
- Implementation of recursive methods for the dynamic analysis of multibody models in MBSLIM.
- Development of sensitivity analysis of recursive multibody formulations.
- Study of continuous and discrete sensitivity analysis approaches.
- Development, implementation and solution of different industrial problems of special interest.
- Performance assessment of the sensitivity analysis of recursive methods.
- Application of recursive methods to optimization problems.

## 1.4 Thesis structure

The present document has been divided into 8 chapters:

**Chapter 1** is devoted to the introduction including the motivation, the state of the art and objectives of the thesis, offering a background of the sensitivity analysis of multibody models.

**Chapter 2** presents the kinematic relations between bodies connected by different types of joints and their composition in order to define the kinematic relations between all the bodies of a kinematic system. These expressions are defined in terms of an arbitrary reference point, and then particularized to two cases of study: RTdyn0 and RTdyn1. Besides, two different accumulation procedures are detailed, leading to a semi-recursive and a fully-recursive method. The solution of unconstrained multibody systems with these two accumulations is unfolded in this chapter as well.

In **Chapter 3** the semi-recursive equations are combined with a set of kinematic constraints. First, the kinematic problems of initial position, velocity and acceleration in relative coordinates are introduced. Second, the dynamic analysis of constrained topological systems is accomplished with an independent-coordinate formulation, namely semi-recursive Matrix R. Third, a dependent-coordinate formulation based on an augmented Lagrangian index-3 scheme with projections is introduced. At last, the computation of the derivatives of the constraint vector required in the previous formulations is presented.

**Chapter 4** introduces the sensitivity analysis of unconstrained open-loop systems for two different schemes: the semi-recursive approach and the fully-recursive method. In the second part of the chapter, the assessment of the derivative of each term related to masses, forces and the kinematics of the system is introduced and particularized for RTdyn0 and RTdyn1. This chapter exposes the derivatives of the expressions introduced in chapter 2.

**Chapter 5** develops the sensitivity analysis of the kinematic and dynamic problems introduced in chapter 3 with two different techniques: the DDM and the AVM. In the sensitivity analysis of the semi-recursive ALI3-P formulation, two different adjoint methods are presented following a continuous and a discrete approach.

**Chapter 6** outlines the implementation work accomplished, emphasizing some algorithmic optimizations which yield a significant reduction on the computational expense of the described methods.

**Chapter 7** gathers the set of numerical experiments with which the previous dynamic and sensitivity formulations have been proved. Each numerical experiment begins with the description of the model and the maneuver executed, followed by the dynamic results and then the solution of the sensitivity analysis with different formulations, zeroing in on the accuracy and computational time. Moreover, some optimization problems are accomplished employing some of the sensitivity analyses described in these document.

**Chapter 8** closes the document with the conclusions and the future research lines opened by this work.

Furthermore, additional appendices are included in order to enhance the comprehension of different parts of the document. They encompass intermediate calculations and mathematical properties of some operators used along the thesis.

# Chapter 2

# Topological formulations for the dynamics of open-loop systems

Relative coordinate models are defined by a set of kinematic joints in a particular sequence composing a kinematic chain. A kinematic joint is defined by R. Featherstone as “a kinematic constraint between two bodies”, [113], being the constraints limitations in the relative motion. In open-loop systems modeled with minimal relative coordinates, the relative motion is implicitly defined by joint relations, hence no additional constraint equations are needed. Considering also the sequence of kinematic joints, the kinematics of each body of the multibody system can be completely described.

The EoM of a relative coordinate model can be obtained from the combination of its kinematic expressions with the mass and inertia of each body and the forces to which the mechanism is subjected. The methods for the calculation of the dynamics of relative coordinate models presented in this chapter are topological recursive methods based on the articulated inertia method of [34] with the variations and notation employed in [46] and the particular expressions derived in [54]. An improved version of the formulation is developed here for open-loop systems not attached to the ground.

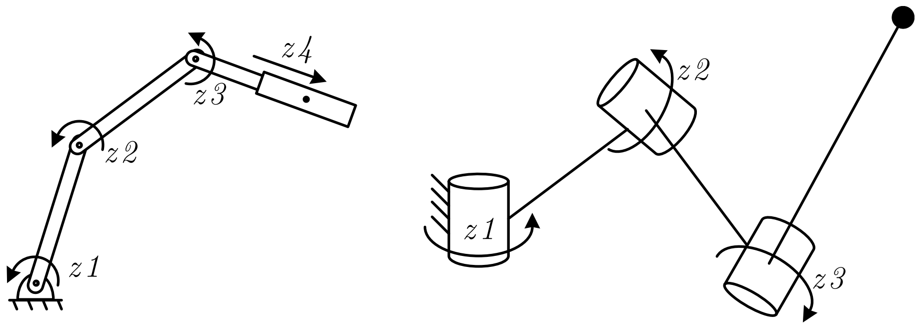


Figure 2.1: Open chain systems

The current chapter is divided in three sections: first, seven types of joints are studied in terms of the relative motion allowed and the kinematic relations between bodies; in the second section, a series of joint-dependent kinematic recursive relations is developed in order to establish a method for the sequential evaluation of the kinematics of any open-loop system; the third section is devoted to the dynamics of open-loop systems using a fully-recursive and a semi-recursive approach. Additionally, two particular formulations are developed taking advantage of the effects that the selection of reference points has on the semi-recursive equations of motion.

## 2.1 Kinematics

Relative coordinate modeling usually constitutes the most natural and efficient method to describe the kinematics of a multibody system. However, despite the reduced number of coordinates, the generation of the equations of motion is not direct and other set of intermediate coordinates (as Cartesian coordinates) is needed.

The use of these intermediate coordinates encompasses a double kinematic problem that has to be addressed both in kinematic and dynamic analyses: first, it is necessary to calculate positions, velocities and accelerations of the intermediate Cartesian coordinates from the relative coordinates in order to generate the EoM; second, a coordinate transformation takes the equations from Cartesian to relative coordinates, which is carried out by means of recursive kinematic relations based on the topology of the system. In this section, the first problem is described.

In the following derivations, bodies will be numbered with superscripts $i-1$, $i$, $i+1$, etc. while points (vectors) belonging to a body will be named with subscripts $j-1$, $j$, $j+1$, etc. and, if needed, with the superscript of the body to which it belongs. For example, point $\mathbf{r}_j$ is the point $j$ (of body $i$) expressed in global coordinates, which is the same as $\mathbf{r}_j^i$ which explicitly indicates that point $j$ belongs to body $i$. The same notation holds for velocities and accelerations, for example, $\dot{\mathbf{r}}_j = \dot{\mathbf{r}}_j^i$ is the velocity of point $j$ of body $i$.

An optional upper bar indicates local coordinates in the reference frame of the body, thus point $\bar{\mathbf{r}}_j = \bar{\mathbf{r}}_j^i$ is the point $j$ (of body $i$) expressed in the local reference frame of the body. It is obvious that $\dot{\bar{\mathbf{r}}}_j = \dot{\bar{\mathbf{r}}}_j^i = \mathbf{0}$ because point $j$ belongs to body $i$.

For angular velocity vectors and matrices the notation is slightly different: a subscript indicates angular velocity of the body, for example, $\tilde{\boldsymbol{\omega}}_i$ and $\boldsymbol{\omega}_i$ indicate angular velocity of body $i$ in the global reference frame and upper bar indicates again local coordinates in the reference frame of the body. For example, $\bar{\tilde{\boldsymbol{\omega}}}_i$ and $\bar{\boldsymbol{\omega}}_i$ indicate angular velocities of body $i$ in the local reference frame of the body. Moreover, a superscript indicates relative angular velocities, for example, $\boldsymbol{\omega}_i^{i-1}$ and $\tilde{\boldsymbol{\omega}}_i^{i-1}$ indicate relative angular velocity of body $i$ with respect to body $i-1$ in the global reference frame while $\bar{\boldsymbol{\omega}}_i^{i-1}$ and $\bar{\tilde{\boldsymbol{\omega}}}_i^{i-1}$ angular velocity of body $i$ with respect to body $i-1$ in the local reference frame of body $i-1$.

Using the equations of the rigid body, the position of a point $j$ and a vector $j$, both belonging to body $i$, can be expressed in terms of their local coordinates, and the position of a reference point $i$ in the same body, as follows,

$$\mathbf{r}_j^i = \mathbf{r}_i^i + \mathbf{A}_i \left(\bar{\mathbf{r}}_j^i - \bar{\mathbf{r}}_i^i\right) \Rightarrow \mathbf{r}_j = \mathbf{r}_i + \mathbf{A}_i \left(\bar{\mathbf{r}}_j - \bar{\mathbf{r}}_i\right) \tag{2.1}$$

$$\mathbf{u}_j^i = \mathbf{A}_i \bar{\mathbf{u}}_j^i \Rightarrow \mathbf{u}_j = \mathbf{A}_i \bar{\mathbf{u}}_j \tag{2.2}$$

Observe that matrix $\mathbf{A}_i$ is the rotation matrix of body $i$ and it is, by definition, an orthogonal matrix, $\mathbf{A}_i^{\mathrm{T}}\mathbf{A}_i = \mathbf{A}_i\mathbf{A}_i^{\mathrm{T}} = \mathbf{I}_3$. Therefore, the following inverse relations hold,

$$\bar{\mathbf{r}}_j = \bar{\mathbf{r}}_i + \mathbf{A}_i^{\mathrm{T}} \left(\mathbf{r}_j - \mathbf{r}_i\right) \tag{2.3}$$

$$\bar{\mathbf{u}}_j = \mathbf{A}_i^{\mathrm{T}} \mathbf{u}_j \tag{2.4}$$

The velocities of the same entities can be expressed by,

$$\dot{\mathbf{r}}_j^i = \dot{\mathbf{r}}_i^i + \dot{\mathbf{A}}_i \left(\bar{\mathbf{r}}_j - \bar{\mathbf{r}}_i\right) + \mathbf{A}_i \left(\cancel{\dot{\bar{\mathbf{r}}}}_j - \cancel{\dot{\bar{\mathbf{r}}}}_i\right) = \dot{\mathbf{r}}_i^i + \dot{\mathbf{A}}_i \mathbf{A}_i^{\mathrm{T}} \left(\mathbf{r}_j - \mathbf{r}_i\right) \Rightarrow \dot{\mathbf{r}}_j = \dot{\mathbf{r}}_i + \tilde{\boldsymbol{\omega}}_i \left(\mathbf{r}_j - \mathbf{r}_i\right) \tag{2.5}$$

$$\dot{\mathbf{u}}_j = \dot{\mathbf{A}}_i \bar{\mathbf{u}}_j + \mathbf{A}_i \cancel{\dot{\bar{\mathbf{u}}}}_j = \dot{\mathbf{A}}_i \mathbf{A}_i^{\mathrm{T}} \mathbf{u}_j \Rightarrow \dot{\mathbf{u}}_j = \tilde{\boldsymbol{\omega}}_i \mathbf{u}_j \tag{2.6}$$

Taking into account the following definitions and properties,

$$\tilde{\boldsymbol{\omega}}_i = \dot{\mathbf{A}}_i \mathbf{A}_i^{\mathrm{T}} \tag{2.7}$$

$$\tilde{\boldsymbol{\omega}} \mathbf{u} = \boldsymbol{\omega} \wedge \mathbf{u} \tag{2.8}$$

$$\tilde{\boldsymbol{\omega}} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \tag{2.9}$$

$$\boldsymbol{\omega} = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^{\mathrm{T}} \tag{2.10}$$

where $\tilde{\boldsymbol{\omega}}$ is the skew-symmetric angular velocity matrix, $\boldsymbol{\omega}$ is the angular velocity vector and $\mathbf{u}$ any point or vector.

The velocities of a point $j$ and a vector $j$, both belonging to body $i$ can be expressed then,

$$\dot{\mathbf{r}}_j = \dot{\mathbf{r}}_i + \boldsymbol{\omega}_i \wedge \left(\mathbf{r}_j - \mathbf{r}_i\right) \tag{2.11}$$

$$\dot{\mathbf{u}}_j = \boldsymbol{\omega}_i \wedge \mathbf{u}_j \tag{2.12}$$

The accelerations of the same magnitudes can be obtained taking derivatives in the previous expressions,

$$\ddot{\mathbf{r}}_j = \ddot{\mathbf{r}}_i + \dot{\boldsymbol{\omega}}_i \wedge \left(\mathbf{r}_j - \mathbf{r}_i\right) + \boldsymbol{\omega}_i \wedge \left(\dot{\mathbf{r}}_j - \dot{\mathbf{r}}_i\right) \tag{2.13}$$

$$\ddot{\mathbf{u}}_j = \dot{\boldsymbol{\omega}}_i \wedge \mathbf{u}_j + \boldsymbol{\omega}_i \wedge \dot{\mathbf{u}}_j \tag{2.14}$$

and replacing the velocities calculated before, the final expressions become:

$$\ddot{\mathbf{r}}_j = \ddot{\mathbf{r}}_i + \dot{\boldsymbol{\omega}}_i \wedge \left(\mathbf{r}_j - \mathbf{r}_i\right) + \boldsymbol{\omega}_i \wedge \left(\boldsymbol{\omega}_i \wedge \left(\mathbf{r}_j - \mathbf{r}_i\right)\right) \tag{2.15}$$

$$\ddot{\mathbf{u}}_j = \dot{\boldsymbol{\omega}}_i \wedge \mathbf{u}_j + \boldsymbol{\omega}_i \wedge \left(\boldsymbol{\omega}_i \wedge \mathbf{u}_j\right) \tag{2.16}$$

Once given the previous definitions, the description of the set of joint types considered in the present work can be tackled.

### 2.1.1 Revolute joint kinematics

A revolute joint is generated when two bodies share one point and one vector, being the rotation around the shared vector the only motion allowed. The angle described by this rotation is measured by the relative coordinate $z_k$ that defines the joint, as it is shown in Figure 2.2.

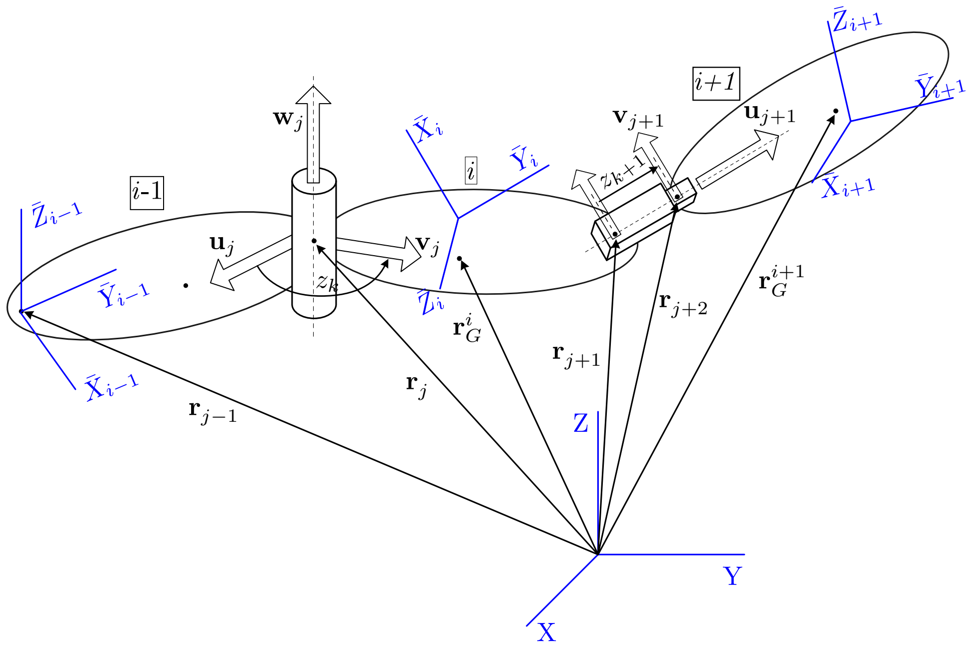


Figure 2.2: Revolute and prismatic joints in an open chain

Given the point $\mathbf{r}_j$ and unit vector $\mathbf{w}_j$ belonging to revolute joint $i$, shared by bodies $i-1$ and $i$, equations (2.1) and (2.2) hold,

$$\mathbf{r}_j = \mathbf{r}_{j-1} + \mathbf{A}_{i-1}\left(\bar{\mathbf{r}}_j^{i-1} - \bar{\mathbf{r}}_{j-1}^{i-1}\right) \tag{2.17}$$

$$\mathbf{w}_j = \mathbf{A}_{i-1}\bar{\mathbf{w}}_j^{i-1} \tag{2.18}$$

where $\mathbf{A}_{i-1}$ is the rotation matrix of body $i-1$.

The rotation matrix of body $i$ can be written in terms of the rotation matrix of the preceding body in the kinematic chain, and the relative rotation matrix,

$$\mathbf{A}_i = \mathbf{A}_{i-1}\mathbf{A}_i^{i-1} \tag{2.19}$$

Moreover, the relative rotation matrix $\mathbf{A}_i^{i-1}$ is composed of two rotations as well,

$$\mathbf{A}_i^{i-1} = \mathbf{A}^1\mathbf{A}^2, \tag{2.20}$$

the second one, $\mathbf{A}^2$ is a single rotation around the axis of the revolute joint, $\bar{\mathbf{w}}_j^i$ of magnitude $\phi = z_k$, while the first term, $\mathbf{A}^1$, is a constant rotation that makes vectors $\bar{\mathbf{w}}_j^{i-1}$ and $\bar{\mathbf{w}}_j^i$ coincident. There are infinite rotations that lead to concurrent vectors

$\bar{\mathbf{w}}_j^{i-1}$ and $\bar{\mathbf{w}}_j^i$ , being the one of interest that one which makes another pair of vectors perpendicular to the axis, $\bar{\mathbf{u}}_j^{i-1}$ and $\bar{\mathbf{v}}_j^i$, coincident too:

$$\mathbf{A}^1 = \left[\bar{\mathbf{w}}_j^{i-1} \quad \bar{\mathbf{u}}_j^{i-1} \quad \bar{\mathbf{w}}_j^{i-1} \wedge \bar{\mathbf{u}}_j^{i-1}\right] \left[\bar{\mathbf{w}}_j^i \quad \bar{\mathbf{v}}_j^i \quad \bar{\mathbf{w}}_j^i \wedge \bar{\mathbf{v}}_j^i\right]^{-1} = \bar{\mathbf{X}}^{i-1} \left(\bar{\mathbf{X}}^i\right)^{-1} \tag{2.21}$$

These newly added pair of vectors $\bar{\mathbf{u}}_j^{i-1}$ and $\bar{\mathbf{v}}_j^i$ belonging to bodies $i-1$ and $i$, respectively, also define the measurement of the joint angle $z_k$. Nevertheless, they do not need to exist themselves, they can be the projection on the normal joint plane of other vectors. Observe that all the magnitudes of the previous expression are local coordinates (constant in rigid bodies), thus $\mathbf{A}^1$ could be calculated only once when the model is defined.

Besides, $\mathbf{A}^2$ can be written in terms of a single rotation $z_k$ around the axis of the revolute joint $\bar{\mathbf{w}}_j^i$, according to [8],

$$\mathbf{A}^2 = \cos z_k \,\mathbf{I} + (1 - \cos z_k)\, \bar{\mathbf{w}}_j^i \left(\bar{\mathbf{w}}_j^i\right)^{\mathrm{T}} + \tilde{\bar{\mathbf{w}}}_j^i \sin z_k \tag{2.22}$$

The position of point $\mathbf{r}_{j+1}$, and the vector belonging to the prismatic pair in element $i$ in connection with the next body $i+1$ are given again by (2.1) and (2.11),

$$\mathbf{r}_{j+1} = \mathbf{r}_j + \mathbf{A}_i \left(\bar{\mathbf{r}}_{j+1}^i - \bar{\mathbf{r}}_j^i\right) \tag{2.23}$$

$$\mathbf{u}_{j+1} = \mathbf{A}_i \bar{\mathbf{u}}_{j+1}^i \tag{2.24}$$

where $\bar{\mathbf{r}}_{j+1}^i$, $\bar{\mathbf{r}}_j^i$ and $\bar{\mathbf{u}}_{j+1}^i$ are expressed in the local frame of body $i$, and the velocities of the mentioned entities, can be calculated as in (2.11) and (2.12)

$$\dot{\mathbf{r}}_{j+1} = \dot{\mathbf{r}}_j + \boldsymbol{\omega}_i \wedge \left(\mathbf{r}_{j+1} - \mathbf{r}_j\right) \tag{2.25}$$

$$\dot{\mathbf{u}}_{j+1} = \boldsymbol{\omega}_i \wedge \mathbf{u}_{j+1} \tag{2.26}$$

The angular velocity of body $i$ is:

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} + \dot{z}_k \mathbf{w}_j \tag{2.27}$$

Finally, the positions and velocities of additional points and vectors of body $i$ (maybe required for evaluation of forces or constraints) can be calculated by means of the rigid body equations for positions and velocities of points and vectors given by (2.1), (2.2), (2.11) and (2.12).

### 2.1.2 Prismatic joint kinematics

The prismatic joint, as the revolute joint, has one degree of freedom, but in this case it is associated to a translation instead of a rotation. In terms of natural or fully Cartesian coordinates, this type of joint is defined, in the simplest case, by two bodies sharing 2 vectors, and with a constraint of alignment of 2 points (one of each body) along one of these vectors. The relative motion of these two points is measured by a translation coordinate $z_{k+1}$.

Let us focus on Figure 2.2 to study the joint. The position of the point belonging to the prismatic joint in element $i+1$ is:

$$\mathbf{r}_{j+2} = \mathbf{r}_{j+1} + z_{k+1}\mathbf{u}_{j+1} \tag{2.28}$$

For any other point of body $i+1$, as the CoM, for instance, its position can be expressed using the equations of the rigid body introduced before,

$$\mathbf{r}_G^{i+1} = \mathbf{r}_{j+2} + \mathbf{A}_{i+1}\left(\bar{\mathbf{r}}_G^{i+1} - \bar{\mathbf{r}}_{j+2}^{i+1}\right) \tag{2.29}$$

and again the rotation matrix $\mathbf{A}_{i+1}$, of body $i+1$ can be expressed in terms of the rotation matrix of $i$,

$$\mathbf{A}_{i+1} = \mathbf{A}_i\mathbf{A}_{i+1}^i \tag{2.30}$$

For a prismatic joint connecting bodies $i$ and $i+1$, the relative rotation matrix $\mathbf{A}_{i+1}^i$ is constant and therefore it can be calculated when the joint is defined and used in simulation time. In order to completely define the joint, at least two vectors in each body are needed: the vector of the axis in each body $\bar{\mathbf{u}}_{j+1}^i$, $\bar{\mathbf{u}}_{j+1}^{i+1}$ and one extra coincident vector $\bar{\mathbf{v}}_{j+1}^i$, $\bar{\mathbf{v}}_{j+1}^{i+1}$.

$$\mathbf{A}_{i+1}^i = \begin{bmatrix}\bar{\mathbf{u}}_{j+1}^i & \bar{\mathbf{v}}_{j+1}^i & \bar{\mathbf{u}}_{j+1}^i \wedge \bar{\mathbf{v}}_{j+1}^i\end{bmatrix}\begin{bmatrix}\bar{\mathbf{u}}_{j+1}^{i+1} & \bar{\mathbf{v}}_{j+1}^{i+1} & \bar{\mathbf{u}}_{j+1}^{i+1} \wedge \bar{\mathbf{v}}_{j+1}^{i+1}\end{bmatrix}^{-1} = \bar{\mathbf{X}}^i\left(\bar{\mathbf{X}}^{i+1}\right)^{-1} \tag{2.31}$$

Due to the constant relative orientation, the angular velocities of bodies $i-1$ and $i$ coincide:

$$\boldsymbol{\omega}_{i+1} = \boldsymbol{\omega}_i \tag{2.32}$$

In order to calculate velocities of points of body $i+1$, the velocity of $\dot{\mathbf{r}}_{j+2}$ should be calculated as reference point for the distribution of velocities,

$$\dot{\mathbf{r}}_{j+2} = \dot{\mathbf{r}}_{j+1} + \dot{z}_{k+1}\mathbf{u}_{j+1} + z_{k+1}\dot{\mathbf{u}}_{j+1} = \dot{\mathbf{r}}_{j+1} + \dot{z}_{k+1}\mathbf{u}_{j+1} + z_{k+1}\boldsymbol{\omega}_{i+1} \wedge \mathbf{u}_{j+1} \tag{2.33}$$

where equation (2.12) has been used.

Finally, the positions and velocities of additional points and vectors belonging to body $i+1$ can be calculated by means of the rigid body equations for positions and velocities of points and vectors given by (2.1), (2.2), (2.11) and (2.12).

### 2.1.3 Cardan joint kinematics

Observing Figure 2.3, it can be deduced that the kinematics of the Cardan joint is equivalent to the kinematics of two consecutive revolute joints. The first revolute joint is defined by the point $\mathbf{r}_j$ shared by bodies $i-1$ and $i$ and the vector $\mathbf{w}_j = \mathbf{w}_j^{i-1}$; the rotation $z_{k1}$ around the vector of the revolute joint is measured by the angle between vectors $z_{k1} = \angle\left\{\mathbf{u}_j, \mathbf{v}_j\right\}$ in the direction of $\mathbf{w}_j$. The second revolute joint is defined by the same point $\mathbf{r}_j$ and the vector $\mathbf{w}_{j+1} = \mathbf{w}_{j+1}^i$; the rotation $z_{k2}$ around the vector of the revolute joint is measured by the angle between vectors $z_{k2} = \angle\{\mathbf{u}_{j+1}, \mathbf{v}_{j+1}\}$ in the direction of $\mathbf{w}_{j+1}$.

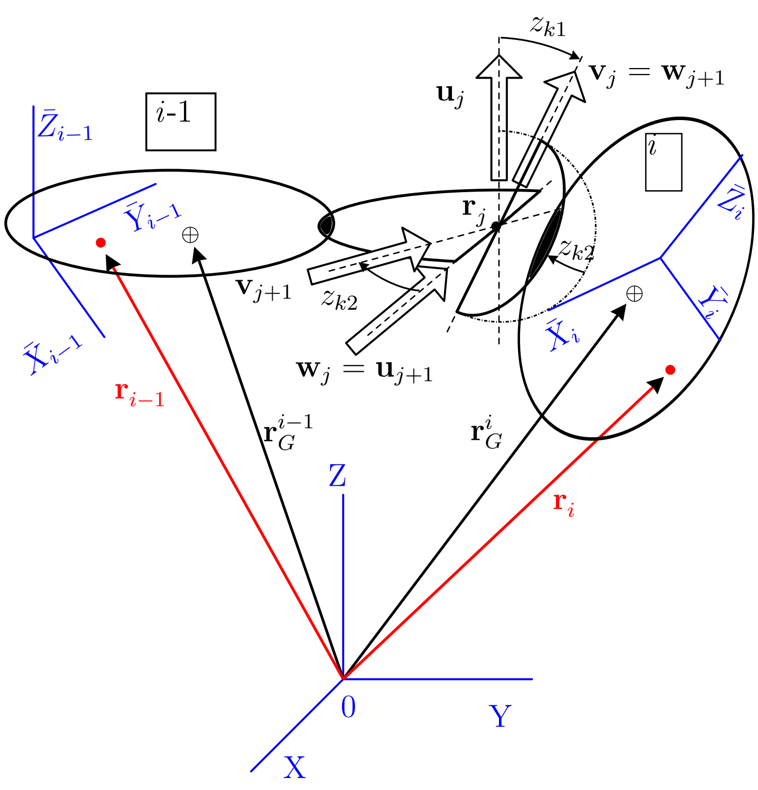


Figure 2.3: Cardan joint

Since $\mathbf{r}_j$ belongs to body $i$, the position of any point of body $i$, and in particular the reference point $\mathbf{r}_i$, can be expressed analogously to the case of the revolute joint,

$$\mathbf{r}_i = \mathbf{r}_j + \mathbf{A}_i \left( \bar{\mathbf{r}}_i^i - \bar{\mathbf{r}}_j^i \right) \tag{2.34}$$

Global positions of Cardan vectors can be obtained in accordance with the body in which they are defined:

$$\mathbf{w}_j = \mathbf{A}_{i-1} \bar{\mathbf{w}}_j^{i-1} \tag{2.35}$$
$$\mathbf{u}_j = \mathbf{A}_{i-1} \bar{\mathbf{u}}_j^{i-1} \tag{2.36}$$
$$\mathbf{w}_{j+1} = \mathbf{A}_i \bar{\mathbf{w}}_{j+1}^i \tag{2.37}$$
$$\mathbf{v}_{j+1} = \mathbf{A}_i \bar{\mathbf{v}}_{j+1}^i \tag{2.38}$$

The rotation matrix of body $i$ shares also the same expression (2.19),

$$\mathbf{A}_i = \mathbf{A}_{i-1} \mathbf{A}_i^{i-1} \tag{2.39}$$

The relative rotation matrix $\mathbf{A}_i^{i-1}$ is composed of other three rotations matrices: one related to each revolute joint plus one constant rotation matrix enforcing the local axes of the bodies to be coincident.

$$\mathbf{A}_i^{i-1} = \mathbf{A}_{k1}^1 \mathbf{A}^2 \mathbf{A}_{k2}^3 \tag{2.40}$$

Rotation matrices $\mathbf{A}^1_{k1}$ and $\mathbf{A}^3_{k2}$ constitute single rotations around the respective axes of the revolute joints, $\bar{\mathbf{w}}^{i-1}_j$ of magnitude $z_{k1}$, and $\bar{\mathbf{w}}^{i}_{j+1}$ of magnitude $z_{k2}$:

$$\mathbf{A}^1_{k1} = \cos z_{k1}\ \mathbf{I} + (1-\cos z_{k1})\, \bar{\mathbf{w}}^{i-1}_j \left(\bar{\mathbf{w}}^{i-1}_j\right)^{\mathrm{T}} + \tilde{\bar{\mathbf{w}}}^{i-1}_j \sin z_{k1} \tag{2.41}$$

$$\mathbf{A}^3_{k2} = \cos z_{k2}\ \mathbf{I} + (1-\cos z_{k2})\, \bar{\mathbf{w}}^{i}_{j+1} \left(\bar{\mathbf{w}}^{i}_{j+1}\right)^{\mathrm{T}} + \tilde{\bar{\mathbf{w}}}^{i}_{j+1} \sin z_{k2} \tag{2.42}$$

while the second term, $\mathbf{A}^2$, is a constant rotation matrix which matches $\bar{\mathbf{w}}^{i}_{j+1}$ with $\bar{\mathbf{u}}^{i-1}_j$, and vector $\bar{\mathbf{v}}^{i}_{j+1}$ with $\bar{\mathbf{w}}^{i-1}_j$, respectively:

$$\mathbf{A}^2 = \begin{bmatrix}\bar{\mathbf{u}}^{i-1}_j & \bar{\mathbf{w}}^{i-1}_j & \bar{\mathbf{u}}^{i-1}_j \wedge \bar{\mathbf{w}}^{i-1}_j\end{bmatrix} \begin{bmatrix}\bar{\mathbf{w}}^{i}_{j+1} & \bar{\mathbf{v}}^{i}_{j+1} & \bar{\mathbf{w}}^{i}_{j+1} \wedge \bar{\mathbf{v}}^{i}_{j+1}\end{bmatrix}^{-1} = \bar{\mathbf{X}}^{i-1}\left(\bar{\mathbf{X}}^{i}\right)^{-1} \tag{2.43}$$

Observe that all the magnitudes of the previous expression are local coordinates, therefore $\mathbf{A}^2$ has to be calculated only once when the model is defined.

Velocity relations for Cardan joints hold:

$$\dot{\mathbf{r}}_i = \dot{\mathbf{r}}_j + \boldsymbol{\omega}_i \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) \tag{2.44}$$

$$\dot{\mathbf{w}}_j = \boldsymbol{\omega}_{i-1} \wedge \mathbf{w}_j \tag{2.45}$$

$$\dot{\mathbf{u}}_j = \boldsymbol{\omega}_{i-1} \wedge \mathbf{u}_j \tag{2.46}$$

$$\dot{\mathbf{w}}_{j+1} = \boldsymbol{\omega}_{i} \wedge \mathbf{w}_{j+1} \tag{2.47}$$

$$\dot{\mathbf{v}}_{j+1} = \boldsymbol{\omega}_{i} \wedge \mathbf{v}_{j+1} \tag{2.48}$$

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} + \dot{z}_{k1}\mathbf{w}_j + \dot{z}_{k2}\mathbf{w}_{j+1} \tag{2.49}$$

### 2.1.4 Cylindrical joint kinematics

The cylindrical joint is a kinematic joint with two degrees of freedom which allows a relative motion around and along one vector. In fact, its kinematics is equivalent to the kinematics of one prismatic joint plus a revolute joint. This type of joint is described by 2 relative coordinates (one angular and other translational), and it is equivalent to two bodies sharing one vector with a constraint of alignment of two points (one of each body) along it.

In accordance with the cylindrical joint displayed in Figure 2.4, points $\mathbf{r}_j$ and $\mathbf{r}_{j+1}$ of body $i$ can be determined as:

$$\mathbf{r}_j = \mathbf{r}_{j-1} + z_{k1}\mathbf{w}_j \tag{2.50}$$

$$\mathbf{r}_{j+1} = \mathbf{r}_j + \mathbf{A}_i\left(\bar{\mathbf{r}}^i_{j+1} - \bar{\mathbf{r}}^i_j\right) \tag{2.51}$$

where $\mathbf{A}_i$ is given by equation (2.19). Therefore,

$$\mathbf{r}_{j+1} = \mathbf{r}_{j-1} + z_{k1}\mathbf{w}_j + \mathbf{A}_i\left(\bar{\mathbf{r}}^i_{j+1} - \bar{\mathbf{r}}^i_j\right) \tag{2.52}$$

The velocities of points $j$ and $j+1$,

$$\dot{\mathbf{r}}_j = \dot{\mathbf{r}}_{j-1} + \dot{z}_{k1}\mathbf{w}_j + \boldsymbol{\omega}_{i-1} \wedge z_{k1}\mathbf{w}_j \tag{2.53}$$

$$\dot{\mathbf{r}}_{j+1} = \dot{\mathbf{r}}_j + \boldsymbol{\omega}_i \wedge \left(\mathbf{r}_{j+1} - \mathbf{r}_j\right) \tag{2.54}$$

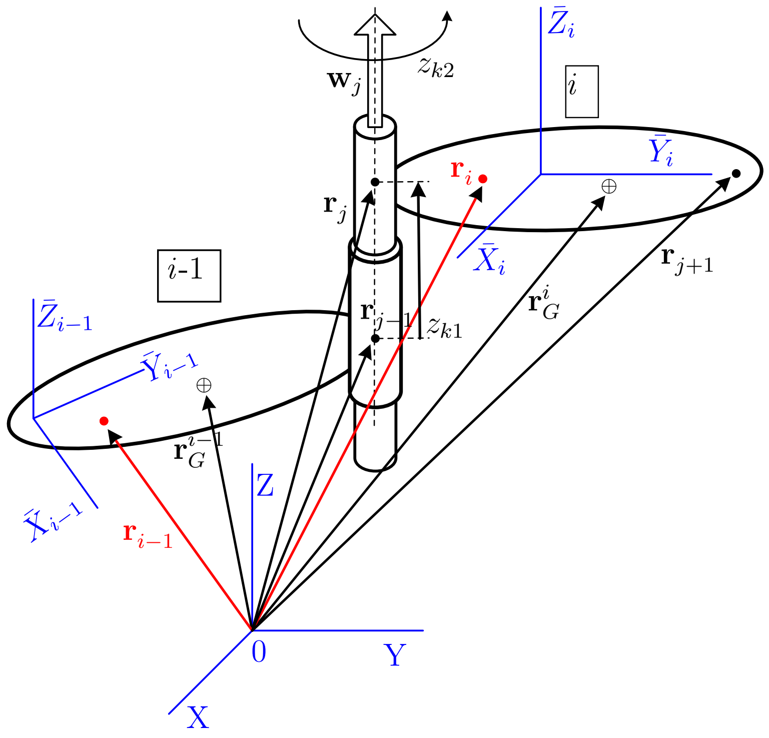


Figure 2.4: Cylindrical joint

Since the relative change in orientation is due to the rotation coordinate, the angular velocity holds an identical equation to the revolute joint, (2.27):

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} + \dot{z}_{k2}\mathbf{w}_j \tag{2.55}$$

Removing the velocity of point $j$ in (2.53) and (2.54), the expression of point $j+1$ in terms of $j-1$ can be obtained,

$$\dot{\mathbf{r}}_{j+1} = \dot{\mathbf{r}}_{j-1} + \dot{z}_{k1}\mathbf{w}_j + \boldsymbol{\omega}_{i-1} \wedge z_{k1}\mathbf{w}_j + \boldsymbol{\omega}_i \wedge \left(\mathbf{r}_{j+1} - \mathbf{r}_j\right) \tag{2.56}$$

However, equation (2.56) is not strictly necessary for the kinematics, because once calculated $\mathbf{r}_j$, $\dot{\mathbf{r}}_j$, $\mathbf{A}_i$ and $\boldsymbol{\omega}_i$, the general expressions for velocities of points and vectors hold: (2.1), (2.2), (2.11) (or (2.54)) and (2.12).

### 2.1.5 Spherical joint kinematics

The spherical joint is analog to two bodies sharing one point, and therefore three relative rotations are allowed while relative translation is prevented. In Figure 2.5, a spherical joint is displayed relating the motion of bodies $i-1$ and $i$.

If the kinematics of body $i-1$ are known, and regarding that the point identified as $j$ in Figure 2.5 is shared between $i-1$ and $i$, the position and velocity of any point rigidly attached to body $i$ can be determined by:

$$\mathbf{r}_{j+1} = \mathbf{r}_j + \mathbf{A}_i\left(\bar{\mathbf{r}}_{j+1}^i - \bar{\mathbf{r}}_j^i\right) \tag{2.57}$$

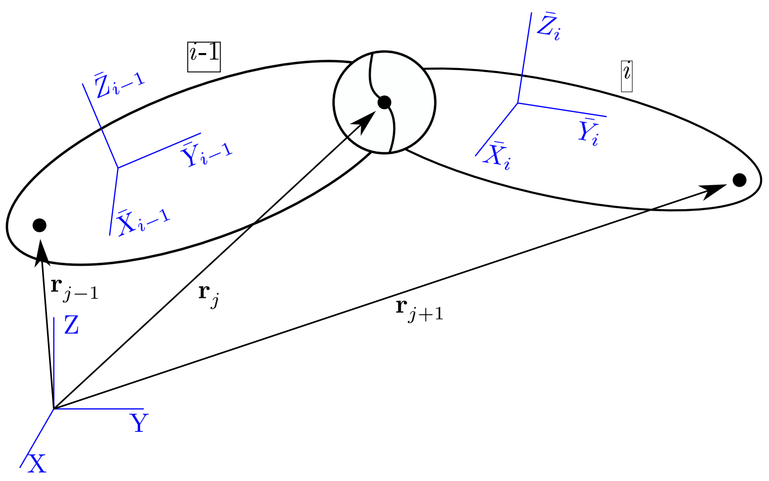


Figure 2.5: Spherical joint

$$\dot{\mathbf{r}}_{j+1} = \dot{\mathbf{r}}_j + \tilde{\boldsymbol{\omega}}_i \left(\mathbf{r}_{j+1} - \mathbf{r}_j\right) \tag{2.58}$$

where the rotation matrix can be obtained as the product of the rotation matrix of the previous body by a relative rotation matrix referred to the joint, as it was established for the revolute joint in (2.19):

$$\mathbf{A}_i = \mathbf{A}_{i-1}\mathbf{A}_i^{i-1} \tag{2.59}$$

Analogously, the relations for angular velocities can be expressed as follows:

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} + \boldsymbol{\omega}_i^{i-1} \tag{2.60}$$

wherein $\boldsymbol{\omega}_i^{i-1}$ is the relative angular velocity of body $i$ with respect to body $i-1$ expressed in the global reference frame.

The three relative rotations can be parameterized by means of Euler angles or roll-pitch-yaw angles, for instance, but any parameterization with three parameters presents singular configurations. In order to avoid any risk of singular configurations, Euler parameters can be chosen to model the spherical joint [114]. The relative rotation matrix between two bodies can be obtained from a set of Euler parameters by means of:

$$\mathbf{A}_i^{i-1} = \left(2e_0^2 - 1\right)\mathbf{I} + 2\left(\bar{\mathbf{e}}\bar{\mathbf{e}}^{\mathrm{T}} + e_0\tilde{\bar{\mathbf{e}}}\right) \tag{2.61a}$$

$$e_0 = \cos\frac{\phi}{2} \tag{2.61b}$$

$$\bar{\mathbf{e}} = \begin{bmatrix} \bar{e}_1 \\ \bar{e}_2 \\ \bar{e}_3 \end{bmatrix} = \bar{\mathbf{u}}^{i-1}\sin\frac{\phi}{2} \tag{2.61c}$$

where $\phi$ is the angle of rotation around the axis $\bar{\mathbf{u}}^{i-1}$.

Only three out of the four Euler parameters $\bar{\mathbf{p}} = \begin{bmatrix} e_0 & \bar{\mathbf{e}}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$ are independent, since they are subjected to the following normalization constraint equation:

$$\bar{\mathbf{p}}^{\mathrm{T}}\bar{\mathbf{p}} - 1 = {e_0}^2 + \bar{\mathbf{e}}^{\mathrm{T}}\bar{\mathbf{e}} - 1 = 0 \tag{2.62}$$

Defining $\bar{\mathbf{E}} = \begin{bmatrix} -\bar{\mathbf{e}} & \tilde{\bar{\mathbf{e}}} + e_0\mathbf{I} \end{bmatrix}$ and $\bar{\mathbf{G}} = \begin{bmatrix} -\mathbf{e} & -\tilde{\bar{\mathbf{e}}} + e_0\mathbf{I} \end{bmatrix}$ the expression of the relative rotation matrix can be simplified,

$$\begin{gathered}\mathbf{A}_i^{i-1} = \bar{\mathbf{E}}\bar{\mathbf{G}}^{\mathrm{T}} = \bar{\mathbf{e}}\bar{\mathbf{e}}^{\mathrm{T}} + \tilde{\bar{\mathbf{e}}}\tilde{\bar{\mathbf{e}}} + 2e_0\tilde{\bar{\mathbf{e}}} + {e_0}^2\mathbf{I} = 2\bar{\mathbf{e}}\bar{\mathbf{e}}^{\mathrm{T}} - \left(\bar{\mathbf{e}}^{\mathrm{T}}\bar{\mathbf{e}}\right)\mathbf{I} + 2e_0\tilde{\bar{\mathbf{e}}} + {e_0}^2\mathbf{I} = \\ \left({e_0}^2 - \bar{\mathbf{e}}^{\mathrm{T}}\bar{\mathbf{e}}\right)\mathbf{I} + 2\left(\bar{\mathbf{e}}\bar{\mathbf{e}}^{\mathrm{T}} + e_0\tilde{\bar{\mathbf{e}}}\right) = \left(2{e_0}^2 - 1\right)\mathbf{I} + 2\left(\bar{\mathbf{e}}\bar{\mathbf{e}}^{\mathrm{T}} + e_0\tilde{\bar{\mathbf{e}}}\right)\end{gathered} \tag{2.63}$$

where the identity $\tilde{\mathbf{a}}\tilde{\mathbf{b}} = \mathbf{b}\mathbf{a}^{\mathrm{T}} - \left(\mathbf{a}^{\mathrm{T}}\mathbf{b}\right)\mathbf{I}$ was used. Moreover, it can be easily proved that matrices $\bar{\mathbf{E}}$ and $\bar{\mathbf{G}}$ satisfy the following relations,

$$\bar{\mathbf{E}}\bar{\mathbf{p}} = \bar{\mathbf{G}}\bar{\mathbf{p}} = \mathbf{0} \tag{2.64}$$

$$\bar{\mathbf{E}}\bar{\mathbf{E}}^{\mathrm{T}} = \bar{\mathbf{G}}\bar{\mathbf{G}}^{\mathrm{T}} = \mathbf{I}_3 \tag{2.65}$$

$$\bar{\mathbf{E}}^{\mathrm{T}}\bar{\mathbf{E}} = \bar{\mathbf{G}}^{\mathrm{T}}\bar{\mathbf{G}} = \mathbf{I}_4 - \bar{\mathbf{p}}\bar{\mathbf{p}}^{\mathrm{T}} \tag{2.66}$$

Some relations of the time derivatives of Euler parameters bring,

$$\dot{\bar{\mathbf{E}}}\bar{\mathbf{E}}^{\mathrm{T}} = -\bar{\mathbf{E}}\dot{\bar{\mathbf{E}}}^{\mathrm{T}} \tag{2.67}$$

$$\dot{\bar{\mathbf{G}}}^{\mathrm{T}}\bar{\mathbf{G}} = -\bar{\mathbf{G}}^{\mathrm{T}}\dot{\bar{\mathbf{G}}} \tag{2.68}$$

$$\bar{\mathbf{p}}^{\mathrm{T}}\dot{\bar{\mathbf{p}}} = \dot{\bar{\mathbf{p}}}^{\mathrm{T}}\bar{\mathbf{p}} = e_0\dot{e}_0 + \dot{\bar{\mathbf{e}}}^{\mathrm{T}}\bar{\mathbf{e}} = 0 \tag{2.69}$$

$$\bar{\mathbf{E}}\dot{\bar{\mathbf{p}}} = -\dot{\bar{\mathbf{E}}}\bar{\mathbf{p}} = \bar{\mathbf{G}}\dot{\bar{\mathbf{p}}} = -\dot{\bar{\mathbf{G}}}\bar{\mathbf{p}} \tag{2.70}$$

$$\dot{\bar{\mathbf{E}}}\bar{\mathbf{G}}^{\mathrm{T}} = \bar{\mathbf{E}}\dot{\bar{\mathbf{G}}}^{\mathrm{T}} \tag{2.71}$$

Then the time derivative of the relative rotation matrix,

$$\dot{\mathbf{A}}_i^{i-1} = \dot{\bar{\mathbf{E}}}\bar{\mathbf{G}}^{\mathrm{T}} + \bar{\mathbf{E}}\dot{\bar{\mathbf{G}}}^{\mathrm{T}} = 2\bar{\mathbf{E}}\dot{\bar{\mathbf{G}}}^{\mathrm{T}} = 2\dot{\bar{\mathbf{E}}}\bar{\mathbf{G}}^{\mathrm{T}} \tag{2.72}$$

and the relative angular velocity,

$$\tilde{\bar{\boldsymbol{\omega}}}_i^{i-1} = \dot{\mathbf{A}}_i^{i-1}\left(\mathbf{A}_i^{i-1}\right)^{\mathrm{T}} = 2\dot{\bar{\mathbf{E}}}\bar{\mathbf{G}}^{\mathrm{T}}\bar{\mathbf{G}}\bar{\mathbf{E}}^{\mathrm{T}} = 2\dot{\bar{\mathbf{E}}}\left(\mathbf{I}_4 - \bar{\mathbf{p}}\bar{\mathbf{p}}^{\mathrm{T}}\right)\bar{\mathbf{E}}^{\mathrm{T}} = 2\dot{\bar{\mathbf{E}}}\bar{\mathbf{E}}^{\mathrm{T}} = -2\bar{\mathbf{E}}\dot{\bar{\mathbf{E}}}^{\mathrm{T}} \tag{2.73}$$

where identities (2.66), (2.64) and (2.67) have been used.

Direct calculation of $\widetilde{\bar{\mathbf{E}}\dot{\bar{\mathbf{p}}}}$ and $\bar{\mathbf{E}}\dot{\bar{\mathbf{E}}}^{\mathrm{T}}$ leads to the following result,

$$\begin{aligned}\widetilde{\bar{\mathbf{E}}\dot{\bar{\mathbf{p}}}} = & \left[-\bar{\mathbf{e}} \quad \tilde{\bar{\mathbf{e}}}\,\widetilde{+ e_0}\mathbf{I}\right]\begin{bmatrix}\dot{e}_0 \\ \dot{\bar{\mathbf{e}}}\end{bmatrix} = -\dot{e}_0\tilde{\bar{\mathbf{e}}} + \widetilde{\tilde{\bar{\mathbf{e}}}\dot{\bar{\mathbf{e}}}} + e_0\dot{\tilde{\bar{\mathbf{e}}}} = -\dot{e}_0\tilde{\bar{\mathbf{e}}} + \tilde{\bar{\mathbf{e}}}\dot{\tilde{\bar{\mathbf{e}}}} - \dot{\tilde{\bar{\mathbf{e}}}}\tilde{\bar{\mathbf{e}}} + e_0\dot{\tilde{\bar{\mathbf{e}}}} = \\ & - \dot{e}_0\tilde{\bar{\mathbf{e}}} + \tilde{\bar{\mathbf{e}}}\dot{\tilde{\bar{\mathbf{e}}}} - \bar{\mathbf{e}}\dot{\bar{\mathbf{e}}}^{\mathrm{T}} + \dot{\bar{\mathbf{e}}}^{\mathrm{T}}\bar{\mathbf{e}}\mathbf{I} + e_0\dot{\tilde{\bar{\mathbf{e}}}} = -\dot{e}_0\tilde{\bar{\mathbf{e}}} + \tilde{\bar{\mathbf{e}}}\dot{\tilde{\bar{\mathbf{e}}}} - \bar{\mathbf{e}}\dot{\bar{\mathbf{e}}}^{\mathrm{T}} - e_0\dot{e}_0\mathbf{I} + e_0\dot{\tilde{\bar{\mathbf{e}}}} = \\ & - \left(\bar{\mathbf{e}}\dot{\bar{\mathbf{e}}}^{\mathrm{T}} - \tilde{\bar{\mathbf{e}}}\dot{\tilde{\bar{\mathbf{e}}}} + \dot{e}_0\tilde{\bar{\mathbf{e}}} - e_0\dot{\tilde{\bar{\mathbf{e}}}} + e_0\dot{e}_0\mathbf{I}\right) = -\left[-\bar{\mathbf{e}} \quad \tilde{\bar{\mathbf{e}}} + e_0\mathbf{I}\right]\begin{bmatrix}-\dot{\bar{\mathbf{e}}}^{\mathrm{T}} \\ -\dot{\tilde{\bar{\mathbf{e}}}} + \dot{e}_0\mathbf{I}\end{bmatrix} = -\bar{\mathbf{E}}\dot{\bar{\mathbf{E}}}^{\mathrm{T}}\end{aligned} \tag{2.74}$$

where the following properties of skew-symmetric matrices were observed, $\widetilde{\tilde{\mathbf{a}}\mathbf{b}} = \tilde{\mathbf{a}}\tilde{\mathbf{b}} - \tilde{\mathbf{b}}\tilde{\mathbf{a}}$ and $\tilde{\mathbf{b}}\tilde{\mathbf{a}} = \mathbf{a}\mathbf{b}^{\mathrm{T}} + \left(\mathbf{b}^{\mathrm{T}}\mathbf{a}\right)\mathbf{I}$ (see appendix B) and $\dot{\bar{\mathbf{e}}}^{\mathrm{T}}\bar{\mathbf{e}} = -e_0\dot{e}_0$.

Therefore, the final expression for the relative angular velocity matrices and their relative angular velocity dual vectors, in the local frame of body $i-1$ and in the global frame, respectively, are,

$$\tilde{\bar{\boldsymbol{\omega}}}_i^{i-1} = -2\bar{\mathbf{E}}\dot{\bar{\mathbf{E}}}^{\mathrm{T}} = 2\widetilde{\bar{\mathbf{E}}\dot{\mathbf{p}}} \tag{2.75}$$

$$\bar{\boldsymbol{\omega}}_i^{i-1} = 2\bar{\mathbf{E}}\dot{\mathbf{p}} \tag{2.76}$$

$$\boldsymbol{\omega}_i^{i-1} = \mathbf{A}_{i-1}\bar{\boldsymbol{\omega}}_i^{i-1} = 2\mathbf{A}_{i-1}\bar{\mathbf{E}}\dot{\mathbf{p}} = 2\mathbf{E}\dot{\mathbf{p}} \tag{2.77}$$

$$\mathbf{E} = \mathbf{A}_{i-1}\bar{\mathbf{E}} \tag{2.78}$$

The inverse relation for $\dot{\mathbf{p}}$ can be obtained multiplying by $\bar{\mathbf{E}}$,

$$\dot{\mathbf{p}} = \frac{1}{2}\bar{\mathbf{E}}^{\mathrm{T}}\bar{\boldsymbol{\omega}}_i^{i-1} = \frac{1}{2}\bar{\mathbf{E}}^{\mathrm{T}}\mathbf{A}_{i-1}^{\mathrm{T}}\boldsymbol{\omega}_i^{i-1} = \frac{1}{2}\mathbf{E}^{\mathrm{T}}\boldsymbol{\omega}_i^{i-1} \tag{2.79}$$

Finally, with the position (and velocity) of the reference point $j$ and the rotation matrix (and angular velocity) of body $i$, the positions (or velocities) of any point and vector of body $i$ can be calculated by (2.1) and (2.2) (or by (2.11) and (2.12)).

### 2.1.6 Floating joint kinematics

There are different forms to manage relative coordinate models unconnected to the ground. One possibility is to reformulate the set of expressions for this particular case, considering the reference point coordinates of one of the bodies of the model. The approach addressed here consists in considering a "fictional" joint which is equivalent to the lack of joint, but which allows to extend the joint modeling formalism to the case of floating multibody models.

Due to the unconstrained motion between a body $i$ and the ground, the floating joint is defined by 3 translations and 3 rotations (DoF of the rigid body). In order to keep consistency, the three translations are modeled by three elemental prismatic joints aligned with the global axes of coordinates while the three rotations are described by one elemental spherical joint in terms of Euler parameters, as it can be seen in Figure 2.6:

- The first prismatic joint is aligned with the global $X$ axis, with $\mathbf{r}_j = \mathbf{0}$, $\mathbf{u}_j = \bar{\mathbf{u}}_j^0 = \begin{bmatrix}1 & 0 & 0\end{bmatrix}^{\mathrm{T}}$ and $\mathbf{r}_{j+1} = \mathbf{r} = \mathbf{0} + z_{k1}\mathbf{u}_j$.
- The second prismatic joint is aligned with the global $Y$ axis, with $\mathbf{r}_j = \mathbf{r}$, $\mathbf{v}_j = \bar{\mathbf{v}}_j^0 = \begin{bmatrix}0 & 1 & 0\end{bmatrix}^{\mathrm{T}}$ and $\mathbf{r}_{j+1} = \mathbf{s} = \mathbf{r} + z_{k2}\mathbf{v}_j$.
- The third prismatic joint is aligned with the global $Z$ axis, with $\mathbf{r}_j = \mathbf{s}$, $\mathbf{w}_j = \bar{\mathbf{w}}_j^0 = \begin{bmatrix}0 & 0 & 1\end{bmatrix}^{\mathrm{T}}$ and $\mathbf{r}_{j+1} = \mathbf{t} = \mathbf{s} + z_{k3}\mathbf{w}_j$.
- The spherical joint is attached to the end point of the last prismatic joint and to the center of gravity of the first body in the chain $\mathbf{r}_j = \mathbf{t} = \mathbf{r}_G^i$.

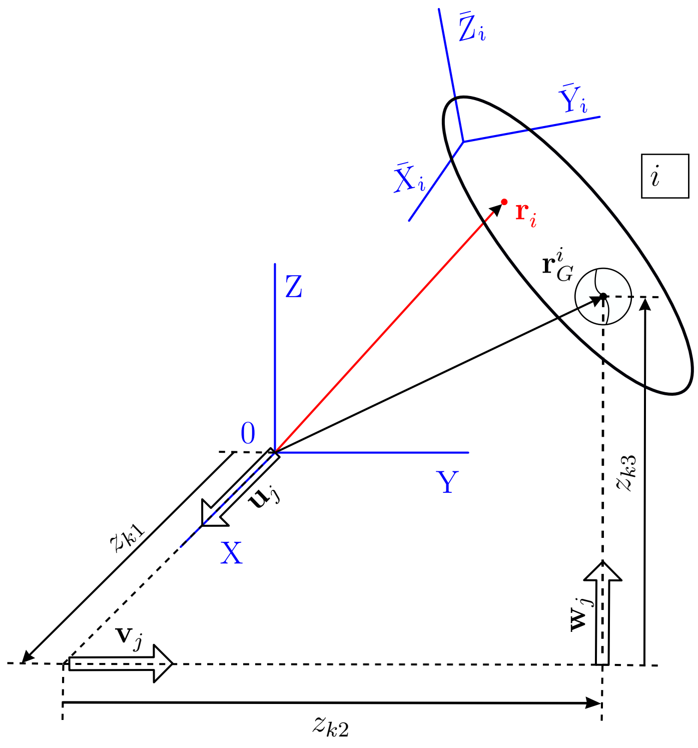


Figure 2.6: Floating joint

The rotation matrices satisfy the following equation,

$$\mathbf{A}_i = \mathbf{A}_0\mathbf{A}_i^0 = \mathbf{A}_i^0 \tag{2.80}$$

regarding that the rotation matrix of the ground (fixed body), $\mathbf{A}_0 = \mathbf{I}_3$, will be always considered the identity.

Similarly, relations for angular velocities can be expressed as follows,

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_0 + \boldsymbol{\omega}_i^0 = \boldsymbol{\omega}_i^0 \tag{2.81}$$

because the angular velocity of the ground is always null, $\boldsymbol{\omega}_0 = \mathbf{0}$.

Again, Euler parameters can be chosen to model the elemental spherical joint:

$$\mathbf{A}_i = \mathbf{A}_i^0 = \left(2e_0^2 - 1\right)\mathbf{I} + 2\left(\bar{\mathbf{e}}\bar{\mathbf{e}}^{\mathrm{T}} + e_0\tilde{\bar{\mathbf{e}}}\right) \tag{2.82a}$$

$$e_0 = \cos\frac{\phi}{2} \tag{2.82b}$$

$$\bar{\mathbf{e}} = \begin{bmatrix}\bar{e}_1\\ \bar{e}_2\\ \bar{e}_3\end{bmatrix} = \bar{\mathbf{u}}^0\sin\frac{\phi}{2} = \mathbf{u}\sin\frac{\phi}{2} = \mathbf{e} \tag{2.82c}$$

where $\phi$ is the angle of rotation around the axis $\bar{\mathbf{u}}^0$, which matches the global $\mathbf{u}$ in this case. Observe also that $\bar{\mathbf{p}} = \begin{bmatrix}e_0 & \bar{\mathbf{e}}^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}} = \begin{bmatrix}e_0 & \mathbf{e}^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}} = \mathbf{p}$, $\bar{\mathbf{E}} = \mathbf{E}$ and $\bar{\mathbf{G}} = \mathbf{G}$ in this case.

The time derivative of the relative rotation matrix holds:

$$\dot{\mathbf{A}}_i = 2\dot{\bar{\mathbf{E}}}\bar{\mathbf{G}}^{\mathrm{T}} = 2\dot{\mathbf{E}}\mathbf{G}^{\mathrm{T}} \tag{2.83}$$

and the angular velocity:

$$\boldsymbol{\omega}_i = 2\mathbf{E}\dot{\mathbf{p}} \tag{2.84}$$

The inverse relation for $\dot{\mathbf{p}}$ can be obtained multiplying by $\mathbf{E}$,

$$\dot{\mathbf{p}} = \frac{1}{2}\mathbf{E}^{\mathrm{T}}\boldsymbol{\omega}_i \tag{2.85}$$

### 2.1.7 Planar joint kinematics

The planar joint displayed in Figure 2.7 is equivalent to the relative motion of two bodies sharing one vector and with the translation prevented along the shared vector direction. The resulting relative motion is equivalent to a free 2D motion in the normal plane to the shared vector, with 3 degrees of freedom. This type of joint is very advantageous for the solution of planar mechanisms, instead of using a floating joint with 3 constraint equations.

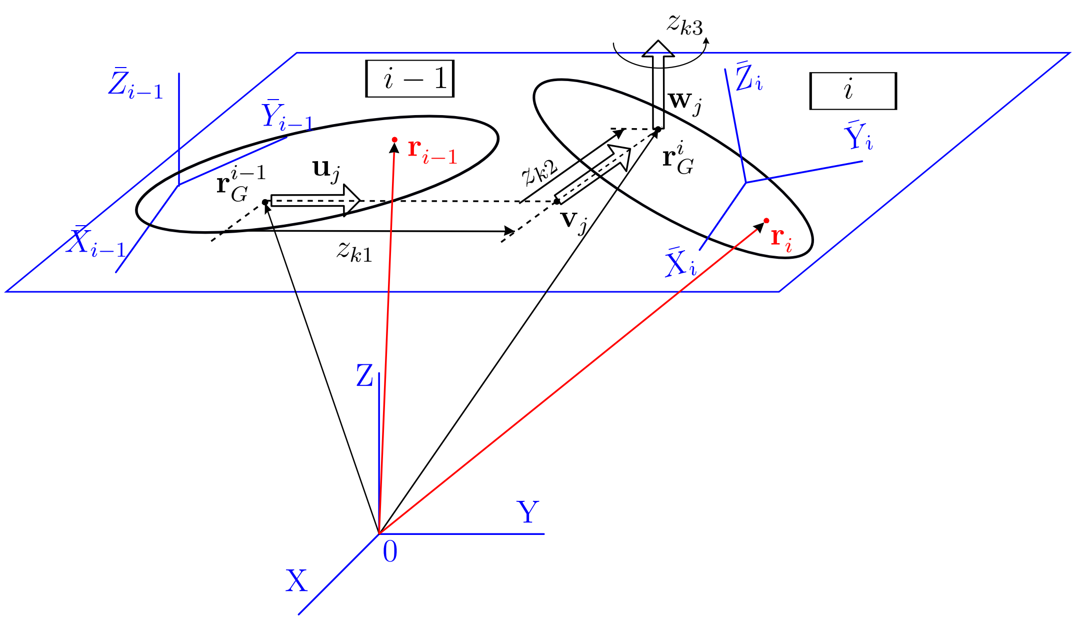


Figure 2.7: Planar joint

The behavior of the planar joint is analogous to the concatenation of two prismatic joints and a revolute joint:

- The first prismatic joint is aligned with a vector $\mathbf{u}_j$ normal to the revolute vector $\mathbf{w}_j$, and with $\mathbf{r}_j = \mathbf{r}_G^{i-1}$ and and $\mathbf{r}_{j+1} = \mathbf{r} = \mathbf{r}_G^{i-1} + z_{k1}\mathbf{u}_j$.

- The second prismatic joint is aligned with a vector $\mathbf{v}_j$ normal to $\mathbf{u}_j$ and $\mathbf{w}_j$, and with $\mathbf{r}_j = \mathbf{r}$ and and $\mathbf{r}_{j+1} = \mathbf{s} = \mathbf{r} + z_{k2}\mathbf{v}_j$.
- The revolute joint is attached to the end point of the last prismatic joint and to the center of gravity of the body $i$, with $\mathbf{r}_j = \mathbf{s} = \mathbf{r}_G^{i+1}$, and with the shared vector $\mathbf{w}_j$ as the axis of rotation.

The relative rotation matrix in this case is the same of the revolute joint, since the prismatic joints prevent rotation. As it has been presented in subsection 2.1.1, the rotation matrix of a revolute joint is composed of a constant rotation matrix and a second matrix corresponding to a single rotation around the revolute axis:

$$\mathbf{A}_i^{i-1} = \mathbf{A}^1\mathbf{A}^2 \tag{2.86}$$

In this case, the vectors defining the prismatic joints can be used to determine $\mathbf{A}^1$:

$$\mathbf{A}^1 = \begin{bmatrix}\bar{\mathbf{w}}_j^{i-1} & \bar{\mathbf{u}}_j^{i-1} & \bar{\mathbf{w}}_j^{i-1} \wedge \bar{\mathbf{u}}_j^{i-1}\end{bmatrix}\begin{bmatrix}\bar{\mathbf{w}}_j^{i} & \bar{\mathbf{v}}_j^{i} & \bar{\mathbf{w}}_j^{i} \wedge \bar{\mathbf{v}}_j^{i}\end{bmatrix}^{-1} = \bar{\mathbf{X}}^{i-1}\left(\bar{\mathbf{X}}^{i}\right)^{-1} \tag{2.87}$$

The matrix $\mathbf{A}^2$ can be calculated with the expressions of a single rotation around a vector $\bar{\mathbf{w}}_j^i$.

$$\mathbf{A}^2 = \cos\left(z_{k3}\right)\ \mathbf{I} + \left(1 - \cos\left(z_{k3}\right)\right)\bar{\mathbf{w}}_j^i\left(\bar{\mathbf{w}}_j^i\right)^{\mathrm{T}} + \tilde{\bar{\mathbf{w}}}_j^i \sin\left(z_{k3}\right) \tag{2.88}$$

The position of any point $\mathbf{r}_{j+1}$ and vector $\mathbf{v}_{j+1}$ belonging to body $i$ is given by:

$$\mathbf{r}_j = \mathbf{r}_{i-1} + z_{k1}\mathbf{u}_j + z_{k2}\mathbf{v}_j \tag{2.89}$$
$$\mathbf{r}_{j+1} = \mathbf{r}_j + \mathbf{A}_i\left(\bar{\mathbf{r}}_{j+1}^i - \bar{\mathbf{r}}_j^i\right) \tag{2.90}$$
$$\mathbf{u}_{j+1} = \mathbf{A}_i\bar{\mathbf{u}}_{j+1}^i \tag{2.91}$$

and their velocities can be calculated with:

$$\dot{\mathbf{r}}_j = \dot{\mathbf{r}}_{i-1} + \dot{z}_{k1}\mathbf{u}_j + \boldsymbol{\omega}_{i-1} \wedge z_{k1}\mathbf{u}_j + \dot{z}_{k2}\mathbf{v}_j + \boldsymbol{\omega}_{i-1} \wedge z_{k2}\mathbf{v}_j \tag{2.92}$$
$$\dot{\mathbf{r}}_{j+1} = \dot{\mathbf{r}}_j + \boldsymbol{\omega}_i \wedge \left(\mathbf{r}_{j+1} - \mathbf{r}_j\right) \tag{2.93}$$

The angular velocity of body $i$ has, consequently, the same expression as the revolute joint:

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} + \dot{z}_{k3}\mathbf{w}_j \tag{2.94}$$

## 2.2 Recursive kinematic relations

The kinematic relations derived before are useful to generate the EoM in Cartesian coordinates from the primary (relative) coordinates of the system. In this section, additional recursive relations are developed so as to take the EoM back to relative coordinates. The kinematic equations of this section are provided in the following order: first, the general expressions of the kinematics of the relative motion; and secondly, the particular expressions for the different types of kinematic joints introduced in the previous section.

Since many important MBS like vehicles are not attached to the ground and can move to very large distances from the origin of coordinates, a special treatment for the floating joints (introduced in subsection 2.1.6) is going to be described here, different than the classical approach thought (and better suited) for robotic systems attached to the ground.

The recursive relations are obtained for any set of reference points, and then applied to two particular reference sets, giving as a result two versions of the EoM. The first version, described and identified in [45] as RTdyn0, relates velocities and accelerations of the CoM of consecutive bodies, while the second one, more extended, used in [35, 36, 46, 54, 115] (among others) and named as RTdyn1 in [45], relates velocities and accelerations of the point of consecutive bodies coincident with the global origin of coordinates in each time step.

The notation described in section 2.1 has to be extended here. The advantage of the formulations described in this document relies on writing the kinematics recursively. Thus, the kinematics of each body, $i$, can be written in terms of the kinematics of the previous body in the kinematic chain, $i-1$. In the following derivations, bodies will be numbered with superscripts $i-1$, $i$, $i+1$, etc. while points (vectors) belonging to the bodies will be named with subscripts $i-1$, $i$, $i+1$, etc. and, if needed, with the superscript of the body indicating belonging to that body. For example, point $\mathbf{r}_i$ is the point $i$ (of body $i$) expressed in global coordinates, which is the same as $\mathbf{r}_i^i$ which explicitly indicates that point $i$ belongs to body $i$. The same notation holds for velocities and accelerations, for example, $\dot{\mathbf{r}}_i = \dot{\mathbf{r}}_i^i$ is the velocity of point $i$ of body $i$. On the other hand $\dot{\mathbf{r}}_i^{i,i-1}$ indicates relative velocity of point $i$ of body $i$ with respect to body $i-1$ expressed in the global reference frame.

An upper bar indicates local coordinates in the reference frame of the body, thus point $\bar{\mathbf{r}}_i = \bar{\mathbf{r}}_i^i$ is the point $i$ (of body $i$) expressed in local coordinates of body $i$. It is obvious that $\dot{\bar{\mathbf{r}}}_i = \dot{\bar{\mathbf{r}}}_i^i = \mathbf{0}$, because point $i$ belongs to body $i$, while $\dot{\bar{\mathbf{r}}}_i^{i,i-1}$ is the relative velocity of point $i$ (of body $i$) with respect to the reference frame of body $i-1$ and expressed in the reference frame of body $i-1$.

The notation for angular velocities is the same indicated in section 2.1.

For a point $i$ in body $i$ and a point $i-1$ in body $i-1$ the general expressions of the relative motion are the following,

$$\mathbf{r}_i^i = \mathbf{r}_{i-1}^{i-1} + \left(\mathbf{r}_i^i - \mathbf{r}_{i-1}^{i-1}\right) = \mathbf{r}_{i-1}^{i-1} + \mathbf{A}_{i-1}\left(\bar{\mathbf{r}}_i^{i,i-1} - \bar{\mathbf{r}}_{i-1}^{i-1}\right) \tag{2.95}$$

$$\begin{aligned} \dot{\mathbf{r}}_i^i &= \dot{\mathbf{r}}_{i-1}^{i-1} + \boldsymbol{\omega}_{i-1} \wedge \left(\mathbf{r}_i - \mathbf{r}_{i-1}\right) + \mathbf{A}_{i-1}\left(\dot{\bar{\mathbf{r}}}_i^{i,i-1} - \cancel{\dot{\bar{\mathbf{r}}}_{i-1}^{i-1}}\right) \\ &\Rightarrow \dot{\mathbf{r}}_i = \dot{\mathbf{r}}_{i-1} + \boldsymbol{\omega}_{i-1} \wedge \left(\mathbf{r}_i - \mathbf{r}_{i-1}\right) + \dot{\mathbf{r}}_i^{i,i-1} \end{aligned} \tag{2.96}$$

where the following equations have been employed for (2.96) (see (2.7), (2.8)),

$$\tilde{\boldsymbol{\omega}}_{i-1} = \dot{\mathbf{A}}_{i-1}\mathbf{A}_{i-1}^{\mathrm{T}} \tag{2.97}$$

$$\bar{\mathbf{r}}_i^{i,i-1} - \bar{\mathbf{r}}_{i-1}^{i-1} = \mathbf{A}_{i-1}^{\mathrm{T}}\left(\mathbf{r}_i - \mathbf{r}_{i-1}\right) \tag{2.98}$$

$$\dot{\mathbf{r}}_i^{i,i-1} = \mathbf{A}_{i-1}\dot{\bar{\mathbf{r}}}_i^{i,i-1} \tag{2.99}$$

Using the distribution of velocities, in the rigid body $i-1$ with the reference point $i-1$, see (2.1), the previous expressions can be written in the following way,

$$\dot{\mathbf{r}}_i = \dot{\mathbf{r}}_i^{i} = \dot{\mathbf{r}}_i^{i-1} + \dot{\mathbf{r}}_i^{i,i-1} \tag{2.100a}$$

$$\dot{\mathbf{r}}_i^{i-1} = \dot{\mathbf{r}}_{i-1} + \boldsymbol{\omega}_{i-1} \wedge \left(\mathbf{r}_i - \mathbf{r}_{i-1}\right) \tag{2.100b}$$

where (2.100b) is the velocity of point $i$ as belonging to body $i-1$, and $\dot{\mathbf{r}}_i^{i,i-1}$ is the relative velocity of point $i$ of body $i$ with respect to body $i-1$, both expressed in the global reference frame.

The same relation (2.100) can be verified for any other point $i+1$ of body $i$:

$$\dot{\mathbf{r}}_{i+1} = \dot{\mathbf{r}}_{i+1}^{i} = \dot{\mathbf{r}}_{i+1}^{i-1} + \dot{\mathbf{r}}_{i+1}^{i,i-1} \tag{2.101}$$

The distribution of (global) velocities in the rigid bodies $i$ and $i-1$ allow to relate the absolute velocities of points $i$ and $i+1$ attached to bodies $i$ and $i-1$ respectively:

$$\dot{\mathbf{r}}_{i+1}^{i} = \dot{\mathbf{r}}_i^{i} + \boldsymbol{\omega}_i \wedge \left(\mathbf{r}_{i+1} - \mathbf{r}_i\right) \tag{2.102}$$

$$\dot{\mathbf{r}}_{i+1}^{i-1} = \dot{\mathbf{r}}_i^{i-1} + \boldsymbol{\omega}_{i-1} \wedge \left(\mathbf{r}_{i+1} - \mathbf{r}_i\right) \tag{2.103}$$

and the distribution of relative velocities with respect to body $i-1$:

$$\dot{\bar{\mathbf{r}}}_{i+1}^{i,i-1} = \dot{\bar{\mathbf{r}}}_i^{i,i-1} + \bar{\boldsymbol{\omega}}_i^{i-1} \wedge \left(\bar{\mathbf{r}}_{i+1}^{i-1} - \bar{\mathbf{r}}_i^{i-1}\right) \tag{2.104}$$

where $\bar{\boldsymbol{\omega}}_i^{i-1}$ is the relative angular velocity of body $i$ with respect to body $i-1$ and all the magnitudes are expressed in the reference frame of body $i-1$.

Of course the distribution of relative velocities has to be true also expressing all the magnitudes in the global reference frame,

$$\dot{\mathbf{r}}_{i+1}^{i,i-1} = \dot{\mathbf{r}}_i^{i,i-1} + \boldsymbol{\omega}_i^{i-1} \wedge \left(\mathbf{r}_{i+1} - \mathbf{r}_i\right) \tag{2.105}$$

Adding equations (2.103) and (2.105), subtracting (2.102) and taking into account (2.100) and (2.101),

$$\begin{aligned}\dot{\mathbf{r}}_{i+1}^{i-1} + \dot{\mathbf{r}}_{i+1}^{i,i-1} - \dot{\mathbf{r}}_{i+1}^{i} = \dot{\mathbf{r}}_i^{i-1} + \left(\boldsymbol{\omega}_{i-1} + \boldsymbol{\omega}_i^{i-1} - \boldsymbol{\omega}_i\right) \wedge \left(\mathbf{r}_{i+1} - \mathbf{r}_i\right) + \dot{\mathbf{r}}_i^{i,i-1} - \dot{\mathbf{r}}_i^{i} \Rightarrow \\ \mathbf{0} = \left(\boldsymbol{\omega}_{i-1} + \boldsymbol{\omega}_i^{i-1} - \boldsymbol{\omega}_i\right) \wedge \left(\mathbf{r}_{i+1} - \mathbf{r}_i\right)\end{aligned} \tag{2.106}$$

Since the previous relation has to be satisfied for any points $i$ and $i+1$, the relation for angular velocities holds:

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} + \boldsymbol{\omega}_i^{i-1} \tag{2.107}$$

Taking derivatives in (2.100), a similar relation for accelerations is obtained,

$$\ddot{\mathbf{r}}_i = \ddot{\mathbf{r}}_{i-1} + \dot{\boldsymbol{\omega}}_{i-1} \wedge \left(\mathbf{r}_i - \mathbf{r}_{i-1}\right) + \boldsymbol{\omega}_{i-1} \wedge \left(\dot{\mathbf{r}}_i - \dot{\mathbf{r}}_{i-1}\right) + \ddot{\mathbf{r}}_i^{i,i-1} + \boldsymbol{\omega}_{i-1} \wedge \dot{\mathbf{r}}_i^{i,i-1} \tag{2.108}$$

where the following relation, derived from (2.99), has been used:

$$\frac{\mathrm{d}\dot{\mathbf{r}}_i^{i,i-1}}{\mathrm{d}t} = \dot{\mathbf{A}}_{i-1}\dot{\bar{\mathbf{r}}}_i^{i,i-1} + \mathbf{A}_{i-1}\ddot{\bar{\mathbf{r}}}_i^{i,i-1} = \dot{\mathbf{A}}_{i-1}\mathbf{A}_{i-1}^{\mathrm{T}}\dot{\mathbf{r}}_i^{i,i-1} + \ddot{\mathbf{r}}_i^{i,i-1} \tag{2.109}$$

Replacing (2.96) in (2.108) and taking into account (2.97) and (2.8), the final relation for relative accelerations is obtained,

$$\ddot{\mathbf{r}}_i = \ddot{\mathbf{r}}_i^i = \ddot{\mathbf{r}}_i^{i-1} + \ddot{\mathbf{r}}_i^{i,i-1} + 2\boldsymbol{\omega}_{i-1} \wedge \dot{\mathbf{r}}_i^{i,i-1} \tag{2.110a}$$

$$\ddot{\mathbf{r}}_i^{i-1} = \ddot{\mathbf{r}}_{i-1} + \dot{\boldsymbol{\omega}}_{i-1} \wedge \left(\mathbf{r}_i - \mathbf{r}_{i-1}\right) + \boldsymbol{\omega}_{i-1} \wedge \left[\boldsymbol{\omega}_{i-1} \wedge \left(\mathbf{r}_i - \mathbf{r}_{i-1}\right)\right] \tag{2.110b}$$

where $\ddot{\mathbf{r}}_i^{i-1}$ is the acceleration of point $i$ with body $i-1$ and $\ddot{\mathbf{r}}_i^{i,i-1}$ is the relative acceleration of point $i$ of body $i$ in its relative motion with respect to body $i-1$. The last term $2\boldsymbol{\omega}_{i-1} \wedge \dot{\mathbf{r}}_i^{i,i-1}$ represents the Coriolis or complementary acceleration.

Similarly, taking derivatives in (2.107):

$$\dot{\boldsymbol{\omega}}_i = \dot{\boldsymbol{\omega}}_{i-1} + \mathbf{A}_{i-1}\dot{\bar{\boldsymbol{\omega}}}_i^{i-1} + \dot{\mathbf{A}}_{i-1}\bar{\boldsymbol{\omega}}_i^{i-1} = \dot{\boldsymbol{\omega}}_{i-1} + \boldsymbol{\alpha}_i^{i-1} + \dot{\mathbf{A}}_{i-1}\mathbf{A}_{i-1}^{\mathrm{T}}\boldsymbol{\omega}_i^{i-1} \tag{2.111}$$

where $\dot{\bar{\boldsymbol{\omega}}}_i^{i-1}$ is the relative angular acceleration of body $i$ with respect to body $i-1$ expressed in the reference frame of body $i-1$.

Again, taking into account expressions (2.97) and (2.8), the relative relations for angular accelerations are obtained:

$$\dot{\boldsymbol{\omega}}_i = \dot{\boldsymbol{\omega}}_{i-1} + \boldsymbol{\alpha}_i^{i-1} + \boldsymbol{\omega}_{i-1} \wedge \boldsymbol{\omega}_i^{i-1} \tag{2.112}$$

It is important to remark that $\boldsymbol{\alpha}_i^{i-1}$ stands for the relative acceleration of body $i$ with respect to body $i-1$ in the global reference frame, which is not the temporal derivative of the relative angular velocity, $\boldsymbol{\alpha}_i^{i-1} \neq \dfrac{\mathrm{d}\boldsymbol{\omega}_i^{i-1}}{\mathrm{d}t}$, but the transformation of the relative angular acceleration to global coordinates, $\boldsymbol{\alpha}_i^{i-1} = \mathbf{A}_{i-1}\dot{\bar{\boldsymbol{\omega}}}_i^{i-1} = \mathbf{A}_{i-1}\dfrac{\mathrm{d}\bar{\boldsymbol{\omega}}_i^{i-1}}{\mathrm{d}t}$.

Gathering velocity equations (2.100) together with angular velocity equations (2.107) and acceleration equations (2.110) together with angular acceleration equations (2.112) in the same expressions:

$$\begin{bmatrix}\dot{\mathbf{r}}_i\\ \boldsymbol{\omega}_i\end{bmatrix} = \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_{i-1} - \tilde{\mathbf{r}}_i\\ \mathbf{0} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\dot{\mathbf{r}}_{i-1}\\ \boldsymbol{\omega}_{i-1}\end{bmatrix} + \begin{bmatrix}\dot{\mathbf{r}}_i^{i,i-1}\\ \boldsymbol{\omega}_i^{i-1}\end{bmatrix} \Rightarrow \mathbf{V}_i = \mathbf{B}_i^v\mathbf{V}_{i-1} + \mathbf{b}_i^v\dot{\mathbf{z}}_i \tag{2.113}$$

$$\begin{bmatrix}\ddot{\mathbf{r}}_i\\ \dot{\boldsymbol{\omega}}_i\end{bmatrix} = \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_{i-1} - \tilde{\mathbf{r}}_i\\ \mathbf{0} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\ddot{\mathbf{r}}_{i-1}\\ \dot{\boldsymbol{\omega}}_{i-1}\end{bmatrix} + \begin{bmatrix}\boldsymbol{\omega}_{i-1} \wedge \left[\boldsymbol{\omega}_{i-1} \wedge \left(\mathbf{r}_i - \mathbf{r}_{i-1}\right)\right] + \ddot{\mathbf{r}}_i^{i,i-1} + 2\boldsymbol{\omega}_{i-1} \wedge \dot{\mathbf{r}}_i^{i,i-1}\\ \boldsymbol{\alpha}_i^{i-1} + \boldsymbol{\omega}_{i-1} \wedge \boldsymbol{\omega}_i^{i-1}\end{bmatrix}$$
$$\Rightarrow \dot{\mathbf{V}}_i = \mathbf{B}_i^v\dot{\mathbf{V}}_{i-1} + \mathbf{b}_i^v\ddot{\mathbf{z}}_i + \dot{\mathbf{B}}_i^v\mathbf{V}_{i-1} + \dot{\mathbf{b}}_i^v\dot{\mathbf{z}}_i = \mathbf{B}_i^v\dot{\mathbf{V}}_{i-1} + \mathbf{b}_i^v\ddot{\mathbf{z}}_i + \mathbf{d}_i^v \tag{2.114}$$

Observe that terms $\mathbf{B}_i^v$ and $\dot{\mathbf{B}}_i^v$ are general for any type of joint:

$$\mathbf{B}_i^v = \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_{i-1} - \tilde{\mathbf{r}}_i\\ \mathbf{0} & \mathbf{I}\end{bmatrix}, \tag{2.115}$$

$$\dot{\mathbf{B}}_i^v = \begin{bmatrix}\mathbf{0} & \dot{\tilde{\mathbf{r}}}_{i-1} - \dot{\tilde{\mathbf{r}}}_i\\ \mathbf{0} & \mathbf{0}\end{bmatrix} = \begin{bmatrix}\mathbf{0} & -\tilde{\boldsymbol{\omega}}_{i-1} \wedge \left(\mathbf{r}_i - \mathbf{r}_{i-1}\right) - \dot{\tilde{\mathbf{r}}}_i^{i,i-1}\\ \mathbf{0} & \mathbf{0}\end{bmatrix} \tag{2.116}$$

but terms $\mathbf{b}_i^v\dot{\mathbf{z}}_i$ and $\mathbf{b}_i^v\ddot{\mathbf{z}}_i + \dot{\mathbf{b}}_i^v\dot{\mathbf{z}}_i$ are particular for each type of kinematic joint because they depend on the relative velocities and accelerations $\dot{\mathbf{r}}_i^{i,i-1}$, $\boldsymbol{\omega}_i^{i-1}$, $\ddot{\mathbf{r}}_i^{i,i-1}$ and $\boldsymbol{\alpha}_i^{i-1}$

which have to be parameterized in terms of the relative (joint) coordinates velocities and accelerations, $\dot{\mathbf{z}}_i$, $\ddot{\mathbf{z}}_i$, which is attained in future sections:

$$\mathbf{b}_i^v \dot{\mathbf{z}}_i = \begin{bmatrix} \dot{\mathbf{r}}_i^{i,i-1} \\ \boldsymbol{\omega}_i^{i-1} \end{bmatrix}, \tag{2.117}$$

$$\mathbf{b}_i^v \ddot{\mathbf{z}}_i + \dot{\mathbf{b}}_i^v \dot{\mathbf{z}}_i = \begin{bmatrix} \ddot{\mathbf{r}}_i^{i,i-1} + \boldsymbol{\omega}_{i-1} \wedge \dot{\mathbf{r}}_i^{i,i-1} \\ \boldsymbol{\alpha}_i^{i-1} + \boldsymbol{\omega}_{i-1} \wedge \boldsymbol{\omega}_i^{i-1} \end{bmatrix} \Rightarrow \tag{2.118}$$

$$\mathbf{b}_i^v \ddot{\mathbf{z}}_i + \mathbf{d}_i^v = \begin{bmatrix} \boldsymbol{\omega}_{i-1} \wedge \left[\boldsymbol{\omega}_{i-1} \wedge \left(\mathbf{r}_i - \mathbf{r}_{i-1}\right)\right] + \ddot{\mathbf{r}}_i^{i,i-1} + 2\boldsymbol{\omega}_{i-1} \wedge \dot{\mathbf{r}}_i^{i,i-1} \\ \boldsymbol{\alpha}_i^{i-1} + \boldsymbol{\omega}_{i-1} \wedge \boldsymbol{\omega}_i^{i-1} \end{bmatrix} \tag{2.119}$$

$$\dot{\mathbf{V}}_i = \mathbf{B}_i^v \dot{\mathbf{V}}_{i-1} + \mathbf{b}_i^v \ddot{\mathbf{z}}_i + \dot{\mathbf{B}}_i^v \mathbf{V}_{i-1} + \dot{\mathbf{b}}_i^v \dot{\mathbf{z}}_i = \mathbf{B}_i^v \dot{\mathbf{V}}_{i-1} + \mathbf{b}_i^v \ddot{\mathbf{z}}_i + \mathbf{d}_i^v \tag{2.120}$$

In general, the recursive relations can be simplified in the following set of equations, valid for any type of joint:

$$\mathbf{V}_i = \mathbf{B}_i^v \mathbf{V}_{i-1} + \mathbf{b}_i^v \dot{\mathbf{z}}_i \tag{2.121a}$$

$$\dot{\mathbf{V}}_i = \mathbf{B}_i^v \dot{\mathbf{V}}_{i-1} + \mathbf{b}_i^v \ddot{\mathbf{z}}_i + \mathbf{d}_i^v \tag{2.121b}$$

$$\mathbf{B}_i^v = \begin{bmatrix} \mathbf{I} & \tilde{\mathbf{r}}_{i-1} - \tilde{\mathbf{r}}_i \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{2.121c}$$

$$\dot{\mathbf{B}}_i^v = \begin{bmatrix} \mathbf{0} & \dot{\tilde{\mathbf{r}}}_{i-1} - \dot{\tilde{\mathbf{r}}}_i \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{2.121d}$$

$$\mathbf{d}_i^v = \dot{\mathbf{B}}_i^v \mathbf{V}_{i-1} + \dot{\mathbf{b}}_i^v \dot{\mathbf{z}}_i \tag{2.121e}$$

where for each type of joint, the solely joint-dependent terms needed are $\mathbf{b}_i^v$ and $\dot{\mathbf{b}}_i^v$. Observe that this scheme of calculation allows a general and simple implementation, with a common structure for every joint type and with only one particular term required for each type of joint ($\mathbf{b}_i^v$) and its time derivative ($\dot{\mathbf{b}}_i^v$), whose expressions involve a reduced set of arithmetic operations with the entities that define the joint. For instance, it will be shown for a prismatic joint in section 2.2.2 that the term $\mathbf{b}_i^v$ only requires the position of the vector defining the joint.

The RTdyn0 approach, with the CoM of each body as reference point, will be noted with the arrays of linear and angular velocities and accelerations as $\mathbf{Y}_i$ and $\dot{\mathbf{Y}}_i$ respectively, and the related terms with the superscript $y$. Therefore, the general expressions become:

$$\mathbf{Y}_i = \mathbf{B}_i^y \mathbf{Y}_{i-1} + \mathbf{b}_i^y \dot{\mathbf{z}}_i \tag{2.122a}$$

$$\dot{\mathbf{Y}}_i = \mathbf{B}_i^y \dot{\mathbf{Y}}_{i-1} + \mathbf{b}_i^y \ddot{\mathbf{z}}_i + \mathbf{d}_i^y \tag{2.122b}$$

$$\mathbf{B}_i^y = \begin{bmatrix} \mathbf{I} & \tilde{\mathbf{r}}_G^{i-1} - \tilde{\mathbf{r}}_G^i \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{2.122c}$$

$$\dot{\mathbf{B}}_i^y = \begin{bmatrix} \mathbf{0} & \dot{\tilde{\mathbf{r}}}_G^{i-1} - \dot{\tilde{\mathbf{r}}}_G^i \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{2.122d}$$

$$\mathbf{d}_i^y = \dot{\mathbf{B}}_i^y \mathbf{Y}_{i-1} + \dot{\mathbf{b}}_i^y \dot{\mathbf{z}}_i \tag{2.122e}$$

The recursive relations for the RTdyn1 version, selecting the origin of coordinates as the reference point of each body in each time step, will be identified with the following notation:

$$\mathbf{Z}_i = \mathbf{B}_i^z \mathbf{Z}_{i-1} + \mathbf{b}_i^z \dot{\mathbf{z}}_i \tag{2.123a}$$

$$\dot{\mathbf{Z}}_i = \mathbf{B}_i^z \dot{\mathbf{Z}}_{i-1} + \mathbf{b}_i^z \ddot{\mathbf{z}}_i + \mathbf{d}_i^z \tag{2.123b}$$

$$\mathbf{B}_i^z = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{2.123c}$$

$$\dot{\mathbf{B}}_i^z = \begin{bmatrix} \mathbf{0} & \dot{\tilde{\mathbf{r}}}_0^{i-1} - \dot{\tilde{\mathbf{r}}}_0^{i} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{2.123d}$$

$$\mathbf{d}_i^z = \dot{\mathbf{B}}_i^z \mathbf{Z}_{i-1} + \dot{\mathbf{b}}_i^z \dot{\mathbf{z}}_i \tag{2.123e}$$

Behold that although $\mathbf{B}_i^z$ is always constant and equal to the identity, its time derivative $\dot{\mathbf{B}}_i^z$ is not null. This condition is related to the relative motion of the reference point inside the local reference frame of the body, which makes the velocity of this reference point different depending on the motion of each body.

## 2.2.1 Revolute joint recursive equations

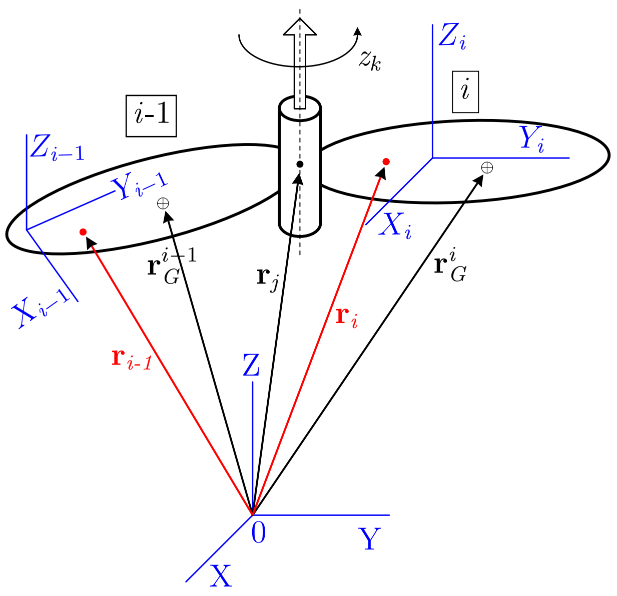


Figure 2.8: Revolute joint.

The angular velocities of both bodies are related by (2.27), recalled here:

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} + \dot{z}_k \mathbf{w}_j \tag{2.124}$$

which is equivalent to equation (2.107) particularized for $\boldsymbol{\omega}_i^{i-1} = \dot{z}_k \mathbf{w}_j$.

Regarding (2.113) and (2.114), it is apparent that the only terms required to calculate the recursive relations between bodies are the relative linear and angular velocities and accelerations of one body with respect to the previous one. These terms can be directly obtained by considering the body $i-1$ as fixed, and applying the distribution of velocities and accelerations of the rigid body to body $i$. Accordingly:

$$\dot{\mathbf{r}}_i = \cancel{\dot{\mathbf{r}}_j} + \boldsymbol{\omega}_i \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) = \dot{z}_k \mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) \tag{2.125}$$

Thus, the relative velocity of point $i$ of body $i$ with respect to body $i-1$ is:

$$\dot{\mathbf{r}}_i^{i,i-1} = \dot{z}_k \mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) \tag{2.126}$$

while the relative angular velocity is,

$$\boldsymbol{\omega}_i^{i-1} = \dot{z}_k \mathbf{w}_j \tag{2.127}$$

Similar relations can be obtained for accelerations. Considering the first body fixed, the acceleration of any point belonging to body $i$ can be obtained as:

$$\begin{aligned}\ddot{\mathbf{r}}_i = \cancel{\ddot{\mathbf{r}}_j} + \boldsymbol{\alpha}_i \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) + \boldsymbol{\omega}_i \wedge \left(\dot{\mathbf{r}}_i - \dot{\mathbf{r}}_j\right) = \boldsymbol{\alpha}_i \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) + \boldsymbol{\omega}_i \wedge \left[\boldsymbol{\omega}_i \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right)\right] = \\ \ddot{z}_k \mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) + \left(\dot{z}_k \mathbf{w}_j\right) \wedge \left[\left(\dot{z}_k \mathbf{w}_j\right) \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right)\right]\end{aligned} \tag{2.128}$$

and therefore:

$$\ddot{\mathbf{r}}_i^{i,i-1} = \ddot{z}_k \mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) + \left(\dot{z}_k \mathbf{w}_j\right) \wedge \left[\left(\dot{z}_k \mathbf{w}_j\right) \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right)\right] \tag{2.129}$$

in which the relative angular acceleration:

$$\boldsymbol{\alpha}_i^{i-1} = \ddot{z}_k \mathbf{w}_j \tag{2.130}$$

Gathering the velocity equations (2.113), (2.126) and (2.127) and the acceleration expressions (2.114), (2.129) and (2.130) in two single expressions,

$$\begin{bmatrix} \dot{\mathbf{r}}_i \\ \boldsymbol{\omega}_i \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \tilde{\mathbf{r}}_{i-1} - \tilde{\mathbf{r}}_i \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{r}}_{i-1} \\ \boldsymbol{\omega}_{i-1} \end{bmatrix} + \begin{bmatrix} \mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) \\ \mathbf{w}_j \end{bmatrix} \dot{z}_k \tag{2.131}$$

$$\begin{aligned}\begin{bmatrix} \ddot{\mathbf{r}}_i \\ \dot{\boldsymbol{\omega}}_i \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \tilde{\mathbf{r}}_{i-1} - \tilde{\mathbf{r}}_i \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{r}}_{i-1} \\ \dot{\boldsymbol{\omega}}_{i-1} \end{bmatrix} + \begin{bmatrix} \mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) \\ \mathbf{w}_j \end{bmatrix} \ddot{z}_k \\ + \begin{bmatrix} \boldsymbol{\omega}_{i-1} \wedge \left[\boldsymbol{\omega}_{i-1} \wedge \left(\mathbf{r}_i - \mathbf{r}_{i-1}\right)\right] + \left(\dot{z}_k \mathbf{w}_j + 2\boldsymbol{\omega}_{i-1}\right) \wedge \left[\dot{z}_k \mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right)\right] \\ \boldsymbol{\omega}_{i-1} \wedge \dot{z}_k \mathbf{w}_j \end{bmatrix}\end{aligned} \tag{2.132}$$

Therefore, recalling (2.121), the following relations for the joint-dependent recursive terms result:

$$\mathbf{b}_i^v = \begin{bmatrix} \mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) \\ \mathbf{w}_j \end{bmatrix} \tag{2.133a}$$

$$\dot{\mathbf{b}}_i^v = \begin{bmatrix} \dot{\mathbf{w}}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) + \mathbf{w}_j \wedge \left(\dot{\mathbf{r}}_i - \dot{\mathbf{r}}_j\right) \\ \dot{\mathbf{w}}_j \end{bmatrix} \tag{2.133b}$$

Observe that from (2.132) a particular expression of $\mathbf{d}_i^v$ for this type of joint can be achieved, which is equal to the general expression for any type of joint:

$$\mathbf{d}_i^v = \begin{bmatrix} \boldsymbol{\omega}_{i-1} \wedge \left[\boldsymbol{\omega}_{i-1} \wedge \left(\mathbf{r}_i - \mathbf{r}_{i-1}\right)\right] + (\dot{z}_k \mathbf{w}_j + 2\boldsymbol{\omega}_{i-1}) \wedge \left[\dot{z}_k \mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right)\right] \\ \boldsymbol{\omega}_{i-1} \wedge \dot{z}_k \mathbf{w}_j \end{bmatrix} = \dot{\mathbf{B}}_i^v \mathbf{V}_{i-1} + \dot{\mathbf{b}}_i^v \dot{\mathbf{z}}_i \tag{2.134}$$

In the calculation of the dynamics of open chain systems, the particular expressions of $\mathbf{d}_i^v$ could be more convenient and faster, but since the term $\dot{\mathbf{b}}_i^v$ is needed, the general expression (2.121e) is more suitable.

The previous recursive expressions can be particularized for different choices of the reference points. Choosing the reference points coincident with the CoM of each body, $\mathbf{r}_i = \mathbf{r}_G^i$, $\mathbf{r}_{i-1} = \mathbf{r}_G^{i-1}$,

$$\mathbf{b}_i^y = \begin{bmatrix} \mathbf{w}_j \wedge \left(\mathbf{r}_G^i - \mathbf{r}_j\right) \\ \mathbf{w}_j \end{bmatrix} \tag{2.135a}$$

$$\dot{\mathbf{b}}_i^y = \begin{bmatrix} \dot{\mathbf{w}}_j \wedge \left(\mathbf{r}_G^i - \mathbf{r}_j\right) + \mathbf{w}_j \wedge \left(\dot{\mathbf{r}}_G^i - \dot{\mathbf{r}}_j\right) \\ \dot{\mathbf{w}}_j \end{bmatrix} \tag{2.135b}$$

The second approach for the recursive relations consists in selecting the points of the bodies coincident with the origin of coordinates at each instant as reference points, $\mathbf{r}_i = \mathbf{r}_0^i = \mathbf{0}$, $\mathbf{r}_{i-1} = \mathbf{r}_0^{i-1} = \mathbf{0}$:

$$\mathbf{b}_i^z = \begin{bmatrix} \mathbf{r}_j \wedge \mathbf{w}_j \\ \mathbf{w}_j \end{bmatrix} \tag{2.136a}$$

$$\dot{\mathbf{b}}_i^z = \begin{bmatrix} \mathbf{r}_j \wedge \dot{\mathbf{w}}_j + \mathbf{w}_j \wedge \left(\dot{\mathbf{r}}_0^i - \dot{\mathbf{r}}_j\right) \\ \dot{\mathbf{w}}_j \end{bmatrix} \tag{2.136b}$$

### 2.2.2 Prismatic joint recursive equations

Recalling (2.32), the angular velocities of two bodies related by a prismatic joint are equal, $\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1}$.

In this case, the relative velocity of a point $i$ of body $i$ with respect to the previous body $i-1$ is straightforward to obtain. Considering body $i-1$ as fixed, a rotation is prevented by the prismatic joint, and only one translation along the axis of the joint is allowed. Moreover, the vector defining the joint is shared between bodies, and since it belongs to body $i-1$ and it is considered as fixed, the vector does not vary and its velocity is null. Therefore, the relative velocity can be expressed as:

$$\dot{\mathbf{r}}_i^{i,i-1} = \dot{z}_k \mathbf{u}_j \tag{2.137}$$

The impossibility of rotation entails that the angular accelerations of body $i$ are also identical to the ones of the previous body $i-1$ in the kinematic chain:

$$\dot{\boldsymbol{\omega}}_i = \dot{\boldsymbol{\omega}}_{i-1} \tag{2.138}$$

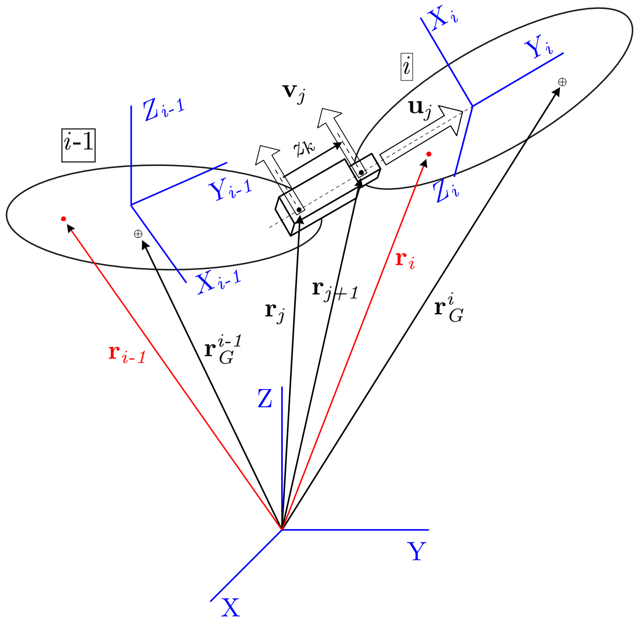


Figure 2.9: Prismatic joint.

The relative acceleration of a point $i$ of body $i$ with respect to body $i-1$ can be obtained applying the same procedure used for velocities. The body $i-1$ is regarded as fixed, so it is the vector defining the axis of the joint. Hence:

$$\ddot{\mathbf{r}}_i^{i,i-1} = \ddot{z}_k \mathbf{u}_j \tag{2.139}$$

The velocity and acceleration recursive relations can be gathered in the following vectors:

$$\begin{bmatrix} \dot{\mathbf{r}}_i \\ \boldsymbol{\omega}_i \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \tilde{\mathbf{r}}_{i-1} - \tilde{\mathbf{r}}_i \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{r}}_{i-1} \\ \boldsymbol{\omega}_{i-1} \end{bmatrix} + \begin{bmatrix} \mathbf{u}_j \\ \mathbf{0} \end{bmatrix} \dot{z}_k \tag{2.140a}$$

$$\begin{bmatrix} \ddot{\mathbf{r}}_i \\ \dot{\boldsymbol{\omega}}_i \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \tilde{\mathbf{r}}_{i-1} - \tilde{\mathbf{r}}_i \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{r}}_{i-1} \\ \dot{\boldsymbol{\omega}}_{i-1} \end{bmatrix} + \begin{bmatrix} \mathbf{u}_j \\ \mathbf{0} \end{bmatrix} \ddot{z}_k + \begin{bmatrix} \boldsymbol{\omega}_{i-1} \wedge \left[\boldsymbol{\omega}_{i-1} \wedge \left(\mathbf{r}_i - \mathbf{r}_{i-1}\right)\right] + 2\boldsymbol{\omega}_{i-1} \wedge \dot{z}_k \mathbf{u}_j \\ \mathbf{0} \end{bmatrix} \tag{2.140b}$$

arriving at the final joint-dependent recursive expressions for this joint type:

$$\mathbf{b}_i^v = \begin{bmatrix} \mathbf{u}_j \\ \mathbf{0} \end{bmatrix} \tag{2.141a}$$

$$\dot{\mathbf{b}}_i^v = \begin{bmatrix} \dot{\mathbf{u}}_j \\ \mathbf{0} \end{bmatrix} \tag{2.141b}$$

The particular expression of $\mathbf{d}_i^v$ obtained from (2.140b) takes the form:

$$\mathbf{d}_i^v = \begin{bmatrix} \boldsymbol{\omega}_{i-1} \wedge \left[\boldsymbol{\omega}_{i-1} \wedge \left(\mathbf{r}_i - \mathbf{r}_{i-1}\right)\right] + 2\boldsymbol{\omega}_{i-1} \wedge \dot{z}_k \mathbf{u}_j \\ \mathbf{0} \end{bmatrix} \tag{2.142}$$

which coincides with the general expression (2.121e).

Note that the expressions of $\mathbf{b}_i^v$ and $\dot{\mathbf{b}}_i^v$ do not involve the position or velocity of the reference point, and therefore they are also valid for RTdyn0 and RTdyn1:

$$\mathbf{b}_i^v = \mathbf{b}_i^y = \mathbf{b}_i^z = \begin{bmatrix} \mathbf{u}_j \\ \mathbf{0} \end{bmatrix} \tag{2.143a}$$

$$\dot{\mathbf{b}}_i^v = \dot{\mathbf{b}}_i^y = \dot{\mathbf{b}}_i^z = \begin{bmatrix} \dot{\mathbf{u}}_j \\ \mathbf{0} \end{bmatrix} \tag{2.143b}$$

### 2.2.3 Cardan joint recursive equations

The recursive expressions for the Cardan joint of Figure 2.3 can be obtained considering two consecutive revolute joints with the respective axes mentioned in section 2.1.3. Thus, the recursive expressions for the first revolute joint, assuming a reference point $\mathbf{r}_{i1}$, lead to,

$$\mathbf{V}_{i1} = \mathbf{B}_{i1}^v \mathbf{V}_{i-1} + \mathbf{b}_{i1}^v \dot{z}_{k1} \tag{2.144a}$$

$$\dot{\mathbf{V}}_{i1} = \mathbf{B}_{i1}^v \dot{\mathbf{V}}_{i-1} + \mathbf{b}_{i1}^v \ddot{z}_{k1} + \mathbf{d}_{i1}^v \tag{2.144b}$$

$$\mathbf{b}_{i1}^v = \begin{bmatrix} \mathbf{w}_j \wedge \left(\mathbf{r}_{i1} - \mathbf{r}_j\right) \\ \mathbf{w}_j \end{bmatrix} \tag{2.144c}$$

$$\mathbf{d}_{i1}^v = \dot{\mathbf{B}}_{i1}^v \mathbf{V}_{i-1} + \dot{\mathbf{b}}_{i1}^v \dot{z}_{k1} \tag{2.144d}$$

And the second revolute joint,

$$\mathbf{V}_{i} = \mathbf{B}_{i2}^v \mathbf{V}_{i1} + \mathbf{b}_{i2}^v \dot{z}_{k2} \tag{2.145a}$$

$$\dot{\mathbf{V}}_{i} = \mathbf{B}_{i2}^v \dot{\mathbf{V}}_{i1} + \mathbf{b}_{i2}^v \ddot{z}_{k2} + \mathbf{d}_{i2}^v \tag{2.145b}$$

$$\mathbf{b}_{i2}^v = \begin{bmatrix} \mathbf{w}_{j+1} \wedge \left(\mathbf{r}_{i} - \mathbf{r}_j\right) \\ \mathbf{w}_{j+1} \end{bmatrix} \tag{2.145c}$$

$$\mathbf{d}_{i2}^v = \dot{\mathbf{B}}_{i2}^v \mathbf{V}_{i1} + \dot{\mathbf{b}}_{i2}^v \dot{z}_{k2} \tag{2.145d}$$

Therefore,

$$\mathbf{V}_i = \mathbf{B}_{i2}^v \mathbf{V}_{i1} + \mathbf{b}_{i2}^v \dot{z}_{k2} = \mathbf{B}_{i2}^v \left(\mathbf{B}_{i1}^v \mathbf{V}_{i-1} + \mathbf{b}_{i1}^v \dot{z}_{k1}\right) + \mathbf{b}_{i2}^v \dot{z}_{k2} = \mathbf{B}_i^v \mathbf{V}_{i-1} + \mathbf{b}_i^v \dot{\mathbf{z}}_i \tag{2.146a}$$

$$\begin{split}\dot{\mathbf{V}}_i = \mathbf{B}_{i2}^v \dot{\mathbf{V}}_{i1} + \mathbf{b}_{i2}^v \ddot{z}_{k2} + \mathbf{d}_{i2}^v = \mathbf{B}_{i2}^v \left(\mathbf{B}_{i1}^v \dot{\mathbf{V}}_{i-1} + \mathbf{b}_{i1}^v \ddot{z}_{k1} + \mathbf{d}_{i1}^v\right) + \mathbf{b}_{i2}^v \ddot{z}_{k2} + \mathbf{d}_{i2}^v = \\ \mathbf{B}_i^v \dot{\mathbf{V}}_{i-1} + \mathbf{b}_i^v \ddot{\mathbf{z}}_i + \mathbf{d}_i^v\end{split} \tag{2.146b}$$

where, $\dot{\mathbf{z}}_i = \begin{bmatrix} \dot{z}_{k1} & \dot{z}_{k2} \end{bmatrix}^{\mathrm{T}}$, $\ddot{\mathbf{z}}_i = \begin{bmatrix} \ddot{z}_{k1} & \ddot{z}_{k2} \end{bmatrix}^{\mathrm{T}}$.

The term $\mathbf{B}_i^v$ can be calculated from (2.146b):

$$\mathbf{B}_i^v = \mathbf{B}_{i2}^v \mathbf{B}_{i1}^v = \begin{bmatrix} \mathbf{I} & \tilde{\mathbf{r}}_{i1} - \tilde{\mathbf{r}}_i \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \tilde{\mathbf{r}}_{i-1} - \tilde{\mathbf{r}}_{i1} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \tilde{\mathbf{r}}_{i-1} - \tilde{\mathbf{r}}_i \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{2.147}$$

$$\tag{2.148}$$

Similarly, $\mathbf{b}_i^v$ is:

$$\mathbf{b}_i^v = \begin{bmatrix}\mathbf{B}_{i2}^v\mathbf{b}_{i1}^v & \mathbf{b}_{i2}^v\end{bmatrix} = \begin{bmatrix}\mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) & \mathbf{w}_{j+1} \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) \\ \mathbf{w}_j & \mathbf{w}_{j+1}\end{bmatrix} \tag{2.149}$$

and its time derivative:

$$\begin{aligned}\dot{\mathbf{b}}_i^v = &\left[\left(\mathbf{B}_{i2}^v\dot{\mathbf{b}}_{i1}^v + \dot{\mathbf{B}}_{i2}^v\mathbf{b}_{i1}^v\right) \quad \dot{\mathbf{b}}_{i2}^v\right] = \\ &\begin{bmatrix}\dot{\mathbf{w}}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) + \mathbf{w}_j \wedge \left(\dot{\mathbf{r}}_i - \dot{\mathbf{r}}_j\right) & \dot{\mathbf{w}}_{j+1} \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) + \mathbf{w}_{j+1} \wedge \left(\dot{\mathbf{r}}_i - \dot{\mathbf{r}}_j\right) \\ \dot{\mathbf{w}}_j & \dot{\mathbf{w}}_{j+1}\end{bmatrix}\end{aligned} \tag{2.150}$$

Using (2.146a), (2.146b), (2.144d) and (2.145d), the general expression of $\mathbf{d}_i^v$ (2.121e) can be reached:

$$\begin{aligned}\mathbf{d}_i^v =&\mathbf{B}_{i2}^v\mathbf{d}_{i1}^v + \mathbf{d}_{i2}^v = \mathbf{B}_{i2}^v\left(\dot{\mathbf{B}}_{i1}^v\mathbf{V}_{i-1} + \dot{\mathbf{b}}_{i1}^v\dot{z}_{k1}\right) + \dot{\mathbf{B}}_{i2}^v\mathbf{V}_{i1} + \dot{\mathbf{b}}_{i2}^v\dot{z}_{k2} = \\ &\mathbf{B}_{i2}^v\left(\dot{\mathbf{B}}_{i1}^v\mathbf{V}_{i-1} + \dot{\mathbf{b}}_{i1}^v\dot{z}_{k1}\right) + \dot{\mathbf{B}}_{i2}^v\left(\mathbf{B}_{i1}^v\mathbf{V}_{i-1} + \mathbf{b}_{i1}^v\dot{z}_{k1}\right) + \dot{\mathbf{b}}_{i2}^v\dot{z}_{k2} = \\ &\left(\mathbf{B}_{i2}^v\dot{\mathbf{B}}_{i1}^v + \dot{\mathbf{B}}_{i2}^v\mathbf{B}_{i1}^v\right)\mathbf{V}_{i-1} + \left(\mathbf{B}_{i2}^v\dot{\mathbf{b}}_{i1}^v + \dot{\mathbf{B}}_{i2}^v\mathbf{b}_{i1}^v\right)\dot{z}_{k1} + \dot{\mathbf{b}}_{i2}^v\dot{z}_{k2} = \\ &\left(\mathbf{B}_{i2}^v\dot{\mathbf{B}}_{i1}^v + \dot{\mathbf{B}}_{i2}^v\mathbf{B}_{i1}^v\right)\mathbf{V}_{i-1} + \left[\left(\mathbf{B}_{i2}^v\dot{\mathbf{b}}_{i1}^v + \dot{\mathbf{B}}_{i2}^v\mathbf{b}_{i1}^v\right) \quad \dot{\mathbf{b}}_{i2}^v\right]\dot{\mathbf{z}}_i = \\ &\dot{\mathbf{B}}_i^v\mathbf{V}_{i-1} + \dot{\mathbf{b}}_i^v\dot{\mathbf{z}}_i\end{aligned} \tag{2.151}$$

The particularization of the previous expressions for RTdyn0 with $\mathbf{r}_i = \mathbf{r}_G^i$, $\mathbf{r}_{i-1} = \mathbf{r}_G^{i-1}$ is forthright:

$$\mathbf{b}_i^y = \begin{bmatrix}\mathbf{w}_j \wedge \left(\mathbf{r}_G^i - \mathbf{r}_j\right) & \mathbf{w}_{j+1} \wedge \left(\mathbf{r}_G^i - \mathbf{r}_j\right) \\ \mathbf{w}_j & \mathbf{w}_{j+1}\end{bmatrix} \tag{2.152a}$$

$$\dot{\mathbf{b}}_i^y = \begin{bmatrix}\dot{\mathbf{w}}_j \wedge \left(\mathbf{r}_G^i - \mathbf{r}_j\right) + \mathbf{w}_j \wedge \left(\dot{\mathbf{r}}_G^i - \dot{\mathbf{r}}_j\right) & \dot{\mathbf{w}}_{j+1} \wedge \left(\mathbf{r}_G^i - \mathbf{r}_j\right) + \mathbf{w}_{j+1} \wedge \left(\dot{\mathbf{r}}_G^i - \dot{\mathbf{r}}_j\right) \\ \dot{\mathbf{w}}_j & \dot{\mathbf{w}}_{j+1}\end{bmatrix} \tag{2.152b}$$

The joint-dependent recursive expressions for RTdyn1, with $\mathbf{r}_i = \mathbf{r}_{i-1} = \mathbf{0}$, become:

$$\mathbf{b}_i^z = \begin{bmatrix}\mathbf{r}_j \wedge \mathbf{w}_j & \mathbf{r}_j \wedge \mathbf{w}_{j+1} \\ \mathbf{w}_j & \mathbf{w}_{j+1}\end{bmatrix} \tag{2.153a}$$

$$\dot{\mathbf{b}}_i^z = \begin{bmatrix}-\dot{\mathbf{w}}_j \wedge \mathbf{r}_j + \mathbf{w}_j \wedge \left(\dot{\mathbf{r}}_0^i - \dot{\mathbf{r}}_j\right) & -\dot{\mathbf{w}}_{j+1} \wedge \mathbf{r}_j + \mathbf{w}_{j+1} \wedge \left(\dot{\mathbf{r}}_0^i - \dot{\mathbf{r}}_j\right) \\ \dot{\mathbf{w}}_j & \dot{\mathbf{w}}_{j+1}\end{bmatrix} \tag{2.153b}$$

### 2.2.4 Cylindrical joint recursive equations

The angular velocities of both bodies are related by (2.55), recalled here:

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} + \dot{z}_{k2}\mathbf{w}_j \tag{2.154}$$

which is equivalent to equation (2.107) particularized for $\boldsymbol{\omega}_i^{i-1} = \dot{z}_{k2}\mathbf{w}_j$.

The relative velocity of a point $i$ belonging to body $i$ with respect to body $i-1$ can be obtained, once again, considering body $i-1$ as fixed, but is easier to calculate it as the combination of the relative velocities of a revolute and a prismatic joint, yielding:

$$\dot{\mathbf{r}}_i^{i,i-1} = \dot{\mathbf{r}}_j^{i,i-1} + \boldsymbol{\omega}_i^{i-1} \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) = \dot{z}_{k1}\mathbf{w}_j + \dot{z}_{k2}\mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) \tag{2.155}$$

The relative angular acceleration is, then, the addition of the contributions of a revolute and a prismatic joint:

$$\boldsymbol{\alpha}_i^{i-1} = \mathbf{w}_j \ddot{z}_{k2} \tag{2.156}$$

Analogously, the relative linear acceleration takes the form:

$$\dot{\mathbf{r}}_i^{i,i-1} = \dot{z}_{k1}\mathbf{w}_j + \dot{z}_{k2}\mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) \tag{2.157}$$

Gathering the velocity equations (2.113), (2.155) and (2.154) and the acceleration expressions (2.114), (2.157) and (2.156) in two single expressions,

$$\begin{bmatrix} \dot{\mathbf{r}}_i \\ \boldsymbol{\omega}_i \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \tilde{\mathbf{r}}_{i-1} - \tilde{\mathbf{r}}_i \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{r}}_{i-1} \\ \boldsymbol{\omega}_{i-1} \end{bmatrix} + \begin{bmatrix} \mathbf{w}_j & \mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) \\ \mathbf{0} & \mathbf{w}_j \end{bmatrix} \dot{\mathbf{z}}_i \tag{2.158}$$

$$\begin{gathered} \begin{bmatrix} \ddot{\mathbf{r}}_i \\ \dot{\boldsymbol{\omega}}_i \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \tilde{\mathbf{r}}_{i-1} - \tilde{\mathbf{r}}_i \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{r}}_{i-1} \\ \dot{\boldsymbol{\omega}}_{i-1} \end{bmatrix} + \begin{bmatrix} \mathbf{w}_j & \mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) \\ \mathbf{0} & \mathbf{w}_j \end{bmatrix} \ddot{\mathbf{z}}_i + \\ \begin{bmatrix} \boldsymbol{\omega}_{i-1} \wedge \left[\boldsymbol{\omega}_{i-1} \wedge \left(\mathbf{r}_i - \mathbf{r}_{i-1}\right)\right] + \left(\dot{z}_{k2}\mathbf{w}_j + 2\boldsymbol{\omega}_{i-1}\right) \wedge \left[\dot{z}_{k2}\mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right)\right] + 2\boldsymbol{\omega}_{i-1} \wedge \dot{z}_{k1}\mathbf{w}_j \\ \boldsymbol{\omega}_{i-1} \wedge \dot{z}_{k2}\mathbf{w}_j \end{bmatrix} \end{gathered} \tag{2.159}$$

where $\dot{\mathbf{z}}_i = \begin{bmatrix} \dot{z}_{k1} & \dot{z}_{k2} \end{bmatrix}^{\mathrm{T}}$, $\ddot{\mathbf{z}}_i = \begin{bmatrix} \ddot{z}_{k1} & \ddot{z}_{k2} \end{bmatrix}^{\mathrm{T}}$.

From (2.158), the joint-dependent recursive relations for the cylindrical joint can be identified:

$$\mathbf{b}_i^v = \begin{bmatrix} \mathbf{w}_j & \mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) \\ \mathbf{0} & \mathbf{w}_j \end{bmatrix} \tag{2.160a}$$

$$\dot{\mathbf{b}}_i^v = \begin{bmatrix} \dot{\mathbf{w}}_j & \dot{\mathbf{w}}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) + \mathbf{w}_j \wedge \left(\dot{\mathbf{r}}_i - \dot{\mathbf{r}}_j\right) \\ \mathbf{0} & \dot{\mathbf{w}}_j \end{bmatrix} \tag{2.160b}$$

The previous recursive expressions can be particularized for different selections of the reference points. Choosing the reference points coincident with the CoM of each body (RTdyn0), $\mathbf{r}_i = \mathbf{r}_G^i$, $\mathbf{r}_{i-1} = \mathbf{r}_G^{i-1}$,

$$\mathbf{b}_i^y = \begin{bmatrix} \mathbf{w}_j & \mathbf{w}_j \wedge \left(\mathbf{r}_G^i - \mathbf{r}_j\right) \\ \mathbf{0} & \mathbf{w}_j \end{bmatrix} \tag{2.161a}$$

$$\dot{\mathbf{b}}_i^y = \begin{bmatrix} \dot{\mathbf{w}}_j & \dot{\mathbf{w}}_j \wedge \left(\mathbf{r}_G^i - \mathbf{r}_j\right) + \mathbf{w}_j \wedge \left(\dot{\mathbf{r}}_G^i - \dot{\mathbf{r}}_j\right) \\ \mathbf{0} & \dot{\mathbf{w}}_j \end{bmatrix} \tag{2.161b}$$

The application of $\mathbf{r}_i = \mathbf{r}_0^i = \mathbf{0}$, $\mathbf{r}_{i-1} = \mathbf{r}_0^{i-1} = \mathbf{0}$ of the RTdyn1 approach leads to the following recursive relations:

$$\mathbf{b}_i^z = \begin{bmatrix} \mathbf{w}_j & \mathbf{r}_j \wedge \mathbf{w}_j \\ \mathbf{0} & \mathbf{w}_j \end{bmatrix} \tag{2.162a}$$

$$\dot{\mathbf{b}}_i^z = \begin{bmatrix} \dot{\mathbf{w}}_j & \mathbf{r}_j \wedge \dot{\mathbf{w}}_j + \mathbf{w}_j \wedge \left(\dot{\mathbf{r}}_0^i - \dot{\mathbf{r}}_j\right) \\ \mathbf{0} & \dot{\mathbf{w}}_j \end{bmatrix} \tag{2.162b}$$

## 2.2.5 Spherical joint recursive equations

Figure 2.5 represents two bodies, $i-1$ and $i$, connected through a spherical joint. According to section 2.1.5, the angular velocities of both bodies are related by (2.60),(2.76) and (2.77).

The relative velocity of a point $i$ contained in body $i$ with respect to the previous body can be easily obtained considering body $i-1$ as fixed. The shared point $j$ between the bodies is fixed as well, and its velocity null:

$$\dot{\mathbf{r}}_i^{i,i-1} = \boldsymbol{\omega}_i^{i-1} \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) = 2\mathbf{A}_{i-1}\bar{\mathbf{E}}\dot{\mathbf{p}} \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) = 2\mathbf{E}\dot{\mathbf{p}} \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) \tag{2.163}$$

Angular accelerations can be determined from (2.60),(2.76) and (2.77):

$$\dot{\boldsymbol{\omega}}_i = \dot{\boldsymbol{\omega}}_{i-1} + 2\mathbf{A}_{i-1}\left(\bar{\mathbf{E}}\ddot{\mathbf{p}} + \dot{\bar{\mathbf{E}}}\dot{\mathbf{p}}\right) + 2\dot{\mathbf{A}}_{i-1}\bar{\mathbf{E}}\dot{\mathbf{p}} \tag{2.164}$$

since,

$$\dot{\bar{\mathbf{E}}}\dot{\mathbf{p}} = \begin{bmatrix} -\dot{\mathbf{e}} & \dot{\tilde{\mathbf{e}}} + \dot{e}_0\mathbf{I} \end{bmatrix} \begin{bmatrix} \dot{e}_0 \\ \dot{\mathbf{e}} \end{bmatrix} = -\dot{e}_0\dot{\mathbf{e}} + \dot{\tilde{\mathbf{e}}}\dot{\mathbf{e}} + \dot{e}_0\dot{\mathbf{e}} = \mathbf{0} \tag{2.165}$$

and,

$$2\dot{\mathbf{A}}_{i-1}\bar{\mathbf{E}}\dot{\mathbf{p}} = \dot{\mathbf{A}}_{i-1}\mathbf{A}_{i-1}^{\mathrm{T}}2\mathbf{A}_{i-1}\bar{\mathbf{E}}\dot{\mathbf{p}} = \tilde{\boldsymbol{\omega}}_{i-1}2\mathbf{E}\dot{\mathbf{p}} = \boldsymbol{\omega}_{i-1} \wedge 2\mathbf{E}\dot{\mathbf{p}} \tag{2.166}$$

therefore the final expression for the recursive angular accelerations takes the following simple form,

$$\dot{\boldsymbol{\omega}}_i = \dot{\boldsymbol{\omega}}_{i-1} + 2\mathbf{E}\ddot{\mathbf{p}} + \boldsymbol{\omega}_{i-1} \wedge 2\mathbf{E}\dot{\mathbf{p}} \tag{2.167}$$

Observe that the previous expression fits the general expression (2.112),

$$\dot{\boldsymbol{\omega}}_i = \dot{\boldsymbol{\omega}}_{i-1} + \boldsymbol{\alpha}_i^{i-1} + \boldsymbol{\omega}_{i-1} \wedge \boldsymbol{\omega}_i^{i-1} \tag{2.168a}$$

$$\boldsymbol{\alpha}_i^{i-1} = \mathbf{A}_{i-1}\dot{\bar{\boldsymbol{\omega}}}_i^{i-1} = 2\mathbf{A}_{i-1}\bar{\mathbf{E}}\ddot{\mathbf{p}} = 2\mathbf{E}\ddot{\mathbf{p}} \tag{2.168b}$$

$$\boldsymbol{\omega}_i^{i-1} = 2\mathbf{A}_{i-1}\bar{\mathbf{E}}\dot{\mathbf{p}} = 2\mathbf{E}\dot{\mathbf{p}} \tag{2.168c}$$

thus finally arriving at expression (2.112) particularized for the spherical joint.

Applying the same procedure used for relative velocities, the relative linear acceleration of a point $i$ belonging to a body $i$ with respect to the previous one $i-1$ yields:

$$\begin{aligned} \ddot{\mathbf{r}}_i^{i,i-1} = \boldsymbol{\alpha}_i^{i-1} \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) + \boldsymbol{\omega}_i^{i-1} \wedge \left(\boldsymbol{\omega}_i^{i-1} \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right)\right) = \\ 2\mathbf{E}\ddot{\mathbf{p}} \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) + 2\mathbf{E}\dot{\mathbf{p}} \wedge \left(2\mathbf{E}\dot{\mathbf{p}} \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right)\right) \end{aligned} \tag{2.169}$$

The velocity and acceleration recursive relations can be gathered in the following vectors.

$$\begin{bmatrix}\dot{\mathbf{r}}_i\\ \boldsymbol{\omega}_i\end{bmatrix} = \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_{i-1}-\tilde{\mathbf{r}}_i\\ \mathbf{0} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\dot{\mathbf{r}}_{i-1}\\ \boldsymbol{\omega}_{i-1}\end{bmatrix} + \begin{bmatrix}2\left(\tilde{\mathbf{r}}_j-\tilde{\mathbf{r}}_i\right)\mathbf{E}\\ 2\mathbf{E}\end{bmatrix}\dot{\bar{\mathbf{p}}} \tag{2.170a}$$

$$\begin{bmatrix}\ddot{\mathbf{r}}_i\\ \dot{\boldsymbol{\omega}}_i\end{bmatrix} = \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_{i-1}-\tilde{\mathbf{r}}_i\\ \mathbf{0} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\ddot{\mathbf{r}}_{i-1}\\ \dot{\boldsymbol{\omega}}_{i-1}\end{bmatrix} + \begin{bmatrix}2\left(\tilde{\mathbf{r}}_j-\tilde{\mathbf{r}}_i\right)\mathbf{E}\\ 2\mathbf{E}\end{bmatrix}\ddot{\bar{\mathbf{p}}} + \begin{bmatrix}\boldsymbol{\omega}_{i-1}\wedge\left(\boldsymbol{\omega}_{i-1}\wedge\left(\mathbf{r}_i-\mathbf{r}_{i-1}\right)\right)\\ \mathbf{0}\end{bmatrix} +$$
$$\begin{bmatrix}2\mathbf{E}\dot{\bar{\mathbf{p}}}\wedge\left(2\mathbf{E}\dot{\bar{\mathbf{p}}}\wedge\left(\mathbf{r}_i-\mathbf{r}_j\right)\right)+2\boldsymbol{\omega}_{i-1}\wedge\left(2\mathbf{E}\dot{\bar{\mathbf{p}}}\wedge\left(\mathbf{r}_i-\mathbf{r}_j\right)\right)\\ \boldsymbol{\omega}_{i-1}\wedge 2\mathbf{E}\dot{\bar{\mathbf{p}}}\end{bmatrix} \tag{2.170b}$$

Therefore, the joint-dependent recursive expressions for this joint become:

$$\mathbf{b}_i^v = \begin{bmatrix}2\left(\tilde{\mathbf{r}}_j-\tilde{\mathbf{r}}_i\right)\mathbf{E}\\ 2\mathbf{E}\end{bmatrix} \tag{2.171a}$$

$$\dot{\mathbf{b}}_i^v = \begin{bmatrix}2\left(\dot{\tilde{\mathbf{r}}}_j-\dot{\tilde{\mathbf{r}}}_i\right)\mathbf{E}+2\left(\tilde{\mathbf{r}}_j-\tilde{\mathbf{r}}_i\right)\dot{\mathbf{E}}\\ 2\dot{\mathbf{E}}\end{bmatrix} \tag{2.171b}$$

$$\mathbf{d}_i^v = \begin{bmatrix}\boldsymbol{\omega}_{i-1}\wedge\left(\boldsymbol{\omega}_{i-1}\wedge\left(\mathbf{r}_i-\mathbf{r}_{i-1}\right)\right)+\left(2\boldsymbol{\omega}_{i-1}+2\mathbf{E}\dot{\bar{\mathbf{p}}}\right)\wedge\left(2\mathbf{E}\dot{\bar{\mathbf{p}}}\wedge\left(\mathbf{r}_i-\mathbf{r}_j\right)\right)\\ \boldsymbol{\omega}_{i-1}\wedge 2\mathbf{E}\dot{\bar{\mathbf{p}}}\end{bmatrix} \tag{2.171c}$$

with $\mathbf{z}_i = \bar{\mathbf{p}}$ for the spherical joint.

The term $\dot{\mathbf{b}}_i^v$ can be simplified using the expressions of the angular velocities in the local reference frame of body $i$:

$$\dot{\mathbf{b}}_i^v = \begin{bmatrix}2\left(\dot{\tilde{\mathbf{r}}}_j-\dot{\tilde{\mathbf{r}}}_i\right)\mathbf{E}+2\left(\tilde{\mathbf{r}}_j-\tilde{\mathbf{r}}_i\right)\left(\tilde{\boldsymbol{\omega}}_{i-1}\mathbf{E}+\mathbf{A}_{i-1}\dot{\bar{\mathbf{E}}}\right)\\ 2\left(\tilde{\boldsymbol{\omega}}_{i-1}\mathbf{E}+\mathbf{A}_{i-1}\dot{\bar{\mathbf{E}}}\right)\end{bmatrix} = \begin{bmatrix}2\tilde{\boldsymbol{\omega}}_i\left(\tilde{\mathbf{r}}_j-\tilde{\mathbf{r}}_i\right)\mathbf{E}\\ 2\tilde{\boldsymbol{\omega}}_i\mathbf{E}\end{bmatrix} \tag{2.172}$$

where the identities $\mathbf{E} = \mathbf{A}_{i-1}\bar{\mathbf{E}}$ and $\dot{\mathbf{A}}_{i-1}\bar{\mathbf{E}} = \dot{\mathbf{A}}_{i-1}\mathbf{A}_{i-1}^{\mathrm{T}}\mathbf{A}_{i-1}\bar{\mathbf{E}} = \tilde{\boldsymbol{\omega}}_{i-1}\mathbf{E}$ have been used and the last form of $\dot{\mathbf{b}}_i^v$ has been obtained by identifying terms in $\dot{\mathbf{b}}_i^v\dot{\bar{\mathbf{p}}}$,

$$\dot{\mathbf{b}}_i^v\dot{\bar{\mathbf{p}}} = \begin{bmatrix}2\mathbf{E}\dot{\bar{\mathbf{p}}}\wedge\left(2\mathbf{E}\dot{\bar{\mathbf{p}}}\wedge\left(\mathbf{r}_i-\mathbf{r}_j\right)\right)+\boldsymbol{\omega}_{i-1}\wedge\left(2\mathbf{E}\dot{\bar{\mathbf{p}}}\wedge\left(\mathbf{r}_i-\mathbf{r}_j\right)\right)\\ \boldsymbol{\omega}_{i-1}\wedge 2\mathbf{E}\dot{\bar{\mathbf{p}}}\end{bmatrix} =$$
$$\begin{bmatrix}\left(\boldsymbol{\omega}_{i-1}+2\mathbf{E}\dot{\bar{\mathbf{p}}}\right)\wedge\left(2\mathbf{E}\dot{\bar{\mathbf{p}}}\wedge\left(\mathbf{r}_i-\mathbf{r}_j\right)\right)\\ \boldsymbol{\omega}_{i-1}\wedge 2\mathbf{E}\dot{\bar{\mathbf{p}}}\end{bmatrix} = \begin{bmatrix}\boldsymbol{\omega}_i\wedge\left(2\mathbf{E}\dot{\bar{\mathbf{p}}}\wedge\left(\mathbf{r}_i-\mathbf{r}_j\right)\right)\\ \boldsymbol{\omega}_i\wedge 2\mathbf{E}\dot{\bar{\mathbf{p}}}\end{bmatrix} = \tag{2.173}$$
$$\begin{bmatrix}\boldsymbol{\omega}_i\wedge\left(\left(\tilde{\mathbf{r}}_j-\tilde{\mathbf{r}}_i\right)2\mathbf{E}\dot{\bar{\mathbf{p}}}\right)\\ \boldsymbol{\omega}_i\wedge 2\mathbf{E}\dot{\bar{\mathbf{p}}}\end{bmatrix} = \begin{bmatrix}2\tilde{\boldsymbol{\omega}}_i\left(\tilde{\mathbf{r}}_j-\tilde{\mathbf{r}}_i\right)\mathbf{E}\\ 2\tilde{\boldsymbol{\omega}}_i\mathbf{E}\end{bmatrix}\dot{\bar{\mathbf{p}}}$$

Observe that, since $\dot{\bar{\mathbf{E}}}\dot{\bar{\mathbf{p}}} = \mathbf{0}$, the general expression of $\mathbf{d}_i^v$ (2.121e) is analog to (2.171c).

Choosing the reference points coincident with the CoM of each body (RTdyn0), $\mathbf{r}_i = \mathbf{r}_G^i$, $\mathbf{r}_{i-1} = \mathbf{r}_G^{i-1}$,

$$\mathbf{b}_i^y = \begin{bmatrix}2\left(\tilde{\mathbf{r}}_j-\tilde{\mathbf{r}}_G^i\right)\mathbf{E}\\ 2\mathbf{E}\end{bmatrix} \tag{2.174a}$$

$$\dot{\mathbf{b}}_i^y = \begin{bmatrix}2\left(\dot{\tilde{\mathbf{r}}}_j-\dot{\tilde{\mathbf{r}}}_G^i\right)\mathbf{E}+2\left(\tilde{\mathbf{r}}_j-\tilde{\mathbf{r}}_G^i\right)\dot{\mathbf{E}}\\ 2\dot{\mathbf{E}}\end{bmatrix} = \begin{bmatrix}2\tilde{\boldsymbol{\omega}}_i\left(\tilde{\mathbf{r}}_j-\tilde{\mathbf{r}}_G^i\right)\mathbf{E}\\ 2\tilde{\boldsymbol{\omega}}_i\mathbf{E}\end{bmatrix} \tag{2.174b}$$

The expressions for the second approach (RTdyn1) with $\mathbf{r}_i = \mathbf{r}_0^i = \mathbf{0}$, $\mathbf{r}_{i-1} = \mathbf{r}_0^{i-1} = \mathbf{0}$ result simpler,

$$\mathbf{b}_i^z = \begin{bmatrix} 2\tilde{\mathbf{r}}_j\mathbf{E} \\ 2\mathbf{E} \end{bmatrix} \tag{2.175a}$$

$$\dot{\mathbf{b}}_i^z = \begin{bmatrix} 2\left(\dot{\tilde{\mathbf{r}}}_j - \dot{\tilde{\mathbf{r}}}_0^i\right)\mathbf{E} + 2\tilde{\mathbf{r}}_j\dot{\mathbf{E}} \\ 2\dot{\mathbf{E}} \end{bmatrix} = \begin{bmatrix} 2\tilde{\boldsymbol{\omega}}_i\tilde{\mathbf{r}}_j\mathbf{E} \\ 2\tilde{\boldsymbol{\omega}}_i\mathbf{E} \end{bmatrix} \tag{2.175b}$$

### 2.2.6 Floating joint recursive equations

As introduced in section 2.1.6, the floating joint is regarded as a formalism to relate the motion of a free-in-space MBS to the ground by means of a “fictional” joint. Three elemental prismatic joints and an elemental spherical joint represent the 6 DoF of the rigid body.

Taking into account that linear and angular velocities and accelerations of the ground are null, the recursive velocities of body $i$ can be determined using the expressions of the prismatic and spherical joints:

$$\mathbf{V}_i = \mathbf{b}_i^v\dot{\mathbf{z}}_i = \begin{bmatrix} \dot{\mathbf{r}}_i \\ \boldsymbol{\omega}_i \end{bmatrix} = \begin{bmatrix} \mathbf{I}_3 & 2\left(\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_i\right)\mathbf{E} \\ \mathbf{0} & 2\mathbf{E} \end{bmatrix}\dot{\mathbf{z}}_i \tag{2.176a}$$

$$\dot{\mathbf{V}}_i = \mathbf{b}_i^v\ddot{\mathbf{z}}_i + \mathbf{d}_i^v = \begin{bmatrix} \ddot{\mathbf{r}}_i \\ \dot{\boldsymbol{\omega}}_1 \end{bmatrix} = \begin{bmatrix} \mathbf{I}_3 & 2\left(\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_i\right)\mathbf{E} \\ \mathbf{0} & 2\mathbf{E} \end{bmatrix}\ddot{\mathbf{z}}_i + \begin{bmatrix} 2\mathbf{E}\dot{\bar{\mathbf{p}}}_i \wedge \left(2\mathbf{E}\dot{\bar{\mathbf{p}}}_i \wedge \left(\mathbf{r}_i - \mathbf{r}_G^i\right)\right) \\ \mathbf{0} \end{bmatrix} \tag{2.176b}$$

where $\mathbf{z}_i = \begin{bmatrix} \mathbf{r}_G^i & \dot{\bar{\mathbf{p}}}_i \end{bmatrix}^{\mathrm{T}}$ and $\mathbf{B}_i^v = \mathbf{I}_6$. Note also that the vectors of the prismatic joints composing this joint are aligned with the axis of the global reference frame, then $[\mathbf{u}_j\,\mathbf{v}_j\,\mathbf{w}_j] = \mathbf{I}_3$.

The time derivatives of the recursive terms describing this particular joint take the form:

$$\dot{\mathbf{B}}_i^v = \mathbf{0}_6 \tag{2.177}$$

$$\dot{\mathbf{b}}_i^v = \begin{bmatrix} \mathbf{0} & 2\left(\dot{\tilde{\mathbf{r}}}_G^i - \dot{\tilde{\mathbf{r}}}_i\right)\mathbf{E} + 2\left(\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_i\right)\dot{\mathbf{E}} \\ \mathbf{0} & 2\dot{\mathbf{E}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & 2\left(\dot{\tilde{\mathbf{r}}}_G^i - \dot{\tilde{\mathbf{r}}}_i\right)\mathbf{E} + 2\left(\mathbf{r}_G^i - \tilde{\mathbf{r}}_i\right)\dot{\bar{\mathbf{E}}} \\ \mathbf{0} & 2\dot{\bar{\mathbf{E}}} \end{bmatrix} \tag{2.178}$$

where the identities $\mathbf{A}_0 = \mathbf{I}_3$, $\mathbf{E} = \mathbf{A}_0\bar{\mathbf{E}} = \bar{\mathbf{E}}$ and $\dot{\mathbf{E}} = \dot{\bar{\mathbf{E}}}$ have been used.

Note that, the expression of $\mathbf{d}_i^v$ obtained from (2.176b) is equivalent to (2.121e) according to $\dot{\bar{\mathbf{E}}}\dot{\bar{\mathbf{p}}} = \mathbf{0}$

The particularization for $\mathbf{r}_i = \mathbf{r}_G^i$ (RTdyn0), leads to the following straightforward expressions:

$$\mathbf{b}_i^y = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0} \\ \mathbf{0} & 2\mathbf{E} \end{bmatrix} \tag{2.179a}$$

$$\mathbf{d}_i^y = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \tag{2.179b}$$

$$\dot{\mathbf{b}}_i^y = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & 2\dot{\mathbf{E}} \end{bmatrix} \tag{2.179c}$$

For the choice $\mathbf{r}_i = \mathbf{0}$ (RTdyn1),

$$\mathbf{b}_i^z = \begin{bmatrix} \mathbf{I}_3 & 2\tilde{\mathbf{r}}_G^i\mathbf{E} \\ \mathbf{0} & 2\mathbf{E} \end{bmatrix} \tag{2.180a}$$

$$\dot{\mathbf{b}}_i^z = \begin{bmatrix} \mathbf{0} & 2\left(\dot{\tilde{\mathbf{r}}}_G^i - \dot{\tilde{\mathbf{r}}}_0^i\right)\mathbf{E} + 2\tilde{\mathbf{r}}_G^i\dot{\mathbf{E}} \\ \mathbf{0} & 2\dot{\mathbf{E}} \end{bmatrix} \tag{2.180b}$$

$$\mathbf{d}_i^z = \begin{bmatrix} -2\mathbf{E}\dot{\bar{\mathbf{p}}}_i \wedge \left(2\mathbf{E}\dot{\bar{\mathbf{p}}}_i \wedge \mathbf{r}_G^i\right) \\ \mathbf{0} \end{bmatrix} \tag{2.180c}$$

From a computational perspective, it can be inferred from (2.179) and (2.180) that the floating joint is better suited for the RTdyn0 approach.

### 2.2.7 Planar joint recursive equations

Planar joints can be regarded as a concatenation of two elemental prismatic joints and an elemental revolute joint, as introduced in section 2.1.7. Since prismatic joints prevent rotation, the relation between angular velocities of the bodies connected by this joint is described by the relative angular velocity of the elemental revolute joint, as presented in (2.94).

Analogously, the relative velocity of a point $i$ of body $i$ with respect to body $i-1$ can be assessed using the relative expressions of each one of the commented elemental joints. Therefore, this relative velocity takes the form:

$$\dot{\mathbf{r}}_i^{i,i-1} = \dot{\mathbf{r}}_j^{i,i-1} + \boldsymbol{\omega}_i^{i-1} \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) = \dot{z}_{k1}\mathbf{u}_j + \dot{z}_{k2}\mathbf{v}_j + \dot{z}_{k3}\mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) \tag{2.181}$$

An equivalent procedure can be applied to linear and angular accelerations. Using (2.94), the relative angular acceleration yields:

$$\boldsymbol{\alpha}_i^{i-1} = \mathbf{w}_j\ddot{z}_{k3} \tag{2.182}$$

Applying the same procedure used for relative velocities, the relative linear acceleration of a point $i$ belonging to a body $i$ with respect to the previous one $i-1$ becomes:

$$\ddot{\mathbf{r}}_i^{i,i-1} = \ddot{z}_{k1}\mathbf{u}_j + \ddot{z}_{k2}\mathbf{v}_j + \ddot{z}_{k3}\mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) + \left(\dot{z}_{k3}\mathbf{w}_j\right) \wedge \left[\left(\dot{z}_{k3}\mathbf{w}_j\right) \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right)\right] \tag{2.183}$$

Considering that $\mathbf{r}_j = \mathbf{r}_G^i$, and gathering linear and angular velocity equations (2.113), (2.113),(2.181) and (2.94) and acceleration expressions (2.114), (2.183) and (2.182) the following relations can be obtained:

$$\begin{bmatrix}\dot{\mathbf{r}}_i\\ \boldsymbol{\omega}_i\end{bmatrix} = \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_{i-1}-\tilde{\mathbf{r}}_i\\ \mathbf{0} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\dot{\mathbf{r}}_{i-1}\\ \boldsymbol{\omega}_{i-1}\end{bmatrix} + \begin{bmatrix}\mathbf{u}_j & \mathbf{v}_j & \mathbf{w}_j \wedge (\mathbf{r}_i - \mathbf{r}_G^i)\\ \mathbf{0} & \mathbf{0} & \mathbf{w}_j\end{bmatrix}\dot{\mathbf{z}}_i \tag{2.184}$$

$$\begin{aligned}\begin{bmatrix}\ddot{\mathbf{r}}_i\\ \dot{\boldsymbol{\omega}}_i\end{bmatrix} &= \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_{i-1}-\tilde{\mathbf{r}}_i\\ \mathbf{0} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\ddot{\mathbf{r}}_{i-1}\\ \dot{\boldsymbol{\omega}}_{i-1}\end{bmatrix} + \begin{bmatrix}\mathbf{u}_j & \mathbf{v}_j & \mathbf{w}_j \wedge (\mathbf{r}_i - \mathbf{r}_G^i)\\ \mathbf{0} & \mathbf{0} & \mathbf{w}_j\end{bmatrix}\ddot{\mathbf{z}}_i\\ &+ \begin{bmatrix}(\dot{z}_{k3}\mathbf{w}_j + 2\boldsymbol{\omega}_{i-1}) \wedge \left[\dot{z}_{k3}\mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right)\right] + 2\boldsymbol{\omega}_{i-1} \wedge (\dot{z}_{k1}\mathbf{u}_j + \dot{z}_{k2}\mathbf{v}_j)\\ \boldsymbol{\omega}_{i-1} \wedge \dot{z}_{k3}\mathbf{w}_j\end{bmatrix}\\ &+ \begin{bmatrix}\boldsymbol{\omega}_{i-1} \wedge \left[\boldsymbol{\omega}_{i-1} \wedge \left(\mathbf{r}_i - \mathbf{r}_{i-1}\right)\right]\\ \mathbf{0}\end{bmatrix}\end{aligned} \tag{2.185}$$

where $\dot{\mathbf{z}}_i = \begin{bmatrix}\dot{z}_{k1} & \dot{z}_{k2} & \dot{z}_{k3}\end{bmatrix}^{\mathrm{T}}$, $\ddot{\mathbf{z}}_i = \begin{bmatrix}\ddot{z}_{k1} & \ddot{z}_{k2} & \ddot{z}_{k3}\end{bmatrix}^{\mathrm{T}}$.

Comparing (2.184) and (2.185) with the general recursive expressions (2.121a) and (2.121b), the following joint-dependent recursive relations can be identified:

$$\mathbf{b}_i^v = \begin{bmatrix}\mathbf{u}_j & \mathbf{v}_j & \mathbf{w}_j \wedge (\mathbf{r}_i - \mathbf{r}_G^i)\\ \mathbf{0} & \mathbf{0} & \mathbf{w}_j\end{bmatrix} \tag{2.186a}$$

$$\dot{\mathbf{b}}_i^v = \begin{bmatrix}\dot{\mathbf{u}}_j & \dot{\mathbf{v}}_j & \dot{\mathbf{w}}_j \wedge (\mathbf{r}_i - \mathbf{r}_G^i) + \mathbf{w}_j \wedge (\dot{\mathbf{r}}_i - \dot{\mathbf{r}}_G^i)\\ \mathbf{0} & \mathbf{0} & \dot{\mathbf{w}}_j\end{bmatrix} \tag{2.186b}$$

Particularizing (2.160a) and (2.186b) for RTdyn0, with the set of reference points coincident with the CoM of each body $\mathbf{r}_i = \mathbf{r}_G^i$, $\mathbf{r}_{i-1} = \mathbf{r}_G^{i-1}$:

$$\mathbf{b}_i^y = \begin{bmatrix}\mathbf{u}_j & \mathbf{v}_j & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{w}_j\end{bmatrix} \tag{2.187a}$$

$$\dot{\mathbf{b}}_i^y = \begin{bmatrix}\dot{\mathbf{u}}_j & \dot{\mathbf{v}}_j & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \dot{\mathbf{w}}_j\end{bmatrix} \tag{2.187b}$$

The RTdyn1 version, selecting the points of the bodies coincident with the origin of coordinates at each instant of time as reference points, $\mathbf{r}_i = \mathbf{r}_0^i$, $\mathbf{r}_{i-1} = \mathbf{r}_0^{i-1}$, delivers:

$$\mathbf{b}_i^z = \begin{bmatrix}\mathbf{u}_j & \mathbf{v}_j & -\mathbf{w}_j \wedge \mathbf{r}_G^i\\ \mathbf{0} & \mathbf{0} & \mathbf{w}_j\end{bmatrix} \tag{2.188a}$$

$$\dot{\mathbf{b}}_i^z = \begin{bmatrix}\dot{\mathbf{u}}_j & \dot{\mathbf{v}}_j & -\dot{\mathbf{w}}_j \wedge \mathbf{r}_G^i - \mathbf{w}_j \wedge \dot{\mathbf{r}}_G^i\\ \mathbf{0} & \mathbf{0} & \dot{\mathbf{w}}_j\end{bmatrix} \tag{2.188b}$$

A special simplified case of this joint emerges in a planar mechanisms with none of its bodies linked to the ground by any kinematic joint. In this sense, a planar joint is added to the mechanism assuming the role of the floating joint. It should be noted that the floating joint could work as the planar joint for this type of mechanisms, but the reduced number of joint coordinates and the absence of joint constraints (as the

Euler parameter normalization constraint required by floating joints) make this type of joint more convenient. The modeling of a planar mechanism is usually tackled, for the sake of simplicity, in the normal plane of one of the vectors composing the global reference frame. If the plane in which the planar mechanism is modeled is normal to the vector $z$, for example, the expression of $\mathbf{b}_i^v$ would be:

$$\mathbf{b}_i^v = \begin{bmatrix} \begin{bmatrix}1 & 0 & 0\end{bmatrix}^{\mathrm{T}} & \begin{bmatrix}0 & 1 & 0\end{bmatrix}^{\mathrm{T}} & \begin{bmatrix}0 & 0 & 1\end{bmatrix}^{\mathrm{T}} \wedge (\mathbf{r}_i - \mathbf{r}_G^i) \\ \mathbf{0} & \mathbf{0} & \begin{bmatrix}0 & 0 & 1\end{bmatrix}^{\mathrm{T}} \end{bmatrix} \tag{2.189}$$

Note that the resulting expressions for this simplified case allow both a simpler model generation and fewer calculations. The fact that all the vectors defining the joint are constant conveys a significant time saving during computation too.

## 2.3 Kinematic analysis of minimal relative coordinate models

The kinematics of any MBS can be fully described by the set of joint coordinates associated to its relative coordinate model. In the case of open-loop systems modeled with minimal coordinates, joint coordinates compose a set of DoF of the system. However, paying attention to the implementation, a different definition of the DoF can be required by the user, including coordinates of points and vectors[1]. Hence, the kinematic and recursive relations presented in previous sections are insufficient for completely determining the kinematics of any body for a given set of DoF values, and a two-stage iterative problem has to be addressed.

In the following subsections, the initial position problem, the kinematic velocity problem and the kinematic acceleration analysis are addressed for relative coordinate models with minimal coordinates. The finite displacements problem is not developed here, since it can be solved with the initial position problem equations.

### 2.3.1 Initial position problem

The process begins with an initial approximate solution of the joint coordinates, which could be supplied by the user or computed by means of the initial approximate position of the points and vectors of each body. With these values, the approximate value of all the points and vectors of the model can be computed recursively, from the base of the mechanism till the tips.

The second step involves the imposition of the DoF. If the set of joint coordinates does not include the degrees of freedom specified by the user, the following equation has to be solved:

$$\mathbf{B}^{\{n\}} \Delta \mathbf{z}^{\{n+1\}} = \mathbf{d} - \left(\mathbf{z}^i\right)^{\{n\}} \tag{2.190}$$

[1]Since the definition of mechanisms in MBSLIM is given in natural coordinates, it is quite typical to use some of these coordinates. For full integration of the topological formulations in MBSLIM supporting DoF not belonging to the joint coordinates vector is mandatory.

in which $\Delta\mathbf{z}^{\{n+1\}}$ is the increment in the joint coordinates, $\mathbf{d}$ is the set of desired values of the degrees of freedom in positions[2], $(\mathbf{z}^i)^{\{n\}}$ is the current value of the degrees of freedom at iteration $n$ and $\mathbf{B}^{\{n\}}$ is a matrix describing the relation between the positions of the degrees of freedom and the joint coordinates[3], whose expression is:

$$\mathbf{B} = \frac{\partial \mathbf{z}^i}{\partial \mathbf{z}} \tag{2.191}$$

being $\mathbf{z}$ the set of joint coordinates and $\mathbf{z}^i$ the array of DoF. This matrix $\mathbf{B}$ will be revisited in future chapters, in which its relevance will be evidenced. It can be proved that, for a proper selection of degrees of freedom, the matrix $\mathbf{B}$ has a left inverse, thus (2.190) has a solution.

After solving each iteration of (2.190), the values of $\mathbf{z}^i$ have to be computed recursively from the new set of joint coordinates. This process have to be repeated until convergence, this is, until the norm of $\Delta\mathbf{z}^{\{n+1\}}$ is lower than a given tolerance.

Behold that $\mathbf{d}$ and $\mathbf{z}^i$ are equal as soon as (2.190) converges. In this sense, $\mathbf{z}^i$ and $\mathbf{d}$ can be regarded as interchangeable values after the problem is solved, but it should be bore in mind that they have different meanings.

## 2.3.2 Velocity problem

Minimal relative coordinate models have all their velocities fully defined by the set of velocities of the joint coordinates, but, similarly to what happens in positions, if the selection of DoF does not match the joint coordinates, a new step has to be added to the velocity problem.

First, the velocities of the joint coordinates have to be determined satisfying the following equation:

$$\mathbf{B}\dot{\mathbf{z}} = \dot{\mathbf{d}} \tag{2.192}$$

in which $\mathbf{B}$ is a matrix relating the variation of DoF with respect to joint coordinates as proposed in (2.191), $\dot{\mathbf{z}}$ are the velocities of the minimal joint coordinates and $\mathbf{d}$ the desired velocity values of the DoF. It should be remarked that (2.192) is only valid for a proper selection of DoF, for which matrix $\mathbf{B}$ has a left inverse.

Behold that in this problem, the velocity values of the DoF, $\dot{\mathbf{z}}^i$, before the solution of (2.192) do not play any role because the relation between joint-coordinate velocities and the desired DoF velocities $\dot{\mathbf{d}}$, is linear.

Once (2.192) is computed, the recursive relations can be applied to the relative coordinate model in order to obtain the kinematics of each body. Observe that it is not an iterative process as the position problem.

## 2.3.3 Acceleration problem

The kinematic acceleration analysis of minimal coordinate problems suffers from the issues commented in the previous sections for position and velocity problems when

[2]Do not confuse with $\mathbf{d}_i^v$, an elemental term of the recursive accumulation.

[3]Do not confuse with $\mathbf{B}_i^v$, an elemental term of the recursive accumulation.

the set of DoF does not coincide with the joint coordinates array. The problem can be solved for a proper selection of DoF with an additional computation:

$$\mathbf{B}\ddot{\mathbf{z}} = \ddot{\mathbf{d}} - \dot{\mathbf{B}}\dot{\mathbf{z}} \tag{2.193}$$

Herein, $\mathbf{B}$ and $\dot{\mathbf{B}}$ are the matrix defined in (2.191) and its time derivative, respectively, $\ddot{\mathbf{z}}$ the accelerations of the joint coordinates, $\ddot{\mathbf{d}}$ the accelerations of the DoF and $\dot{\mathbf{z}}$ the velocities of the joint coordinates.

Analogously to the velocity problem of section 2.3.2, the acceleration values of the DoF, $\ddot{\mathbf{z}}^i$, prior to the solution of (2.193) are not involved in the kinematic acceleration problem since it is linear on joint coordinate accelerations.

Point and vector accelerations along with angular accelerations can be computed recursively from the base of the mechanism to the tips using the recursive relations presented in section 2.2.

## 2.4 Equations of motion for unconstrained open-loop systems

In this section, the general equations of motion for any reference point are derived using two different procedures: the semi-recursive method and the fully-recursive method. Different algorithms of solution for the sets of equations generated are proposed, two for the semi-recursive approach and one for the fully-recursive approach. Finally, the equations of motion are particularized for two sets of reference points, leading to the RTdyn0 and RTdyn1 formulations presented in [45].

### 2.4.1 Semi-recursive method

Semi-recursive methods involve both recursive calculations based on the topology of the mechanism as well as non-recursive or global procedures. The semi-recursive method here presented, and thoroughly described in [45], uses a recursive method to compute the kinematics of the open-loop system, while the rest of the stages for the generation of the EoM are solved by means of global methods. The recursive evaluation of the kinematics has a direct impact on the generation of the mass matrix of the system and the generalized forces vector, as it will be patent in the following developments.

Let us begin by applying the virtual power principle to a multibody system composed of $n_b$ bodies,

$$\sum_{i=1}^{n_b} \mathbf{Y}_i^{*\mathrm{T}} \left[\mathbf{M}_i \dot{\mathbf{Y}}_i - \mathbf{Q}_i\right] = 0 \tag{2.194a}$$

$$\sum_{i=1}^{n_b} \begin{bmatrix} \dot{\mathbf{r}}_G^{i*} \\ \boldsymbol{\omega}_i^* \end{bmatrix} \left( \begin{bmatrix} m_i \mathbf{I}_3 & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_i^G \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{r}}_G^i \\ \dot{\boldsymbol{\omega}}_i \end{bmatrix} - \begin{bmatrix} \mathbf{f}_i \\ \mathbf{n}_i^G - \boldsymbol{\omega}_i \wedge \mathbf{J}_i^G \boldsymbol{\omega}_i \end{bmatrix} \right) = 0 \tag{2.194b}$$

where the star indicates virtual velocities, $\mathbf{J}_i^G$ is the inertia tensor in the CoM, $\mathbf{f}_i$ is the external forces vector over the body and $\mathbf{n}_i^G$ is the external torque vector with respect to the CoM. Note that if $\mathbf{f}_i$ is applied in a different point than the CoM, it will generate a torque equal to:

$$\mathbf{n}_i^G = \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_G^i\right)\mathbf{f}_i \tag{2.195}$$

Returning to (2.194), observe that all the magnitudes are expressed in the global reference frame. The inertia tensor referred to global axes can be expressed in terms of the local inertia tensor (which is constant), as:

$$\mathbf{J}_i^G = \mathbf{A}_i\bar{\mathbf{J}}_i^G\mathbf{A}_i^{\mathrm{T}} \tag{2.196}$$

The general recursive relations are not written in terms of $\mathbf{Y}_i$ but of $\mathbf{V}_i = \begin{bmatrix}\dot{\mathbf{r}}_i^{\mathrm{T}} & \boldsymbol{\omega}_i^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}}$. The following kinematic relations hold, between the reference point in body $i$, $\mathbf{r}_i = \mathbf{r}_i^i$ and the CoM, $\mathbf{r}_G^i$,

$$\mathbf{Y}_i^* = \begin{bmatrix}\dot{\mathbf{r}}_G^{i*}\\ \boldsymbol{\omega}_i^*\end{bmatrix} = \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_G^i\\ \mathbf{0} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\dot{\mathbf{r}}_i^*\\ \boldsymbol{\omega}_i^*\end{bmatrix} = \mathbf{D}_i^v\mathbf{V}_i^* \tag{2.197}$$

$$\dot{\mathbf{Y}}_i = \begin{bmatrix}\ddot{\mathbf{r}}_G^i\\ \dot{\boldsymbol{\omega}}_i\end{bmatrix} = \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_G^i\\ \mathbf{0} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\ddot{\mathbf{r}}_i\\ \dot{\boldsymbol{\omega}}_i\end{bmatrix} + \begin{bmatrix}\boldsymbol{\omega}_i \wedge \left(\boldsymbol{\omega}_i \wedge \left(\mathbf{r}_G^i - \mathbf{r}_i\right)\right)\\ \mathbf{0}\end{bmatrix} = \mathbf{D}_i^v\dot{\mathbf{V}}_i + \mathbf{e}_i^v \tag{2.198}$$

Applying the previous relations to the virtual power equations (2.194a),

$$\sum_{i=1}^{n_b}\mathbf{V}_i^{*\mathrm{T}}\left[\mathbf{M}_i^v\dot{\mathbf{V}}_i - \mathbf{Q}_i^v\right] = 0 \tag{2.199}$$

$$\begin{gathered}\mathbf{M}_i^v = \left(\mathbf{D}_i^v\right)^{\mathrm{T}}\mathbf{M}_i\mathbf{D}_i^v = \begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_i & \mathbf{I}\end{bmatrix}\begin{bmatrix}m_i\mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{J}_i^G\end{bmatrix}\begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_G^i\\ \mathbf{0} & \mathbf{I}\end{bmatrix} = \\ \begin{bmatrix}m_i\mathbf{I} & -m_i\left(\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_i\right)\\ m_i\left(\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_i\right) & \mathbf{J}_i^G - m_i\left(\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_i\right)\left(\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_i\right)\end{bmatrix}\end{gathered} \tag{2.200}$$

$$\begin{gathered}\mathbf{Q}_i^v = \left(\mathbf{D}_i^v\right)^{\mathrm{T}}\left(\mathbf{Q}_i - \mathbf{M}_i\mathbf{e}_i^v\right) = \begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_i & \mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{f}_i - m_i\boldsymbol{\omega}_i \wedge \left(\boldsymbol{\omega}_i \wedge \left(\mathbf{r}_G^i - \mathbf{r}_i\right)\right)\\ \mathbf{n}_i^G - \boldsymbol{\omega}_i \wedge \mathbf{J}_i^G\boldsymbol{\omega}_i\end{bmatrix} = \\ \begin{bmatrix}\mathbf{f}_i - m_i\boldsymbol{\omega}_i \wedge \left(\boldsymbol{\omega}_i \wedge \left(\mathbf{r}_G^i - \mathbf{r}_i\right)\right)\\ \mathbf{n}_i^G - \boldsymbol{\omega}_i \wedge \mathbf{J}_i^G\boldsymbol{\omega}_i + \left(\mathbf{r}_G^i - \mathbf{r}_i\right) \wedge \left[\mathbf{f}_i - m_i\boldsymbol{\omega}_i \wedge \left(\boldsymbol{\omega}_i \wedge \left(\mathbf{r}_G^i - \mathbf{r}_i\right)\right)\right]\end{bmatrix}\end{gathered} \tag{2.201}$$

Defining,

$$\mathbf{M}^v = \begin{bmatrix}\mathbf{M}_1^v & \mathbf{0} & \dots & \mathbf{0}\\ \mathbf{0} & \mathbf{M}_2^v & \dots & \mathbf{0}\\ \vdots & \vdots & \ddots & \vdots\\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{M}_{n_b}^v\end{bmatrix};\quad \mathbf{Q}^v = \begin{bmatrix}\mathbf{Q}_1^v\\ \mathbf{Q}_2^v\\ \vdots\\ \mathbf{Q}_{n_b}^v\end{bmatrix};\quad \mathbf{V} = \begin{bmatrix}\mathbf{V}_1\\ \mathbf{V}_2\\ \vdots\\ \mathbf{V}_{n_b}\end{bmatrix} \tag{2.202}$$

the dynamics equations can be grouped in matrix form:

$$\mathbf{V}^{*\mathrm{T}}\left[\mathbf{M}^v\dot{\mathbf{V}} - \mathbf{Q}^v\right] = 0 \tag{2.203}$$

It will be proved in the following lines the subsequent linear relations of virtual velocities and real velocities and accelerations,

$$\mathbf{V} = \mathbf{R}^v \dot{\mathbf{z}} \tag{2.204}$$

$$\mathbf{V}^* = \mathbf{R}^v \dot{\mathbf{z}}^* \tag{2.205}$$

$$\dot{\mathbf{V}} = \mathbf{R}^v \ddot{\mathbf{z}} + \dot{\mathbf{R}}^v \dot{\mathbf{z}} \tag{2.206}$$

where $\mathbf{z} = \begin{bmatrix} \mathbf{z}_1{}^{\mathrm{T}} & \mathbf{z}_2{}^{\mathrm{T}} & \dots & \mathbf{z}_{n_b}{}^{\mathrm{T}} \end{bmatrix} \in \mathbb{R}^n$ with $n$ the total number of joint coordinates.

Then equation (2.203) becomes:

$$\left(\dot{\mathbf{z}}^{*\mathrm{T}} \mathbf{R}^{v\mathrm{T}}\right) \left[\mathbf{M}^v \left(\mathbf{R}^v \ddot{\mathbf{z}} + \dot{\mathbf{R}}^v \dot{\mathbf{z}}\right) - \mathbf{Q}^v\right] = \dot{\mathbf{z}}^{*\mathrm{T}} \left[\mathbf{R}^{v\mathrm{T}} \mathbf{M}^v \left(\mathbf{R}^v \ddot{\mathbf{z}} + \dot{\mathbf{R}}^v \dot{\mathbf{z}}\right) - \mathbf{R}^{v\mathrm{T}} \mathbf{Q}^v\right] = \\ \dot{\mathbf{z}}^{*\mathrm{T}} \left[\left(\mathbf{R}^{v\mathrm{T}} \mathbf{M}^v \mathbf{R}^v\right) \ddot{\mathbf{z}} - \mathbf{R}^{v\mathrm{T}} \left(\mathbf{Q}^v - \mathbf{M}^v \dot{\mathbf{R}}^v \dot{\mathbf{z}}\right)\right] = 0 \tag{2.207}$$

Or, more compactly:

$$\dot{\mathbf{z}}^{*\mathrm{T}} \left[\mathbf{M}^d \ddot{\mathbf{z}} - \mathbf{Q}^d\right] = 0 \tag{2.208a}$$

$$\mathbf{M}^d = \mathbf{R}^{v\mathrm{T}} \mathbf{M}^v \mathbf{R}^v \tag{2.208b}$$

$$\mathbf{Q}^d = \mathbf{R}^{v\mathrm{T}} \left(\mathbf{Q}^v - \mathbf{M}^v \dot{\mathbf{R}}^v \dot{\mathbf{z}}\right) \tag{2.208c}$$

Since we are dealing with open-loop systems, joint coordinates can be considered DoF of the system[4] and their virtual velocities chosen arbitrarily, to arrive at the following ODE system of equations of motion:

$$\mathbf{M}^d \ddot{\mathbf{z}} = \mathbf{Q}^d \tag{2.209}$$

The EoM (2.209) involve the calculation of matrix $\mathbf{R}^v$ and vector $\dot{\mathbf{R}}^v \dot{\mathbf{z}}$. Their assessment will be explained for the open-loop system of Figure 2.10, but the process is general for any multibody system.

Applying (2.113) recursively to the six body mechanism and comparing with (2.204) gives us,

$$\mathbf{V} = \begin{pmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \\ \mathbf{V}_3 \\ \mathbf{V}_4 \\ \mathbf{V}_5 \\ \mathbf{V}_6 \end{pmatrix} = \begin{bmatrix} \mathbf{b}_1^v & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{B}_2^v \mathbf{b}_1^v & \mathbf{b}_2^v & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{B}_3^v \mathbf{B}_2^v \mathbf{b}_1^v & \mathbf{B}_3^v \mathbf{b}_2^v & \mathbf{b}_3^v & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{B}_4^v \mathbf{b}_1^v & \mathbf{0} & \mathbf{0} & \mathbf{b}_4^v & \mathbf{0} & \mathbf{0} \\ \mathbf{B}_5^v \mathbf{B}_4^v \mathbf{b}_1^v & \mathbf{0} & \mathbf{0} & \mathbf{B}_5^v \mathbf{b}_4^v & \mathbf{b}_5^v & \mathbf{0} \\ \mathbf{B}_6^v \mathbf{B}_4^v \mathbf{b}_1^v & \mathbf{0} & \mathbf{0} & \mathbf{B}_6^v \mathbf{b}_4^v & \mathbf{0} & \mathbf{b}_6^v \end{bmatrix} \begin{pmatrix} \dot{\mathbf{z}}_1 \\ \dot{\mathbf{z}}_2 \\ \dot{\mathbf{z}}_3 \\ \dot{\mathbf{z}}_4 \\ \dot{\mathbf{z}}_5 \\ \dot{\mathbf{z}}_6 \end{pmatrix} = \mathbf{R}^v \dot{\mathbf{z}} \tag{2.210}$$

[4]A special case of non-minimal joint coordinates is the spherical joint modeled with Euler parameters, where only three out of the four parameters are DoF. In order to use the formulation for open-loop systems with these joints, three parameters must be chosen as DoF and the other one expressed in terms of them. Otherwise the formulation for closed-loop systems should be used, adding the corresponding normalization constraint for the Euler parameters of each spherical (or floating) joint.

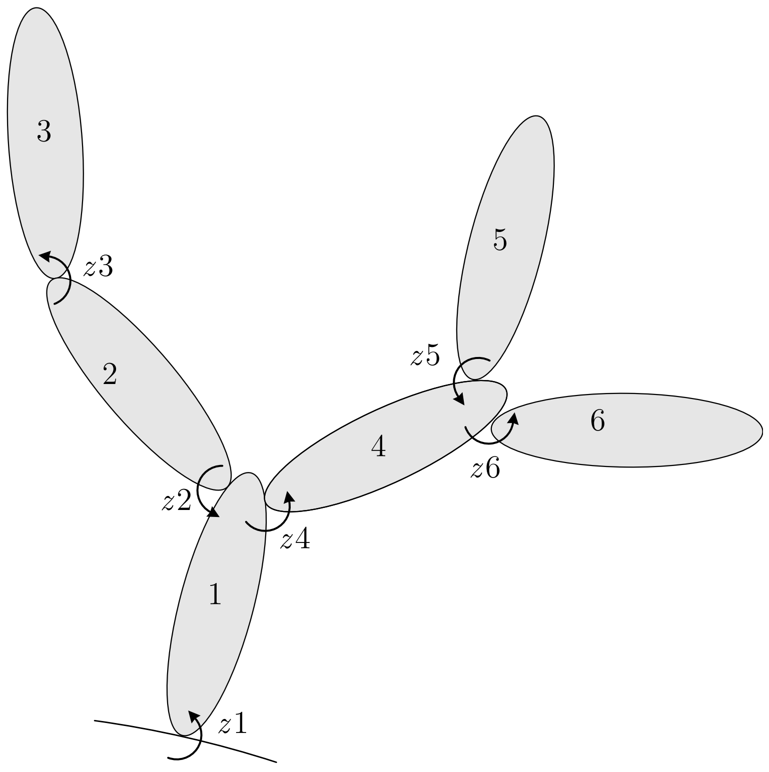


Figure 2.10: Six body open chain mechanism

Thus it is proved expression (2.204) and terms in $\mathbf{R}^v$ can be expressed as,

$$\mathbf{R}^v = \begin{bmatrix} \mathbf{b}^v_{1,1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{b}^v_{2,1} & \mathbf{b}^v_{2,2} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{b}^v_{3,1} & \mathbf{b}^v_{3,2} & \mathbf{b}^v_{3,3} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{b}^v_{4,1} & \mathbf{0} & \mathbf{0} & \mathbf{b}^v_{4,4} & \mathbf{0} & \mathbf{0} \\ \mathbf{b}^v_{5,1} & \mathbf{0} & \mathbf{0} & \mathbf{b}^v_{5,4} & \mathbf{b}^v_{5,5} & \mathbf{0} \\ \mathbf{b}^v_{6,1} & \mathbf{0} & \mathbf{0} & \mathbf{b}^v_{6,4} & \mathbf{0} & \mathbf{b}^v_{6,6} \end{bmatrix} = \begin{bmatrix} \mathbf{b}^v_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{B}^v_2\mathbf{b}^v_1 & \mathbf{b}^v_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{B}^v_3\mathbf{B}^v_2\mathbf{b}^v_1 & \mathbf{B}^v_3\mathbf{b}^v_2 & \mathbf{b}^v_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{B}^v_4\mathbf{b}^v_1 & \mathbf{0} & \mathbf{0} & \mathbf{b}^v_4 & \mathbf{0} & \mathbf{0} \\ \mathbf{B}^v_5\mathbf{B}^v_4\mathbf{b}^v_1 & \mathbf{0} & \mathbf{0} & \mathbf{B}^v_5\mathbf{b}^v_4 & \mathbf{b}^v_5 & \mathbf{0} \\ \mathbf{B}^v_6\mathbf{B}^v_4\mathbf{b}^v_1 & \mathbf{0} & \mathbf{0} & \mathbf{B}^v_6\mathbf{b}^v_4 & \mathbf{0} & \mathbf{b}^v_6 \end{bmatrix} \tag{2.211}$$

Observe that terms $\mathbf{b}^v_{i,j}$ can be obtained recursively by means of the following relations,

$$\mathbf{b}^v_{i,i} = \mathbf{b}^v_i \tag{2.212a}$$

$$\mathbf{b}^v_{i,j} = \mathbf{B}^v_i\mathbf{b}^v_{h,j};\ i > j \tag{2.212b}$$

$$\mathbf{b}^v_{i,j} = \mathbf{0};\ i < j \tag{2.212c}$$

where $h$ is the parent body of $i$, i.e. the preceding body in the kinematic chain. Therefore these relations are calculated recursively, traversing the rows of the matrix, from the root to the leaves of the mechanism. Note that the previous relations imply that $\mathbf{b}^v_{i,j} = \mathbf{0}$ if $j$ is not an ancestor of $i$.

Matrix $\mathbf{R}^v$ can be divided in the following matrices with scarce interest for this formulation but useful for the fully-recursive formulation derived later,

$$\mathbf{R}^v = \begin{bmatrix} \mathbf{I}_6 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{B}_2^v & \mathbf{I}_6 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{B}_3^v\mathbf{B}_2^v & \mathbf{B}_3^v & \mathbf{I}_6 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{B}_4^v & \mathbf{0} & \mathbf{0} & \mathbf{I}_6 & \mathbf{0} & \mathbf{0} \\ \mathbf{B}_5^v\mathbf{B}_4^v & \mathbf{0} & \mathbf{0} & \mathbf{B}_5^v & \mathbf{I}_6 & \mathbf{0} \\ \mathbf{B}_6^v\mathbf{B}_4^v & \mathbf{0} & \mathbf{0} & \mathbf{B}_6^v & \mathbf{0} & \mathbf{I}_6 \end{bmatrix} \begin{bmatrix} \mathbf{b}_1^v & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{b}_2^v & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{b}_3^v & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{b}_4^v & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{b}_5^v & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{b}_6^v \end{bmatrix} = \mathbf{T}^v\mathbf{R}_d^v \tag{2.213}$$

The vector $\dot{\mathbf{R}}^v\dot{\mathbf{z}}$ can be analogously obtained from the recursive application of (2.114) to the six body mechanism and through its comparison with (2.206):

$$\dot{\mathbf{R}}^v\dot{\mathbf{z}} = \begin{bmatrix} \mathbf{d}_1^v \\ \mathbf{d}_2^{v\Sigma} \\ \mathbf{d}_3^{v\Sigma} \\ \mathbf{d}_4^{v\Sigma} \\ \mathbf{d}_5^{v\Sigma} \\ \mathbf{d}_6^{v\Sigma} \end{bmatrix} = \begin{bmatrix} \mathbf{d}_1^v \\ \mathbf{d}_2^v + \mathbf{B}_2^v\mathbf{d}_1^v \\ \mathbf{d}_3^v + \mathbf{B}_3^v\left(\mathbf{d}_2^v + \mathbf{B}_2^v\mathbf{d}_1^v\right) \\ \mathbf{d}_4^v + \mathbf{B}_4^v\mathbf{d}_1^v \\ \mathbf{d}_5^v + \mathbf{B}_5^v\left(\mathbf{d}_4^v + \mathbf{B}_4^v\mathbf{d}_1^v\right) \\ \mathbf{d}_6^v + \mathbf{B}_6^v\left(\mathbf{d}_4^v + \mathbf{B}_4^v\mathbf{d}_1^v\right) \end{bmatrix} = \begin{bmatrix} \mathbf{d}_1^v \\ \mathbf{d}_2^v + \mathbf{B}_2^v\mathbf{d}_1^v \\ \mathbf{d}_3^v + \mathbf{B}_3^v\mathbf{d}_2^{v\Sigma} \\ \mathbf{d}_4^v + \mathbf{B}_4^v\mathbf{d}_1^v \\ \mathbf{d}_5^v + \mathbf{B}_5^v\mathbf{d}_4^{v\Sigma} \\ \mathbf{d}_6^v + \mathbf{B}_6^v\mathbf{d}_4^{v\Sigma} \end{bmatrix} \tag{2.214}$$

Observe that each body receives recursively the contributions with the $\mathbf{d}_i^v$ terms from the parent body in the kinematic chain by means of the following recursive equation:

$$\mathbf{d}_i^{v\Sigma} = \mathbf{d}_i^v + \mathbf{B}_i^v\mathbf{d}_h^{v\Sigma} \tag{2.215}$$

where $h$ is again the parent body of $i$, this is the preceding body in the kinematic chain and $\mathbf{d}_i^v$ is defined by (2.121e).

With the help of the recursive kinematic expressions defined before, the mass matrix of the system,

$$\mathbf{M}^d = \mathbf{R}^{v\mathrm{T}}\mathbf{M}^v\mathbf{R}^v =$$

$$\begin{bmatrix} \mathbf{b}_1^{v\mathrm{T}}\mathbf{M}_1^{v\Sigma}\mathbf{b}_1^v & \mathbf{b}_{2,1}^{v\mathrm{T}}\mathbf{M}_2^{v\Sigma}\mathbf{b}_2^v & \mathbf{b}_{3,1}^{v\mathrm{T}}\mathbf{M}_3^v\mathbf{b}_3^v & \mathbf{b}_{4,1}^{v\mathrm{T}}\mathbf{M}_4^{v\Sigma}\mathbf{b}_4^v & \mathbf{b}_{5,1}^{v\mathrm{T}}\mathbf{M}_5^v\mathbf{b}_5^v & \mathbf{b}_{6,1}^{v\mathrm{T}}\mathbf{M}_6^v\mathbf{b}_6^v \\ \mathbf{b}_2^{v\mathrm{T}}\mathbf{M}_2^{v\Sigma}\mathbf{b}_{2,1}^v & \mathbf{b}_2^{v\mathrm{T}}\mathbf{M}_2^{v\Sigma}\mathbf{b}_2^v & \mathbf{b}_{3,2}^{v\mathrm{T}}\mathbf{M}_3^v\mathbf{b}_3^v & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{b}_3^{v\mathrm{T}}\mathbf{M}_3^v\mathbf{b}_{3,1}^v & \mathbf{b}_3^{v\mathrm{T}}\mathbf{M}_3^v\mathbf{b}_{3,2}^v & \mathbf{b}_3^{v\mathrm{T}}\mathbf{M}_3^v\mathbf{b}_3^v & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{b}_4^{v\mathrm{T}}\mathbf{M}_4^{v\Sigma}\mathbf{b}_{4,1}^v & \mathbf{0} & \mathbf{0} & \mathbf{b}_4^{v\mathrm{T}}\mathbf{M}_4^{v\Sigma}\mathbf{b}_4^v & \mathbf{b}_{5,4}^{v\mathrm{T}}\mathbf{M}_5^v\mathbf{b}_5^v & \mathbf{b}_{6,4}^{v\mathrm{T}}\mathbf{M}_6^v\mathbf{b}_6^v \\ \mathbf{b}_5^{v\mathrm{T}}\mathbf{M}_5^v\mathbf{b}_{5,1}^v & \mathbf{0} & \mathbf{0} & \mathbf{b}_5^{v\mathrm{T}}\mathbf{M}_5^v\mathbf{b}_{5,4}^v & \mathbf{b}_5^{v\mathrm{T}}\mathbf{M}_5^v\mathbf{b}_5^v & \mathbf{0} \\ \mathbf{b}_6^{v\mathrm{T}}\mathbf{M}_6^v\mathbf{b}_{6,1}^v & \mathbf{0} & \mathbf{0} & \mathbf{b}_6^{v\mathrm{T}}\mathbf{M}_6^v\mathbf{b}_{6,4}^v & \mathbf{0} & \mathbf{b}_6^{v\mathrm{T}}\mathbf{M}_6^v\mathbf{b}_6^v \end{bmatrix} \tag{2.216a}$$

$$\mathbf{M}_4^{v\Sigma} = \mathbf{M}_4^v + \mathbf{B}_5^{v\mathrm{T}}\mathbf{M}_5^v\mathbf{B}_5^v + \mathbf{B}_6^{v\mathrm{T}}\mathbf{M}_6^v\mathbf{B}_6^v \tag{2.216b}$$

$$\mathbf{M}_2^{v\Sigma} = \mathbf{M}_2^v + \mathbf{B}_3^{v\mathrm{T}}\mathbf{M}_3^v\mathbf{B}_3^v \tag{2.216c}$$

$$\mathbf{M}_1^{v\Sigma} = \mathbf{M}_1^v + \mathbf{B}_2^{v\mathrm{T}}\mathbf{M}_2^{v\Sigma}\mathbf{B}_2^v + \mathbf{B}_4^{v\mathrm{T}}\mathbf{M}_4^{v\Sigma}\mathbf{B}_4^v \tag{2.216d}$$

The mass matrix can be expressed in compact form as follows:

$$\mathbf{M}^d(i,j) = \mathbf{R}^{v\mathrm{T}}\mathbf{M}^v\mathbf{R}^v(i,j) = \mathbf{b}_i^{v\mathrm{T}}\mathbf{M}_i^{v\Sigma}\mathbf{b}_{i,j}^v; \qquad i > j, \tag{2.217a}$$

$$\mathbf{M}^d(i,i) = \mathbf{R}^{v\mathrm{T}}\mathbf{M}^v\mathbf{R}^v(i,i) = \mathbf{b}_i^{v\mathrm{T}}\mathbf{M}_i^{v\Sigma}\mathbf{b}_i^v, \tag{2.217b}$$

$$\mathbf{M}^d(i,j) = \mathbf{R}^{v\mathrm{T}}\mathbf{M}^v\mathbf{R}^v(i,j) = \mathbf{b}_{j,i}^{v\mathrm{T}}\mathbf{M}_j^{v\Sigma}\mathbf{b}_j^v; \qquad i < j, \tag{2.217c}$$

$$\mathbf{M}_i^{v\Sigma} = \mathbf{M}_i^v + \sum_{s=1}^{n_s^i}\mathbf{B}_s^{v\mathrm{T}}\mathbf{M}_s^{v\Sigma}\mathbf{B}_s^v \tag{2.217d}$$

with $n_s^i$ the number of children of body $i$. Observe that each body receives a recursive contribution of accumulated masses from its children, therefore equation (2.217d) implies an accumulation of masses from the leaves to the root of the branches.

Similarly, the vector of generalized forces becomes:

$$\mathbf{R}^{v\mathrm{T}}\left(\mathbf{Q}^v - \mathbf{M}^v\dot{\mathbf{R}}^v\dot{\mathbf{z}}\right) = \begin{bmatrix} \mathbf{b}_1^{v\mathrm{T}} & \mathbf{b}_{2,1}^{v\mathrm{T}} & \mathbf{b}_{3,1}^{v\mathrm{T}} & \mathbf{b}_{4,1}^{v\mathrm{T}} & \mathbf{b}_{5,1}^{v\mathrm{T}} & \mathbf{b}_{6,1}^{v\mathrm{T}} \\ \mathbf{0} & \mathbf{b}_2^{v\mathrm{T}} & \mathbf{b}_{3,2}^{v\mathrm{T}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{b}_3^{v\mathrm{T}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{b}_4^{v\mathrm{T}} & \mathbf{b}_{5,4}^{v\mathrm{T}} & \mathbf{b}_{6,4}^{v\mathrm{T}} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{b}_5^{v\mathrm{T}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{b}_6^{v\mathrm{T}} \end{bmatrix} \begin{bmatrix} \mathbf{Q}_1^v - \mathbf{M}_1^v\mathbf{d}_1^v \\ \mathbf{Q}_2^v - \mathbf{M}_2^v\mathbf{d}_2^{v\Sigma} \\ \mathbf{Q}_3^v - \mathbf{M}_3^v\mathbf{d}_3^{v\Sigma} \\ \mathbf{Q}_4^v - \mathbf{M}_4^v\mathbf{d}_4^{v\Sigma} \\ \mathbf{Q}_5^v - \mathbf{M}_5^v\mathbf{d}_5^{v\Sigma} \\ \mathbf{Q}_6^v - \mathbf{M}_6^v\mathbf{d}_6^{v\Sigma} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1^{v\mathrm{T}}\mathbf{Q}_1^{v\Sigma} \\ \mathbf{b}_2^{v\mathrm{T}}\mathbf{Q}_2^{v\Sigma} \\ \mathbf{b}_3^{v\mathrm{T}}\mathbf{Q}_v^{3\Sigma} \\ \mathbf{b}_4^{v\mathrm{T}}\mathbf{Q}_4^{v\Sigma} \\ \mathbf{b}_5^{v\mathrm{T}}\mathbf{Q}_5^{v\Sigma} \\ \mathbf{b}_6^{v\mathrm{T}}\mathbf{Q}_6^{v\Sigma} \end{bmatrix} \tag{2.218a}$$

$$\mathbf{Q}_6^{v\Sigma} = \mathbf{Q}_6^v - \mathbf{M}_6^v\mathbf{d}_6^{v\Sigma} \tag{2.218b}$$

$$\mathbf{Q}_5^{v\Sigma} = \mathbf{Q}_5^v - \mathbf{M}_5^v\mathbf{d}_5^{v\Sigma} \tag{2.218c}$$

$$\mathbf{Q}_4^{v\Sigma} = \mathbf{Q}_4^v - \mathbf{M}_4^v\mathbf{d}_4^{v\Sigma} + \mathbf{B}_5^{v\mathrm{T}}\mathbf{Q}_5^{v\Sigma} + \mathbf{B}_6^{v\mathrm{T}}\mathbf{Q}_6^{v\Sigma} \tag{2.218d}$$

$$\mathbf{Q}_3^{v\Sigma} = \mathbf{Q}_3^v - \mathbf{M}_3^v\mathbf{d}_3^{v\Sigma} \tag{2.218e}$$

$$\mathbf{Q}_2^{v\Sigma} = \mathbf{Q}_2^v - \mathbf{M}_2^v\mathbf{d}_2^{v\Sigma} + \mathbf{B}_3^{v\mathrm{T}}\mathbf{Q}_3^{v\Sigma} \tag{2.218f}$$

$$\mathbf{Q}_1^{v\Sigma} = \mathbf{Q}_1^v - \mathbf{M}_1^v\mathbf{d}_1^v + \mathbf{B}_2^{v\mathrm{T}}\mathbf{Q}_2^{v\Sigma} + \mathbf{B}_4^{v\mathrm{T}}\mathbf{Q}_4^{v\Sigma} \tag{2.218g}$$

The vector of generalized forces can be expressed in compact form as follows,

$$\mathbf{Q}_i^d = \mathbf{R}_i^{v\mathrm{T}}\left(\mathbf{Q}^v - \mathbf{M}^v\dot{\mathbf{R}}^v\dot{\mathbf{z}}\right) = \mathbf{b}_i^{v\mathrm{T}}\mathbf{Q}_i^{v\Sigma}, \tag{2.219a}$$

$$\mathbf{Q}_i^{v\Sigma} = \mathbf{Q}_i^v - \mathbf{M}_i^v\mathbf{d}_i^{v\Sigma} + \sum_{s=1}^{n_s^i}\mathbf{B}_s^{v\mathrm{T}}\mathbf{Q}_s^{v\Sigma} \tag{2.219b}$$

where $n_s^i$ is again the number of children of body $i$. Behold that forces, alike masses, are accumulated from the leaves of the mechanisms to the root through a recursive contribution from each children's body.

The construction of the general mass matrix and generalized forces vector of a relative coordinate model in the framework of a semi-recursive method constitutes the bulk of the evaluation of the dynamics of open-loop systems. However, once posed the equations of motion, they have to be numerically solved, for what two methods are proposed.

#### 2.4.1.1 Fixed point solution

The ODE presented in (2.209) can be directly solved using a fixed point scheme in accelerations. In the case of unconstrained open-loop systems, or which is the same, open-loop systems with independent minimal coordinates, the mass matrix $\mathbf{M}^d$ always has an inverse for physically well defined problems. Thus, the accelerations of the relative coordinates can be directly obtained as:

$$\ddot{\mathbf{z}} = \left[\mathbf{M}^d\right]^{-1}\mathbf{Q}^d \tag{2.220}$$

Once the accelerations are known, positions and velocities can be obtained from a numerical integrator. In the MBSLIM multibody library, there are multiple numerical integration possibilities, such as Hilbert-Hughes-Taylor integrator, generalized-$\alpha$ integrator or the implicit Newmark's family integrators, being the latest the option currently explored. The Newmark's family unfold a method to compute positions and velocities from accelerations based on the time step $h$ and two additional parameters $\gamma$ and $\beta$, which govern its stability, accuracy and energy conservation:

$$\mathbf{z}_{n+1} = \mathbf{z}_n + h\dot{\mathbf{z}}_n + \frac{h^2}{2}\left\{(1-2\beta)\,\ddot{\mathbf{z}}_n + 2\beta\ddot{\mathbf{z}}_{n+1}\right\} \tag{2.221a}$$

$$\dot{\mathbf{z}}_{n+1} = \dot{\mathbf{z}}_n + h\left\{(1-\gamma)\,\ddot{\mathbf{z}}_n + \gamma\ddot{\mathbf{z}}_{n+1}\right\} \tag{2.221b}$$

This type of numerical integrator is applied to ODEs and DAEs in a predictor-corrector scheme. Once computed the states for a time step, a prediction of the states at the following time step is required. For a Newmark's integrator, the predictor stage implemented takes the form:

$$\mathbf{z}_{n+1} = \mathbf{z}_n + h\dot{\mathbf{z}}_n + \frac{h^2}{2}\ddot{\mathbf{z}}_n \tag{2.222a}$$

$$\dot{\mathbf{z}}_{n+1} = \dot{\mathbf{z}}_n + h\ddot{\mathbf{z}}_n \tag{2.222b}$$

Behold that the predictor selected is independent of the values of $\gamma$ and $\beta$. Regarding (2.221), those coefficients represent scalar factors penalizing the weight of current and past accelerations in the framework of an implicit rule of integration. On the contrary, the predictor constitutes an explicit step, and therefore, $\gamma$ and $\beta$ do not play any role there.

#### 2.4.1.2 Newton-Raphson solution

The general method used by default in MBSLIM to solve (2.209) involves a Newton-Raphson (NR) method primarily due to the reuse of the scheme applied to constrained systems. Moreover, the NR method is more robust than the fixed-point scheme in some problems involving contact or very stiff forces.

Similarly to section 2.4.1.1, a Newmark's family integrator with a fixed time step is used in this section, due to its simplicity and its good behavior in the integration of DAEs and ODEs, specially in terms of accuracy, stability and energy conservation.

The properties and performance of this integrator applied to multibody systems are profusely studied in [54]. Velocities and accelerations can be expressed in terms of positions by applying a Newmark's family integrator with the following relations:

$$\dot{\mathbf{z}}_{n+1} = \frac{\gamma}{\beta h}\mathbf{z}_{n+1} + \hat{\dot{\mathbf{z}}}_n; \qquad \hat{\dot{\mathbf{z}}}_n = -\left(\frac{\gamma}{\beta h}\mathbf{z}_n + \left(\frac{\gamma}{\beta} - 1\right)\dot{\mathbf{z}}_n + \left(\frac{\gamma}{2\beta} - 1\right)h\ddot{\mathbf{z}}_n\right) \tag{2.223a}$$

$$\ddot{\mathbf{z}}_{n+1} = \frac{1}{\beta h^2}\mathbf{z}_{n+1} + \hat{\ddot{\mathbf{z}}}_n; \qquad \hat{\ddot{\mathbf{z}}}_n = -\left(\frac{1}{\beta h^2}\mathbf{z}_n + \frac{1}{\beta h}\dot{\mathbf{z}}_n + \left(\frac{1}{2\beta} - 1\right)\ddot{\mathbf{z}}_n\right) \tag{2.223b}$$

where the subscript indicates the time step.

If the previous coefficients are selected as $\beta = \frac{1}{4}$ and $\gamma = \frac{1}{2}$, the particular expressions of the implicit trapezoidal rule are obtained.

Now, let us recall here (2.209), identifying its dependencies:

$$\mathbf{M}^d(\mathbf{z})\,\ddot{\mathbf{z}} = \mathbf{Q}^d(\mathbf{z}, \dot{\mathbf{z}}, t) \tag{2.224}$$

Then, the application of the NR scheme in positions yields:

$$\frac{\mathrm{d}}{\mathrm{d}\mathbf{z}}\left(\mathbf{M}^d\ddot{\mathbf{z}} - \mathbf{Q}^d\right)\Delta\mathbf{z} = \mathbf{Q}^d - \mathbf{M}^d\ddot{\mathbf{z}} \tag{2.225}$$

Expanding the derivative:

$$\left(\frac{\partial \mathbf{M}^d}{\partial \mathbf{z}}\ddot{\mathbf{z}} + \mathbf{M}^d\frac{\partial \ddot{\mathbf{z}}}{\partial \mathbf{z}} - \frac{\partial \mathbf{Q}^d}{\partial \mathbf{z}} - \frac{\partial \mathbf{Q}^d}{\partial \dot{\mathbf{z}}}\frac{\partial \dot{\mathbf{z}}}{\partial \mathbf{z}}\right)\Delta\mathbf{z} = \mathbf{Q}^d - \mathbf{M}^d\ddot{\mathbf{z}} \tag{2.226}$$

where forces are assumed to be exclusively dependent on $\mathbf{z}$, $\dot{\mathbf{z}}$ and the time and masses on $\mathbf{z}$ and the time.

Looking at the expressions of the Newmark's family integrator (2.223), the following relations among positions, velocities and accelerations are obtained:

$$\dot{\mathbf{z}}_{n+1} = \frac{\gamma}{\beta h}\mathbf{z}_{n+1} + \hat{\dot{\mathbf{z}}}_n \Rightarrow \frac{\partial \dot{\mathbf{z}}}{\partial \mathbf{z}} = \frac{\gamma}{\beta h} \tag{2.227a}$$

$$\ddot{\mathbf{z}}_{n+1} = \frac{1}{\beta h^2}\mathbf{z}_{n+1} + \hat{\ddot{\mathbf{z}}}_n \Rightarrow \frac{\partial \ddot{\mathbf{z}}}{\partial \mathbf{z}} = \frac{1}{\beta h^2} \tag{2.227b}$$

Now, substituting (2.227) in (2.226), and scaling it by $\beta h^2$:

$$\beta h^2\left(\frac{\partial \mathbf{M}^d}{\partial \mathbf{z}}\ddot{\mathbf{z}} + \mathbf{M}^d\frac{1}{\beta h^2} - \frac{\partial \mathbf{Q}^d}{\partial \mathbf{z}} - \frac{\partial \mathbf{Q}^d}{\partial \dot{\mathbf{z}}}\frac{\gamma}{\beta h}\right)\Delta\mathbf{z} = \beta h^2\left(\mathbf{Q}^d - \mathbf{M}^d\ddot{\mathbf{z}}\right) \tag{2.228}$$

A Newton-Raphson iterative scheme of solution of a nonlinear system can be tackled with an approximate tangent matrix instead of an exact one with a minimum impact on convergence or number of iterations, as discussed in [54]. However, the determination of the terms which can be neglected varies with the system, and has to be carefully studied prior to any general solution. In this particular case, the effect of $\dfrac{\partial \mathbf{M}^d}{\partial \mathbf{z}}$ is negligible, thus it can be eliminated from the composition of the tangent matrix without impairing convergence but lightening the computational burden. Consequently, the new Newton-Raphson scheme becomes:

$$\left(\mathbf{M}^d + \beta h^2 \mathbf{K} + \gamma h \mathbf{C}\right) \Delta \mathbf{z} = \beta h^2 \left(\mathbf{Q}^d - \mathbf{M}^d \ddot{\mathbf{z}}\right) \tag{2.229}$$

being $\mathbf{K} = -\dfrac{\partial \mathbf{Q}^d}{\partial \mathbf{z}}$ the equivalent stiffness matrix of the mechanism and $\mathbf{C} = -\dfrac{\partial \mathbf{Q}^d}{\partial \dot{\mathbf{z}}}$ the equivalent damping matrix. Moreover, these two derivatives of the generalized forces vector can be calculated approximately too, but this will be addressed in a future chapter discussing the implementation of the presented methods in MBSLIM.

The algorithm used to solve the dynamics of topological multibody models with a semi-recursive approach in the multibody library MBSLIM is depicted in Figure 2.11. Each one of the stages of this flowchart is described below.

1. Solution of the initial position and velocity problems. If the selection of degrees of freedom matches the joint coordinates, the position and velocity values of the joint coordinates are directly the values of the degrees of freedom. Otherwise, equations (2.190) and (2.192) have to be resorted to determine the values of the joint coordinates. After that, positions and velocities of points and vectors can be recursively updated, and then the recursive relations described in section 2.2 can be computed to allow the accumulation of forces and masses for the dynamic acceleration problem.

2. Solution of the dynamic problem of accelerations. Since there is no constraint in the model, the accelerations can be directly obtained from (2.209).

3. Prediction of positions using (2.222a) and prediction of velocities and accelerations with (2.223).

4. Assessment of points and vectors, and the recursive relations of each joint.

5. Accumulation and assembly of forces and masses to compose the approximated tangent matrix and residual.

6. Solution of equation (2.229).

7. Updating of velocities and accelerations by means of (2.227).

8. Evaluation of error. There are different possible measurements of error, being the 2-norm of the increment in the positions $\Delta \mathbf{z}$ the one implemented. If the error is higher than a specified tolerance, return to step 4.

9. Time loop. If $t < t_{end}$, return to step 3.

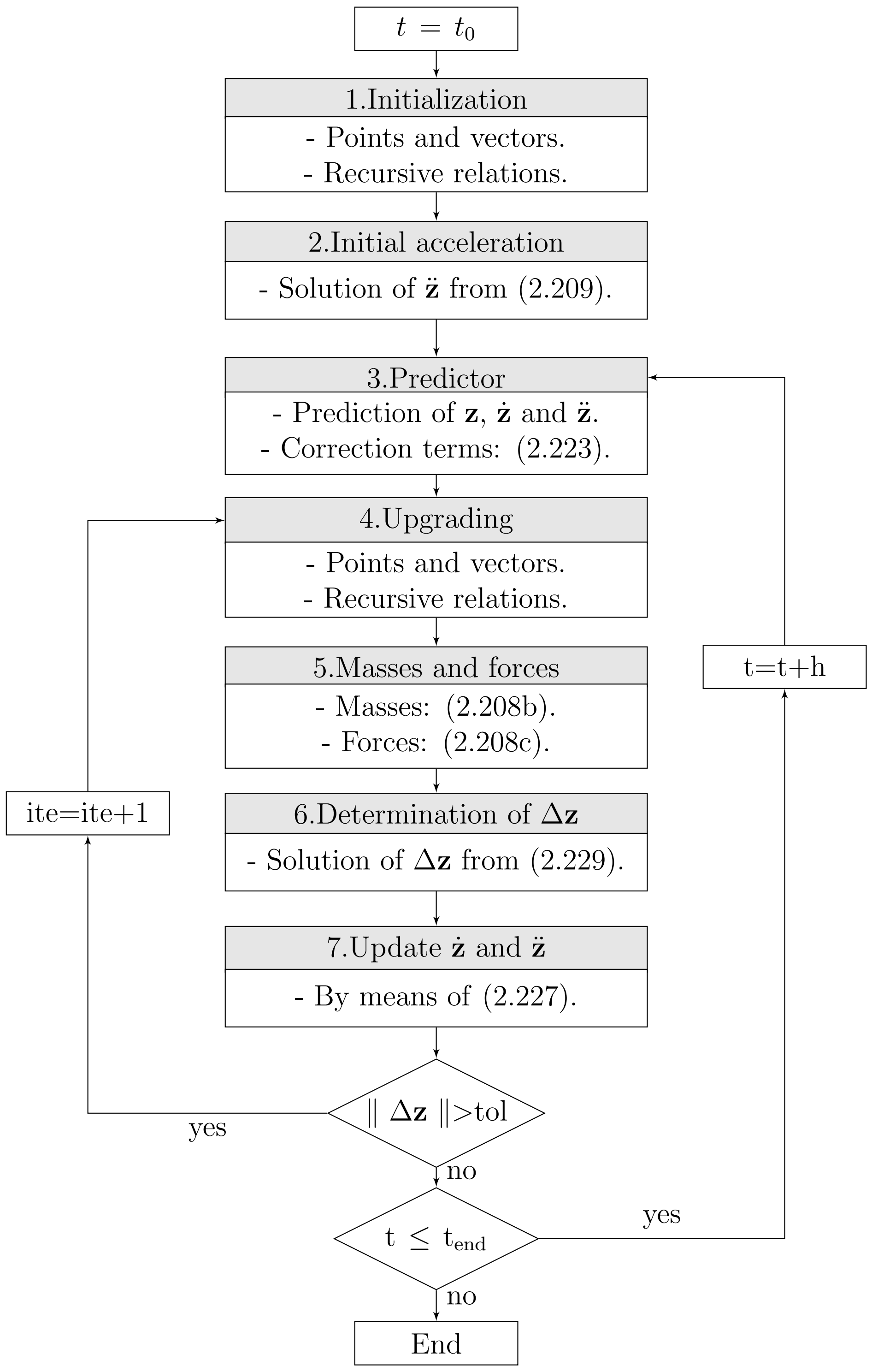

$t = t_0$
1.Initialization
- Points and vectors.
- Recursive relations.
2.Initial acceleration
- Solution of $\ddot{\mathbf{z}}$ from (2.209).
3.Predictor
- Prediction of $\mathbf{z}$, $\dot{\mathbf{z}}$ and $\ddot{\mathbf{z}}$.
- Correction terms: (2.223).
4.Upgrading
- Points and vectors.
- Recursive relations.
5.Masses and forces
- Masses: (2.208b).
- Forces: (2.208c).
6.Determination of $\Delta\mathbf{z}$
- Solution of $\Delta\mathbf{z}$ from (2.229).
7.Update $\dot{\mathbf{z}}$ and $\ddot{\mathbf{z}}$
- By means of (2.227).
$\| \Delta\mathbf{z} \| >$tol
yes
no
ite=ite+1
$\mathrm{t} \leq \mathrm{t}_{\mathrm{end}}$
yes
no
t=t+h
End

Figure 2.11: NR algorithm for the semi-recursive dynamics of open-loop systems

### 2.4.2 Fully-recursive method

The fully-recursive method presented in this section is a generalization of the fully-recursive formulations presented in [116][5], named RTdyn2 formulation in [45] and equivalent to the formulation derived in [35] for open-loop systems. The starting point of the present development is valid for any set of reference points, while the previous works use the particular case of each reference point placed at the origin of coordinates. For this reason, this formulation is not referred to as RTdyn2, but as a general fully-recursive approach.

The development of the fully-recursive expressions starts with the virtual power equations (2.203). Splitting the last two terms:

$$\sum_{i=1}^{n_b-2} \mathbf{V}_i^{*\mathrm{T}} \left[\mathbf{M}_i^v \dot{\mathbf{V}}_i - \mathbf{Q}_i^v\right] + \mathbf{V}_{n_b-1}^{*\mathrm{T}} \left[\mathbf{M}_{n_b-1}^v \dot{\mathbf{V}}_{n_b-1} - \mathbf{Q}_{n_b-1}^v\right] + \mathbf{V}_{n_b}^{*\mathrm{T}} \left[\mathbf{M}_{n_b}^v \dot{\mathbf{V}}_{n_b} - \mathbf{Q}_{n_b}^v\right] = 0 \tag{2.230}$$

Equations (2.113) and (2.114) for $i = n_b$:

$$\mathbf{V}_{n_b}^* = \mathbf{B}_{n_b}^v \mathbf{V}_{n_b-1}^* + \mathbf{b}_{n_b}^v \dot{\mathbf{z}}_{n_b}^* \tag{2.231}$$

$$\dot{\mathbf{V}}_{n_b} = \mathbf{B}_{n_b}^v \dot{\mathbf{V}}_{n_b-1} + \mathbf{b}_{n_b}^v \ddot{\mathbf{z}}_{n_b} + \mathbf{d}_{n_b}^v \tag{2.232}$$

Applying the previous recursive relations to the last term in (2.230)

$$\begin{aligned} &\sum_{i=1}^{n_b-2} \mathbf{V}_i^{*\mathrm{T}} \left[\mathbf{M}_i^v \dot{\mathbf{V}}_i - \mathbf{Q}_i^v\right] + \mathbf{V}_{n_b-1}^{*\mathrm{T}} \left[\mathbf{M}_{n_b-1}^v \dot{\mathbf{V}}_{n_b-1} - \mathbf{Q}_{n_b-1}^v\right] \\ &+ \left(\mathbf{B}_{n_b}^v \mathbf{V}_{n_b-1}^* + \mathbf{b}_{n_b}^v \dot{\mathbf{z}}_{n_b}^*\right)^{\mathrm{T}} \left[\mathbf{M}_{n_b}^v \dot{\mathbf{V}}_{n_b} - \mathbf{Q}_{n_b}^v\right] = \sum_{i=1}^{n_b-2} \mathbf{V}_i^{*\mathrm{T}} \left[\mathbf{M}_i^v \dot{\mathbf{V}}_i - \mathbf{Q}_i^v\right] \\ &+ \mathbf{V}_{n_b-1}^{*\mathrm{T}} \left[\mathbf{M}_{n_b-1}^v \dot{\mathbf{V}}_{n_b-1} - \mathbf{Q}_{n_b-1}^v\right] + \mathbf{V}_{n_b-1}^{*\mathrm{T}} \mathbf{B}_{n_b}^{v\mathrm{T}} \left[\mathbf{M}_{n_b}^v \dot{\mathbf{V}}_{n_b} - \mathbf{Q}_{n_b}^v\right] \\ &+ \dot{\mathbf{z}}_{n_b}^{*\mathrm{T}} \mathbf{b}_{n_b}^{v\mathrm{T}} \left[\mathbf{M}_{n_b}^v \left(\mathbf{B}_{n_b}^v \dot{\mathbf{V}}_{n_b-1} + \mathbf{b}_{n_b}^v \ddot{\mathbf{z}}_{n_b} + \mathbf{d}_{n_b}^v\right) - \mathbf{Q}_{n_b}^v\right] = 0 \end{aligned} \tag{2.233}$$

Taking into account that the joint coordinates are assumed to be independent, the factor affected by the virtual velocity $\dot{\mathbf{z}}_{n_b}^*$ has to vanish, resulting in the following equation,

$$\left[\mathbf{b}_{n_b}^{v\mathrm{T}} \mathbf{M}_{n_b}^v \mathbf{b}_{n_b}^v\right] \ddot{\mathbf{z}}_{n_b} = \mathbf{b}_{n_b}^{v\mathrm{T}} \left[\mathbf{Q}_{n_b}^v - \mathbf{M}_{n_b}^v \left(\mathbf{B}_{n_b}^v \dot{\mathbf{V}}_{n_b-1} + \mathbf{d}_{n_b}^v\right)\right] \tag{2.234}$$

[5] In [116] the reference point is always the origin while the expressions provided here are general, for any reference point.

Assuming a non-singular leading matrix in (2.234)[6]

$$\ddot{\mathbf{z}}_{n_b} = \left[\mathbf{b}_{n_b}^{v\mathrm{T}} \mathbf{M}_{n_b}^{v} \mathbf{b}_{n_b}^{v}\right]^{-1} \mathbf{b}_{n_b}^{v\mathrm{T}} \left[\mathbf{Q}_{n_b}^{v} - \mathbf{M}_{n_b}^{v} \left(\mathbf{B}_{n_b}^{v} \dot{\mathbf{V}}_{n_b-1} + \mathbf{d}_{n_b}^{v}\right)\right] \tag{2.236}$$

The remaining terms in equation (2.233) become,

$$\sum_{i=1}^{n_b-2} \mathbf{V}_i^{*\mathrm{T}} \left[\mathbf{M}_i^v \dot{\mathbf{V}}_i - \mathbf{Q}_i^v\right] + \mathbf{V}_{n_b-1}^{*\mathrm{T}} \left[\mathbf{M}_{n_b-1}^{v} \dot{\mathbf{V}}_{n_b-1} + \mathbf{B}_{n_b}^{v\mathrm{T}} \mathbf{M}_{n_b}^{v} \dot{\mathbf{V}}_{n_b} - \mathbf{Q}_{n_b-1}^{v} - \mathbf{B}_{n_b}^{v\mathrm{T}} \mathbf{Q}_{n_b}^{v}\right] = 0 \tag{2.237}$$

which, after replacing (2.232),

$$\begin{aligned}\sum_{i=1}^{n_b-2} \mathbf{V}_i^{*\mathrm{T}} \left[\mathbf{M}_i^v \dot{\mathbf{V}}_i - \mathbf{Q}_i^v\right] + \mathbf{V}_{n_b-1}^{*\mathrm{T}} \Big[\left(\mathbf{M}_{n_b-1}^{v} + \mathbf{B}_{n_b}^{v\mathrm{T}} \mathbf{M}_{n_b}^{v} \mathbf{B}_{n_b}^{v}\right) \dot{\mathbf{V}}_{n_b-1} + \mathbf{B}_{n_b}^{v\mathrm{T}} \mathbf{M}_{n_b}^{v} \mathbf{b}_{n_b}^{v} \ddot{\mathbf{z}}_{n_b} \\ - \mathbf{Q}_{n_b-1}^{v} - \mathbf{B}_{n_b}^{v\mathrm{T}} \left(\mathbf{Q}_{n_b}^{v} - \mathbf{M}_{n_b}^{v} \mathbf{d}_{n_b}^{v}\right)\Big] = 0\end{aligned} \tag{2.238}$$

and (2.236),

$$\begin{aligned}\sum_{i=1}^{n_b-2} \mathbf{V}_i^{*\mathrm{T}} \left[\mathbf{M}_i^v \dot{\mathbf{V}}_i - \mathbf{Q}_i^v\right] + \mathbf{V}_{n_b-1}^{*\mathrm{T}} \Big[\left(\mathbf{M}_{n_b-1}^{v} + \mathbf{B}_{n_b}^{v\mathrm{T}} \mathbf{M}_{n_b}^{v} \mathbf{B}_{n_b}^{v}\right) \dot{\mathbf{V}}_{n_b-1} \\ + \mathbf{B}_{n_b}^{v\mathrm{T}} \mathbf{M}_{n_b}^{v} \mathbf{b}_{n_b}^{v} \left(\left[\mathbf{b}_{n_b}^{v\mathrm{T}} \mathbf{M}_{n_b}^{v} \mathbf{b}_{n_b}^{v}\right]^{-1} \mathbf{b}_{n_b}^{v\mathrm{T}} \left[\mathbf{Q}_{n_b}^{v} - \mathbf{M}_{n_b}^{v} \left(\mathbf{B}_{n_b}^{v} \dot{\mathbf{V}}_{n_b-1} + \mathbf{d}_{n_b}^{v}\right)\right]\right) \\ - \mathbf{Q}_{n_b-1}^{v} - \mathbf{B}_{n_b}^{v\mathrm{T}} \left(\mathbf{Q}_{n_b}^{v} - \mathbf{M}_{n_b}^{v} \mathbf{d}_{n_b}^{v}\right)\Big] = 0\end{aligned} \tag{2.239}$$

Grouping terms,

$$\begin{aligned}\sum_{i=1}^{n_b-2} \mathbf{V}_i^{*\mathrm{T}} \left[\mathbf{M}_i^v \dot{\mathbf{V}}_i - \mathbf{Q}_i^v\right] \\ + \mathbf{V}_{n_b-1}^{*\mathrm{T}} \Big\{\Big[\mathbf{M}_{n_b-1}^{v} + \mathbf{B}_{n_b}^{v\mathrm{T}} \left(\mathbf{I}_6 - \mathbf{M}_{n_b}^{v} \mathbf{b}_{n_b}^{v} \left[\mathbf{b}_{n_b}^{v\mathrm{T}} \mathbf{M}_{n_b}^{v} \mathbf{b}_{n_b}^{v}\right]^{-1} \mathbf{b}_{n_b}^{v\mathrm{T}}\right) \mathbf{M}_{n_b}^{v} \mathbf{B}_{n_b}^{v}\Big] \dot{\mathbf{V}}_{n_b-1} \\ - \mathbf{Q}_{n_b-1}^{v} - \mathbf{B}_{n_b}^{v\mathrm{T}} \left(\mathbf{I}_6 - \mathbf{M}_{n_b}^{v} \mathbf{b}_{n_b}^{v} \left[\mathbf{b}_{n_b}^{v\mathrm{T}} \mathbf{M}_{n_b}^{v} \mathbf{b}_{n_b}^{v}\right]^{-1} \mathbf{b}_{n_b}^{v\mathrm{T}}\right) \left(\mathbf{Q}_{n_b}^{v} - \mathbf{M}_{n_b}^{v} \mathbf{d}_{n_b}^{v}\right)\Big\} = 0\end{aligned} \tag{2.240}$$

[6]It is important to remark that the leading matrix (2.234) is not always invertible. The most common case is when joint $n_b$ is a spherical (or floating) joint modeled with non-minimal joint coordinates like the Euler parameters proposed along this work. In these cases the following equation derived from the normalization constraint (2.62) completes system (2.234) to reach full rank.

$$\bar{\mathbf{p}}^{\mathrm{T}} \dot{\bar{\mathbf{p}}} = 0 \Rightarrow \dot{\bar{\mathbf{p}}}^{\mathrm{T}} \dot{\bar{\mathbf{p}}} + \bar{\mathbf{p}}^{\mathrm{T}} \ddot{\bar{\mathbf{p}}} = 0 \Rightarrow \bar{\mathbf{p}}^{\mathrm{T}} \ddot{\bar{\mathbf{p}}} = -\dot{\bar{\mathbf{p}}}^{\mathrm{T}} \dot{\bar{\mathbf{p}}} \tag{2.235}$$

We arrive at the following expression not dependent on the accelerations $\ddot{\mathbf{z}}_{n_b}$

$$\sum_{i=1}^{n_b-2} \mathbf{V}_i^{*\mathrm{T}} \left[\mathbf{M}_i^v \dot{\mathbf{V}}_i - \mathbf{Q}_i^v\right] + \mathbf{V}_{n_b-1}^{*\mathrm{T}} \left\{\hat{\mathbf{M}}_{n_b-1}^v \dot{\mathbf{V}}_{n_b-1} - \hat{\mathbf{Q}}_{n_b-1}^v\right\} = 0 \tag{2.241}$$

$$\hat{\mathbf{M}}_{n_b-1}^v = \mathbf{M}_{n_b-1}^v + \mathbf{B}_{n_b}^{v\mathrm{T}} \mathbf{K}_{n_b} \mathbf{M}_{n_b}^v \mathbf{B}_{n_b}^v \tag{2.242}$$

$$\hat{\mathbf{Q}}_{n_b-1}^v = \mathbf{Q}_{n_b-1}^v + \mathbf{B}_{n_b}^{v\mathrm{T}} \mathbf{K}_{n_b} \left(\mathbf{Q}_{n_b}^v - \mathbf{M}_{n_b}^v \mathbf{d}_{n_b}^v\right) \tag{2.243}$$

$$\mathbf{K}_{n_b} = \mathbf{I}_6 - \mathbf{M}_{n_b}^v \mathbf{b}_{n_b}^v \left[\mathbf{b}_{n_b}^{v\mathrm{T}} \mathbf{M}_{n_b}^v \mathbf{b}_{n_b}^v\right]^{-1} \mathbf{b}_{n_b}^{v\mathrm{T}} \tag{2.244}$$

Observe that the previous expression is extended only to the first $n_b - 1$ bodies and the effect of body $n_b$ is taken into account by means of the articulated inertia $\hat{\mathbf{M}}_{n_b-1}^v$ and forces $\hat{\mathbf{Q}}_{n_b-1}^v$. This same procedure can be extended recursively towards the root of the kinematic tree, giving rise to the following expressions for body $i$,

$$\hat{\mathbf{M}}_{i-1}^v = \mathbf{M}_{i-1}^v + \mathbf{B}_i^{v\mathrm{T}} \mathbf{K}_i \hat{\mathbf{M}}_i^v \mathbf{B}_i^v \tag{2.245a}$$

$$\hat{\mathbf{Q}}_{i-1}^v = \mathbf{Q}_{i-1}^v + \mathbf{B}_i^{v\mathrm{T}} \mathbf{K}_i \left(\hat{\mathbf{Q}}_i^v - \hat{\mathbf{M}}_i^v \mathbf{d}_i^v\right) \tag{2.245b}$$

$$\mathbf{K}_i = \mathbf{I}_6 - \hat{\mathbf{M}}_i^v \mathbf{b}_i^v \left[\mathbf{b}_i^{v\mathrm{T}} \hat{\mathbf{M}}_i^v \mathbf{b}_i^v\right]^{-1} \mathbf{b}_i^{v\mathrm{T}} \tag{2.245c}$$

$$\ddot{\mathbf{z}}_i = \left[\mathbf{b}_i^{v\mathrm{T}} \hat{\mathbf{M}}_i^v \mathbf{b}_i^v\right]^{-1} \mathbf{b}_i^{v\mathrm{T}} \left[\hat{\mathbf{Q}}_i^v - \hat{\mathbf{M}}_i^v \left(\mathbf{B}_i^v \dot{\mathbf{V}}_{i-1} + \mathbf{d}_i^v\right)\right] \tag{2.245d}$$

### 2.4.2.1 Generalized forces applied directly on joint coordinates

The previous set of equations and recursive accumulation have to be modified and extended for the case of forces or torques applied directly on kinematic joints. Let us consider for example the case of a mechanism with a revolute joint and a motor coupled to it and generating a specific torque which varies with time. This torque can be directly applied to the joint coordinate $\mathbf{z}_i$ instead of assembling it on the reference point coordinates $\mathbf{V}_i$ of body $i$.

Returning to (2.230), an additional term appears if generalized forces are directly applied on the joints:

$$\sum_{i=1}^{n_b} \left(\mathbf{V}_i^{*\mathrm{T}} \left[\mathbf{M}_i^v \dot{\mathbf{V}}_i - \mathbf{Q}_i^v\right] - \dot{\mathbf{z}}_i^{*\mathrm{T}} \mathbf{Q}_i^{\mathbf{z}}\right) = 0 \tag{2.246}$$

Applying the same scheme of accumulation introduced in 2.4.2, the general expressions including this type of forces are:

$$\hat{\mathbf{M}}_{i-1}^v = \mathbf{M}_{i-1}^v + \mathbf{B}_i^{v\mathrm{T}} \mathbf{K}_i \hat{\mathbf{M}}_i^v \mathbf{B}_i^v \tag{2.247a}$$

$$\hat{\mathbf{Q}}_{i-1}^v = \mathbf{Q}_{i-1}^v + \mathbf{B}_i^{v\mathrm{T}} \left(\mathbf{K}_i \left(\hat{\mathbf{Q}}_i^v - \hat{\mathbf{M}}_i^v \mathbf{d}_i^v\right) + \mathbf{K}_i^{\mathbf{z}} \mathbf{Q}_i^{\mathbf{z}}\right) \tag{2.247b}$$

$$\mathbf{K}_i^{\mathbf{z}} = -\hat{\mathbf{M}}_i^v \mathbf{b}_i^v \left[\mathbf{b}_i^{v\mathrm{T}} \hat{\mathbf{M}}_i^v \mathbf{b}_i^v\right]^{-1} \tag{2.247c}$$

$$\mathbf{K}_i = \mathbf{I}_6 + \mathbf{K}_i^{\mathbf{z}} \mathbf{b}_i^{v\mathrm{T}} \tag{2.247d}$$

$$\ddot{\mathbf{z}}_i = \left[\mathbf{b}_i^{v\mathrm{T}} \hat{\mathbf{M}}_i^v \mathbf{b}_i^v\right]^{-1} \left(\mathbf{b}_i^{v\mathrm{T}} \left[\hat{\mathbf{Q}}_i^v - \hat{\mathbf{M}}_i^v \left(\mathbf{B}_i^v \dot{\mathbf{V}}_{i-1} + \mathbf{d}_i^v\right)\right] + \mathbf{Q}_i^{\mathbf{z}}\right) \tag{2.247e}$$

where $\mathbf{Q}_i^{\mathbf{z}_i}$ are the forces and\or torques directly applied to the relative coordinates of joint $i$. The new term $\mathbf{K}_i^{\mathbf{z}}$, which allows the accumulation of these forces and torques, is in fact a part of the already introduced accumulation matrix $\mathbf{K}_i$, so no further computations are needed, only the storage of a new matrix.

#### 2.4.2.2 Algorithm

A possible solution procedure begins with the accumulation of masses and forces from the leaves towards the root and the determination of the acceleration of the first body, $\ddot{\mathbf{z}}_1$. From the root, equations (2.114) can be applied recursively towards the leaves in order to calculate all the accelerations $\ddot{\mathbf{z}} = \begin{bmatrix}\ddot{\mathbf{z}}_1 & \ddot{\mathbf{z}}_2 & \cdots & \ddot{\mathbf{z}}_i & \cdots & \ddot{\mathbf{z}}_{n_b}\end{bmatrix}^{\mathrm{T}} = \begin{bmatrix}\ddot{z}_1 & \ddot{z}_2 & \cdots & \ddot{z}_n\end{bmatrix}^{\mathrm{T}}$. After each acceleration is known, the time-stepping equations can be applied (for example, Newmark's equations) to obtain velocities and positions from accelerations. These equations can be applied in a loop until convergence.

It should be remarked that this formulation is specially well suited to be solved in a fixed-point scheme with an integrator which requires accelerations and returns positions and velocities.

The solution process outlined in Figure 2.12 consists in the following steps:

1. Prediction of positions $\mathbf{z}^{n+1}$, velocities $\dot{\mathbf{z}}^{n+1}$ and accelerations $\ddot{\mathbf{z}}^{n+1}$ of the states from the former instant of time $\mathbf{z}^n$, $\dot{\mathbf{z}}^n$ and $\ddot{\mathbf{z}}^n$ using (2.222).

2. Forward recursive kinematics from the root to the leaves, in order to determine positions and velocities of points and vectors required for the composition of the equations of motion by means of (2.121) and the recursive position relations introduced in section 2.1.

3. Backward recursive accumulation of masses and forces with (2.247).

4. Forward solution of accelerations $\ddot{\mathbf{z}}^{n+1}$ by means of (2.247e). This process requires to update the recursive terms $\mathbf{B}_i^v$, $\mathbf{b}_i^v$, $\mathbf{d}_i^v$ and the vector of accelerations $\dot{\mathbf{V}}_i$ just after the solution of each joint acceleration in order to determine the accelerations of the following joint in the kinematic chain.

5. Iterate from step 2 to 4 until convergence. Once the error is below a demanded tolerance, proceed to the following step.

6. Return to step 1 until $t = t_{fin}$.

### 2.4.3 Specific semi-recursive formulations

The expressions and algorithms introduced in sections 2.4.1 and 2.4.2 are valid for any set of reference points. However, there are two particular selections of reference points that enhance these methods due to the inclusion of different simplifications in the accumulation process or in the computation of elemental forces and masses. These two sets of reference points lead to the RTdyn0 and RTdyn1 formulations.

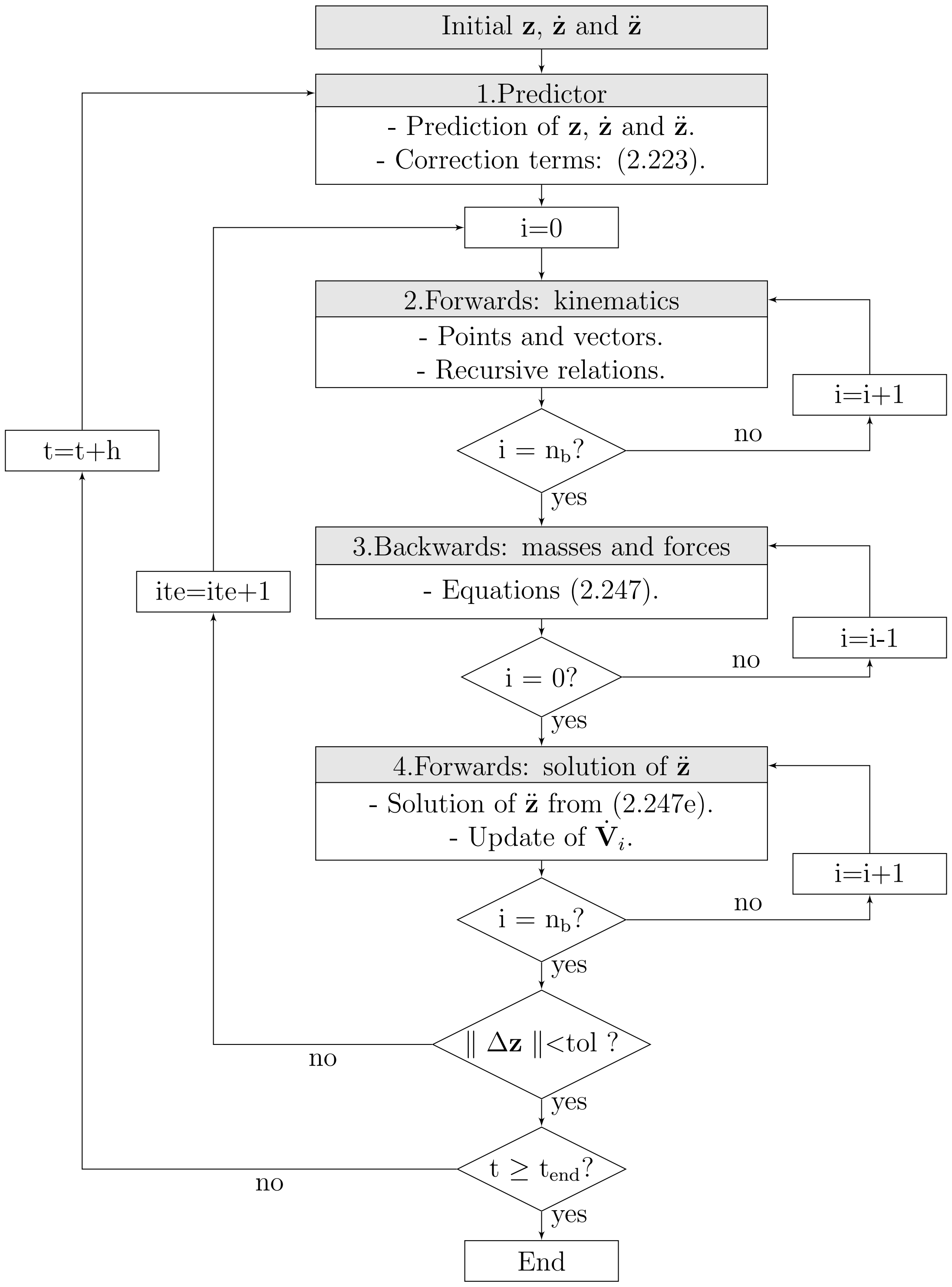

Initial $\mathbf{z}$, $\dot{\mathbf{z}}$ and $\ddot{\mathbf{z}}$
1.Predictor
- Prediction of $\mathbf{z}$, $\dot{\mathbf{z}}$ and $\ddot{\mathbf{z}}$.
- Correction terms: (2.223).
i=0
2.Forwards: kinematics
- Points and vectors.
- Recursive relations.
i=i+1
$i = n_b$?
no
yes
t=t+h
3.Backwards: masses and forces
- Equations (2.247).
ite=ite+1
i=i-1
$i = 0$?
no
yes
4.Forwards: solution of $\ddot{\mathbf{z}}$
- Solution of $\ddot{\mathbf{z}}$ from (2.247e).
- Update of $\dot{\mathbf{V}}_i$.
i=i+1
$i = n_b$?
no
yes
$\| \Delta\mathbf{z} \| <$tol ?
no
yes
$t \geq t_{end}$?
no
yes
End

Figure 2.12: Flowchart for the fully-recursive dynamics of open-loop systems.

Traditionally, RTdyn1 has delivered better performance than RTdyn0 formulations in terms of efficiency, but it has been proved that it entails some drawbacks, such as the addition of numerical errors when the mechanism moves away from the origin. Moreover, the set of reference points used in RTdyn1 is regarded as less natural than RTdyn0, since each reference point is considered rigidly attached to the body to which it makes reference at each instant of time, but it moves inside the local reference frame of the body between different instants of time, while in RTdyn0 the reference point is always fixed in the local reference frame of the body. Bearing in mind sensitivity analysis implications, both methods have been considered and implemented in MBSLIM.

#### 2.4.3.1 RTdyn0

The RTdyn0 formulation results from choosing the CoM of each body as reference point $\mathbf{r}_i = \mathbf{r}_G^i$. The recursive kinematics equations are those derived in section 2.2 for the $\mathbf{Y}$ and $\dot{\mathbf{Y}}$ vectors and the general equations developed in section 2.4 have a simplified form thanks to that $\mathbf{D}_i^v$ becomes $\mathbf{D}_i^y = \mathbf{I}$ and $\mathbf{e}_i^v$ becomes $\mathbf{e}_i^y = \mathbf{0}$. Following the scheme introduced in section 2.4.1, the elemental mass matrix $\mathbf{M}_i^y$ and elemental generalized forces vector $\mathbf{Q}_i^y$ referred to body $i$ take the form:

$$\mathbf{M}_i^y = (\mathbf{D}_i^y)^{\mathrm{T}}\mathbf{M}_i\mathbf{D}_i^y = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_G^i & \mathbf{I} \end{bmatrix} \begin{bmatrix} m_i\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_i^G \end{bmatrix} \begin{bmatrix} \mathbf{I} & \tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_G^i \\ \mathbf{0} & \mathbf{I} \end{bmatrix} = \begin{bmatrix} m_i\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_i^G \end{bmatrix} = \mathbf{M}_i \tag{2.248}$$

$$\mathbf{Q}_i^y = (\mathbf{D}_i^y)^{\mathrm{T}}(\mathbf{Q}_i - \mathbf{M}_i\mathbf{e}_i^y) = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_G^i & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{f}_i - m_i\boldsymbol{\omega}_i \wedge (\boldsymbol{\omega}_i \wedge (\mathbf{r}_G^i - \tilde{\mathbf{r}}_G^i)) \\ \mathbf{n}_i^G - \boldsymbol{\omega}_i \wedge \mathbf{J}_i^G\boldsymbol{\omega}_i \end{bmatrix} = \begin{bmatrix} \mathbf{f}_i \\ \mathbf{n}_i^G - \boldsymbol{\omega}_i \wedge \mathbf{J}_i^G\boldsymbol{\omega}_i \end{bmatrix} = \mathbf{Q}_i \tag{2.249}$$

Then, equation (2.203) becomes:

$$\mathbf{Y}^{*\mathrm{T}}\left[\mathbf{M}^y\dot{\mathbf{Y}} - \mathbf{Q}^y\right] = 0 \tag{2.250}$$

with

$$\mathbf{M}^y = \begin{bmatrix} \mathbf{M}_1^y & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{M}_2^y & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{M}_{n_b}^y \end{bmatrix} \; ; \quad \mathbf{Q}^y = \begin{bmatrix} \mathbf{Q}_1^y \\ \mathbf{Q}_2^y \\ \vdots \\ \mathbf{Q}_{n_b}^y \end{bmatrix} \; ; \quad \mathbf{Y} = \begin{bmatrix} \mathbf{Y}_1 \\ \mathbf{Y}_2 \\ \vdots \\ \mathbf{Y}_{n_b} \end{bmatrix} \tag{2.251}$$

which is identical to (2.194a), as expected.

Equations (2.205) become,

$$\mathbf{Y} = \mathbf{R}^y\dot{\mathbf{z}} \tag{2.252}$$

$$\mathbf{Y}^* = \mathbf{R}^y\dot{\mathbf{z}}^* \tag{2.253}$$

$$\dot{\mathbf{Y}} = \mathbf{R}^y\ddot{\mathbf{z}} + \dot{\mathbf{R}}^y\dot{\mathbf{z}} \tag{2.254}$$

where, again, $\mathbf{z} = \begin{bmatrix}\mathbf{z}_1{}^{\mathrm{T}} & \mathbf{z}_2{}^{\mathrm{T}} & \dots & \mathbf{z}_{n_b}{}^{\mathrm{T}}\end{bmatrix} \in \mathbb{R}^n$ with $n$ the total number of joint coordinates.

And equations (2.208b) and (2.208c) have the following form,

$$\mathbf{M}^d = \mathbf{R}^{y\mathrm{T}}\mathbf{M}^y\mathbf{R}^y \tag{2.255}$$

$$\mathbf{Q}^d = \mathbf{R}^{y\mathrm{T}}\left(\mathbf{Q}^y - \mathbf{M}\dot{\mathbf{R}}^y\dot{\mathbf{z}}\right) \tag{2.256}$$

The recursive calculation of terms in RTdyn0 is identical to that presented for the general formulation, just replacing $v$ with $y$.

The application of RTDyn0 to the fully-recursive approach imply the use of the particular expressions of the recursive relations of each joint particularized for the CoM of each body as reference point and the use of $\mathbf{M}_i^y$ and $\mathbf{Q}_i^y$ as elemental mass matrix and elemental generalized forces vector for each body. This formulation is referred to as fully-recursive RTdyn0 or RTdyn0-FR.

### 2.4.3.2 RTdyn1

The RTdyn1 formulation results from the selection of the point coincident with the global origin of coordinates as reference point $\mathbf{r}_i = \mathbf{r}_0^i$. The recursive kinematics equations are those derived in section 2.2 for the $\mathbf{Z}$ and $\dot{\mathbf{Z}}$ vectors. The equations of this formulation were thoroughly described in [54] but the formulation will be summarized here again.

The following kinematic relations hold, between the reference point, $\mathbf{r}_0^i$, and the CoM, $\mathbf{r}_G^i$,

$$\mathbf{Y}_i^* = \begin{bmatrix}\dot{\mathbf{r}}_G^{i*} \\ \boldsymbol{\omega}_i^*\end{bmatrix} = \begin{bmatrix}\mathbf{I} & -\tilde{\mathbf{r}}_G^i \\ \mathbf{0} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\dot{\mathbf{r}}_0^i \\ \boldsymbol{\omega}_i^*\end{bmatrix} = \mathbf{D}_i\mathbf{Z}_i^* \tag{2.257}$$

$$\dot{\mathbf{Y}}_i = \begin{bmatrix}\ddot{\mathbf{r}}_G^i \\ \dot{\boldsymbol{\omega}}_i\end{bmatrix} = \begin{bmatrix}\mathbf{I} & -\tilde{\mathbf{r}}_G^i \\ \mathbf{0} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\ddot{\mathbf{r}}_0^i \\ \dot{\boldsymbol{\omega}}_i\end{bmatrix} + \begin{bmatrix}\boldsymbol{\omega}_i \wedge (\boldsymbol{\omega}_i \wedge \mathbf{r}_G^i) \\ \mathbf{0}\end{bmatrix} = \mathbf{D}_i\dot{\mathbf{Z}}_i + \mathbf{e}_i \tag{2.258}$$

Applying the previous relations to the virtual power equations (2.194a),

$$\sum_{i=1}^{n_b}\mathbf{Z}_i^{*\mathrm{T}}\left[\left(\mathbf{D}_i^{\mathrm{T}}\mathbf{M}_i\mathbf{D}_i\right)\dot{\mathbf{Z}}_i - \mathbf{D}_i^{\mathrm{T}}\left(\mathbf{Q}_i - \mathbf{M}_i\mathbf{e}_i\right)\right] = 0 \tag{2.259}$$

where,

$$\mathbf{M}_i^z = \mathbf{D}_i^{\mathrm{T}}\mathbf{M}_i\mathbf{D}_i = \begin{bmatrix}\mathbf{I} & \mathbf{0} \\ \tilde{\mathbf{r}}_G^i & \mathbf{I}\end{bmatrix}\begin{bmatrix}m_i\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_i^G\end{bmatrix}\begin{bmatrix}\mathbf{I} & -\tilde{\mathbf{r}}_G^i \\ \mathbf{0} & \mathbf{I}\end{bmatrix} = \begin{bmatrix}m_i\mathbf{I} & -m_i\tilde{\mathbf{r}}_G^i \\ m_i\tilde{\mathbf{r}}_G^i & \mathbf{J}_i^G - m_i\tilde{\mathbf{r}}_G^i\tilde{\mathbf{r}}_G^i\end{bmatrix} \tag{2.260}$$

$$\begin{aligned}\mathbf{Q}_i^z = \mathbf{D}_i^{\mathrm{T}}\left(\mathbf{Q}_i - \mathbf{M}_i\mathbf{e}_i\right) &= \begin{bmatrix}\mathbf{I} & \mathbf{0} \\ \tilde{\mathbf{r}}_G^i & \mathbf{I}\end{bmatrix}\left(\begin{bmatrix}\mathbf{f}_i \\ \mathbf{n}_i^G - \boldsymbol{\omega}_i \wedge \mathbf{J}_i^G\boldsymbol{\omega}_i\end{bmatrix} - \begin{bmatrix}m_i\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_i^G\end{bmatrix}\begin{bmatrix}\boldsymbol{\omega}_i \wedge (\boldsymbol{\omega}_i \wedge \mathbf{r}_G^i) \\ \mathbf{0}\end{bmatrix}\right) \\ &= \begin{bmatrix}\mathbf{f}_i - m_i\boldsymbol{\omega}_i \wedge (\boldsymbol{\omega}_i \wedge \mathbf{r}_G^i) \\ \mathbf{n}_i^G - \boldsymbol{\omega}_i \wedge \mathbf{J}_i^G\boldsymbol{\omega}_i + \mathbf{r}_G^i \wedge \left[\mathbf{f}_i - m_i\boldsymbol{\omega}_i \wedge (\boldsymbol{\omega}_i \wedge \mathbf{r}_G^i)\right]\end{bmatrix}\end{aligned} \tag{2.261}$$

Defining,

$$\mathbf{M}^z = \begin{bmatrix} \mathbf{M}_1^z & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{M}_2^z & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{M}_{n_b}^z \end{bmatrix} \; ; \quad \mathbf{Q}^z = \begin{bmatrix} \mathbf{Q}_1^z \\ \mathbf{Q}_2^z \\ \vdots \\ \mathbf{Q}_{n_b}^z \end{bmatrix} \; ; \quad \mathbf{Z} = \begin{bmatrix} \mathbf{Z}_1 \\ \mathbf{Z}_2 \\ \vdots \\ \mathbf{Z}_{n_b} \end{bmatrix} \tag{2.262}$$

The dynamics equations can be grouped in matrix form,

$$\mathbf{Z}^{*\mathrm{T}} \left[ \mathbf{M}^z \dot{\mathbf{Z}} - \mathbf{Q}^z \right] = 0 \tag{2.263}$$

In RTdyn1, the following linear relations of virtual velocities and real velocities and accelerations hold:

$$\mathbf{Z} = \mathbf{R}^z \dot{\mathbf{z}} \tag{2.264}$$

$$\mathbf{Z}^* = \mathbf{R}^z \dot{\mathbf{z}}^* \tag{2.265}$$

$$\dot{\mathbf{Z}} = \mathbf{R}^z \ddot{\mathbf{z}} + \dot{\mathbf{R}}^z \dot{\mathbf{z}} \tag{2.266}$$

where, again, $\mathbf{z} = \begin{bmatrix} \mathbf{z}_1{}^{\mathrm{T}} & \mathbf{z}_2{}^{\mathrm{T}} & \dots & \mathbf{z}_{n_b}{}^{\mathrm{T}} \end{bmatrix} \in \mathbb{R}^n$ with $n$ the total number of joint coordinates and $n_b$ the number of bodies (or joints).

Then equation (2.263) becomes,

$$\dot{\mathbf{z}}^{*\mathrm{T}} \left[ \left( \mathbf{R}^{z\mathrm{T}} \mathbf{M}^z \mathbf{R}^z \right) \ddot{\mathbf{z}} - \mathbf{R}^{z\mathrm{T}} \left( \mathbf{Q}^z - \mathbf{M}^z \dot{\mathbf{R}}^z \dot{\mathbf{z}} \right) \right] = 0 \tag{2.267}$$

Similarly, the final ODE system of equations of motion (2.209), becomes,

$$\left( \mathbf{R}^{z\mathrm{T}} \mathbf{M}^z \mathbf{R}^z \right) \ddot{\mathbf{z}} = \mathbf{R}^{z\mathrm{T}} \left( \mathbf{Q}^z - \mathbf{M}^z \dot{\mathbf{R}}^z \dot{\mathbf{z}} \right) \tag{2.268}$$

The equations of motion (2.268) involve a new matrix $\mathbf{R}^z$ and a new vector $\dot{\mathbf{R}}^z \dot{\mathbf{z}}$.

Recalling the six-body mechanism of Figure 2.10, the evaluation of these terms is set forth for this particular mechanism as a means to clarify the accumulation process. The recursive application of the relations (2.123a) and (2.123b) to the six-body mechanism yields the following matrix equation:

$$\mathbf{Z} = \begin{pmatrix} \mathbf{Z}_1 \\ \mathbf{Z}_2 \\ \mathbf{Z}_3 \\ \mathbf{Z}_4 \\ \mathbf{Z}_5 \\ \mathbf{Z}_6 \end{pmatrix} = \begin{bmatrix} \mathbf{b}_1^z & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{b}_1^z & \mathbf{b}_2^z & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{b}_1^z & \mathbf{b}_2^z & \mathbf{b}_3^z & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{b}_1^z & \mathbf{0} & \mathbf{0} & \mathbf{b}_4^z & \mathbf{0} & \mathbf{0} \\ \mathbf{b}_1^z & \mathbf{0} & \mathbf{0} & \mathbf{b}_4^z & \mathbf{b}_5^z & \mathbf{0} \\ \mathbf{b}_1^z & \mathbf{0} & \mathbf{0} & \mathbf{b}_4^z & \mathbf{0} & \mathbf{b}_6^z \end{bmatrix} \begin{pmatrix} \dot{\mathbf{z}}_1 \\ \dot{\mathbf{z}}_2 \\ \dot{\mathbf{z}}_3 \\ \dot{\mathbf{z}}_4 \\ \dot{\mathbf{z}}_5 \\ \dot{\mathbf{z}}_6 \end{pmatrix} = \mathbf{R}^z \dot{\mathbf{z}} \tag{2.269}$$

Observe the recursive nature of the previous expression, and that the evaluation of the relative relation (2.123a) for all bodies of a multibody system leads to the previously introduced matrix equation (2.264). Matrix $\mathbf{R}^z$ can be divided in the following matrices (with scarce interest for the studied formulations):

$$\mathbf{R}^z = \begin{bmatrix} \mathbf{I}_6 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{I}_6 & \mathbf{I}_6 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{I}_6 & \mathbf{I}_6 & \mathbf{I}_6 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{I}_6 & \mathbf{0} & \mathbf{0} & \mathbf{I}_6 & \mathbf{0} & \mathbf{0} \\ \mathbf{I}_6 & \mathbf{0} & \mathbf{0} & \mathbf{I}_6 & \mathbf{I}_6 & \mathbf{0} \\ \mathbf{I}_6 & \mathbf{0} & \mathbf{0} & \mathbf{I}_6 & \mathbf{0} & \mathbf{I}_6 \end{bmatrix} \begin{bmatrix} \mathbf{b}_1^z & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{b}_2^z & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{b}_3^z & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{b}_4^z & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{b}_5^z & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{b}_6^z \end{bmatrix} \tag{2.270}$$

For accelerations, comparing equation (2.266) with the recursive application of (2.123a):

$$\dot{\mathbf{R}}^z\dot{\mathbf{z}} = \begin{bmatrix} \mathbf{d}_1^z \\ \mathbf{d}_2^{z\Sigma} \\ \mathbf{d}_3^{z\Sigma} \\ \mathbf{d}_4^{z\Sigma} \\ \mathbf{d}_5^{z\Sigma} \\ \mathbf{d}_6^{z\Sigma} \end{bmatrix} = \begin{bmatrix} \mathbf{d}_1^z \\ \mathbf{d}_2^z + \mathbf{d}_1^z \\ \mathbf{d}_3^z + \mathbf{d}_2^z + \mathbf{d}_1^z \\ \mathbf{d}_4^z + \mathbf{d}_1^z \\ \mathbf{d}_5^z + \mathbf{d}_4^z + \mathbf{d}_1^z \\ \mathbf{d}_6^z + \mathbf{d}_4^z + \mathbf{d}_1^z \end{bmatrix} = \begin{bmatrix} \mathbf{d}_1^z \\ \mathbf{d}_2^z + \mathbf{d}_1^z \\ \mathbf{d}_3^z + \mathbf{d}_2^{z\Sigma} \\ \mathbf{d}_4^z + \mathbf{d}_1^z \\ \mathbf{d}_5^z + \mathbf{d}_4^{z\Sigma} \\ \mathbf{d}_6^z + \mathbf{d}_4^{z\Sigma} \end{bmatrix} \tag{2.271}$$

Observe that, again, each body receives recursively the contributions with the $\mathbf{d}_i^z$ terms from the parent body in the kinematic chain.

With the help of the recursive kinematic expressions defined before, the mass matrix of the system takes the form:

$$\mathbf{M}^d = \mathbf{R}^{z\mathrm{T}}\mathbf{M}^z\mathbf{R}^z =$$
$$\begin{bmatrix} \mathbf{b}_1^{z\mathrm{T}}\mathbf{M}_1^{z\Sigma}\mathbf{b}_1^z & \mathbf{b}_1^{z\mathrm{T}}\mathbf{M}_2^{z\Sigma}\mathbf{b}_2^z & \mathbf{b}_1^{z\mathrm{T}}\mathbf{M}_3^z\mathbf{b}_3^z & \mathbf{b}_1^{z\mathrm{T}}\mathbf{M}_4^{z\Sigma}\mathbf{b}_4^z & \mathbf{b}_1^{z\mathrm{T}}\mathbf{M}_5^z\mathbf{b}_5^z & \mathbf{b}_1^{z\mathrm{T}}\mathbf{M}_6^z\mathbf{b}_6^z \\ \mathbf{b}_2^{z\mathrm{T}}\mathbf{M}_2^{z\Sigma}\mathbf{b}_1^z & \mathbf{b}_2^{z\mathrm{T}}\mathbf{M}_2^{z\Sigma}\mathbf{b}_2^z & \mathbf{b}_2^{z\mathrm{T}}\mathbf{M}_3^z\mathbf{b}_3^z & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{b}_3^{z\mathrm{T}}\mathbf{M}_3^z\mathbf{b}_1^z & \mathbf{b}_3^{z\mathrm{T}}\mathbf{M}_3^z\mathbf{b}_2^z & \mathbf{b}_3^{z\mathrm{T}}\mathbf{M}_3^z\mathbf{b}_3^z & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{b}_4^{z\mathrm{T}}\mathbf{M}_4^{z\Sigma}\mathbf{b}_1^z & \mathbf{0} & \mathbf{0} & \mathbf{b}_4^{z\mathrm{T}}\mathbf{M}_4^{z\Sigma}\mathbf{b}_4^z & \mathbf{b}_4^{z\mathrm{T}}\mathbf{M}_5^z\mathbf{b}_5^z & \mathbf{b}_4^{z\mathrm{T}}\mathbf{M}_6^z\mathbf{b}_6^z \\ \mathbf{b}_5^{z\mathrm{T}}\mathbf{M}_5^z\mathbf{b}_1^z & \mathbf{0} & \mathbf{0} & \mathbf{b}_5^{z\mathrm{T}}\mathbf{M}_5^z\mathbf{b}_4^z & \mathbf{b}_5^{z\mathrm{T}}\mathbf{M}_5^z\mathbf{b}_5^z & \mathbf{0} \\ \mathbf{b}_6^{z\mathrm{T}}\mathbf{M}_6^z\mathbf{b}_1^z & \mathbf{0} & \mathbf{0} & \mathbf{b}_6^{z\mathrm{T}}\mathbf{M}_6^z\mathbf{b}_4^z & \mathbf{0} & \mathbf{b}_6^{z\mathrm{T}}\mathbf{M}_6^z\mathbf{b}_6^z \end{bmatrix} \tag{2.272a}$$

$$\mathbf{M}_4^{z\Sigma} = \mathbf{M}_4^z + \mathbf{M}_5^z + \mathbf{M}_6^z \tag{2.272b}$$

$$\mathbf{M}_2^{z\Sigma} = \mathbf{M}_2^z + \mathbf{M}_3^z \tag{2.272c}$$

$$\mathbf{M}_1^{z\Sigma} = \mathbf{M}_1^z + \mathbf{M}_2^{z\Sigma} + \mathbf{M}_4^{z\Sigma} \tag{2.272d}$$

and the vector of generalized forces,

$$\mathbf{Q}^d = \mathbf{R}^{z\mathrm{T}}\left(\mathbf{Q}^z - \mathbf{M}^z\dot{\mathbf{R}}^z\dot{\mathbf{z}}\right) = \begin{bmatrix} \mathbf{b}_1^{z\mathrm{T}}\mathbf{Q}_1^{z\Sigma} \\ \mathbf{b}_2^{z\mathrm{T}}\mathbf{Q}_2^{z\Sigma} \\ \mathbf{b}_3^{z\mathrm{T}}\mathbf{Q}_3^{z\Sigma} \\ \mathbf{b}_4^{z\mathrm{T}}\mathbf{Q}_4^{z\Sigma} \\ \mathbf{b}_5^{z\mathrm{T}}\mathbf{Q}_5^{z\Sigma} \\ \mathbf{b}_6^{z\mathrm{T}}\mathbf{Q}_6^{z\Sigma} \end{bmatrix} \tag{2.273}$$

$$\mathbf{Q}_6^{z\Sigma} = \mathbf{Q}_6^z - \mathbf{M}_6^z\left(\dot{\mathbf{R}}^z\dot{\mathbf{z}}\right)_6 \tag{2.274}$$

$$\mathbf{Q}_5^{z\Sigma} = \mathbf{Q}_5^z - \mathbf{M}_5^z\left(\dot{\mathbf{R}}^z\dot{\mathbf{z}}\right)_5 \tag{2.275}$$

$$\mathbf{Q}_4^{z\Sigma} = \mathbf{Q}_4^z - \mathbf{M}_4^z\left(\dot{\mathbf{R}}^z\dot{\mathbf{z}}\right)_4 + \mathbf{Q}_5^{z\Sigma} + \mathbf{Q}_6^{z\Sigma} \tag{2.276}$$

$$\mathbf{Q}_3^{z\Sigma} = \mathbf{Q}_3^z - \mathbf{M}_3^z\left(\dot{\mathbf{R}}^z\dot{\mathbf{z}}\right)_3 \tag{2.277}$$

$$\mathbf{Q}_2^{z\Sigma} = \mathbf{Q}_2^z - \mathbf{M}_2^z\left(\dot{\mathbf{R}}^z\dot{\mathbf{z}}\right)_2 + \mathbf{Q}_3^{z\Sigma} \tag{2.278}$$

$$\mathbf{Q}_1^{z\Sigma} = \mathbf{Q}_1^z - \mathbf{M}_1^z\left(\dot{\mathbf{R}}^z\dot{\mathbf{z}}\right)_1 + \mathbf{Q}_2^{z\Sigma} + \mathbf{Q}_4^{z\Sigma} \tag{2.279}$$

In the fully-recursive approach, the same expressions and accumulations introduced in 2.4.2.1 can be used when the origin of coordinates is selected as reference point, but with two main changes: $\mathbf{B}_i^z$ becomes the identity, so it can be eliminated from the accumulative expressions; and (2.260) and (2.261) have to be used as the expressions of the elemental mass matrix and generalized forces vector for each body. This fully-recursive formulation is similar to the one identified as RTdyn2 by Jimenez in [45], but due to the particular consideration of forces directly assembled on relative coordinates, and for the sake of clarity, this formulation is here referred to as fully-recursive RTDyn1 or RTDyn1-FR.

This selection of reference points seems to lead to the simplest accumulation possible, but there is one important drawback, as it was commented before: as the mechanisms move away from the origin, recursive relations such as $\mathbf{b}_i^v$ are worse numerically conditioned. For instance, let us recall the expressions of $\mathbf{b}_i^v$ for the revolute joint for an arbitrary reference point (2.133a):

$$\mathbf{b}_i^v = \begin{bmatrix} \mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) \\ \mathbf{w}_j \end{bmatrix} \tag{2.280}$$

Observe that if the reference point is rigidly attached to the body, the term $\mathbf{r}_i - \mathbf{r}_j$ is equivalent to a vector with a fixed norm, and any change in the mechanism only modifies its orientation. However, if the reference point $\mathbf{r}_i$ is not rigidly fixed to body $i$, then the 2-norm of the vector $\mathbf{r}_i - \mathbf{r}_j$ varies along with the orientation, being its magnitude directly affected by the mechanism motion. Since $\mathbf{b}_i^v$ is one of the basic pieces required to accumulate masses and forces and to compute the forward kinematics of open-loop systems, small numerical errors could lead to important problems during the simulation and on a sensitivity analysis.

It is important to remark that both RTdyn0 and RTdyn1 formulations theoretically lead to the exact same equations, so it can be deduced that there is an unique analytical set of dynamic expressions that could be reached without explicitly involving any reference point. In the implementation in MBSLIM, the reference points have been eliminated from some analytical expressions, but for other terms, their presence is convenient in order to simplify some calculations, and their elimination could lead to an entangled code which could result more inefficient.

# Chapter 3

# Topological formulations for the dynamics of closed-loop systems

The description of a multibody system by means of a joint coordinate model usually leads to the minimum possible number of coordinates. In a joint coordinate model, constraints of relative motion between bodies are internally enforced by means of the kinematic recursive relations imposed by each joint. The presence of this recursive kinematics encompasses a minimization of the number of constraints required in those models, although there are multiple cases in which constraints have to be considered. The most obvious case is referred to closed-loop systems. In that case, the coordinates of the kinematic chain are no longer independent, but they have additional constraints of motion. This problem is generally addressed by means of cutting a joint and imposing of a loop-closure constraint. Depending on the joint being eliminated, the set of constraint equations required varies.

Another issue that entails the fulfillment of a constraint is the particular modeling of spherical and floating joints by means of Euler parameters. As commented in section 2.1.5, this non-minimal parameterization of rotations is subjected to a normalization constraint.

Besides, relative coordinate models could be subjected to driving constraints or kinematic relations not described by a joint. It is also possible to include constraints stemming from the combination of a kinematic or dynamic multibody problem with other of different nature.

In order to tackle any multibody system in terms of a joint coordinate model independently of its topology in a general manner, the kinematics and dynamics of constrained joint coordinate models are studied in this chapter, which is structured as follows. In section 3.1, a method for the evaluation of the kinematics of closed-loop multibody systems modeled in relative coordinates is described, including the case of any set of natural coordinates as degrees of freedom. In section 3.2, the Matrix R method [117] is applied to the the semi-recursive formulation for open-loop systems presented in chapter 2. As a result, a formulation for closed-loop systems in independent coordinates is achieved, which has been deeply described and tested in past years [54, 55, 118]. As primary novelty, this formulation is extended for supporting a definition of degrees of freedom out from the vector of dependent joint coordinates, which allows, for example, to define degrees of freedom in natural coordinates, instead of joint coordinates. Finally, in section 3.3, the Augmented Lagrangian index-3 formulation with velocity and acceleration projections is combined with the semi-recursive formulation for open-loop systems presented in chapter 2, generating the so called RTdyn0-ALI3-P and RTdyn1-ALI3-P formulations for closed-loop systems. As a result of the implementation and generalization of these formulations, a paper was published [60].

## 3.1 Kinematics for non-minimal relative coordinates

"The kinematic analysis of a mechanical system concerns the motion of the system independent of forces that produce the motion", [114]. In this sense, the kinematic analysis encompasses the solution of the initial position problem, the finite displacement problem along with the kinematic velocity and acceleration analysis. These problems involve purely geometrical considerations without including the effect of inertia or the forces applied to the mechanism.

Any multibody system modeled in natural (or reference point) coordinates entails a set of constraints referred to geometrical considerations, relating the positions, velocities or accelerations of the points and vectors of the model. On the contrary, not all the relative coordinate models require a set of constraints to be fully defined, as it occurs in open-loop systems with minimal coordinates. In that case, if the relative joint coordinates are selected as the set of degrees of freedom of the system, the kinematics of every body of each kinematic chain can be fully determined from the kinematics of the preceding body in the kinematic chain by relatively simple and systematic operations with the relative coordinates vector of the system.

However, most of the mechanisms studied and modeled in Mechanical Engineering have, in one way or another, constraints limiting the motion and complementing the definition of the multibody model. The presence of constraint equations into a multibody model indicates that the number of coordinates being used is larger than the number of degrees of freedom, thus the combination of the kinematic-joint recursive relations and the constraint equations must be addressed.

The kinematic problems previously mentioned, i.e. the initial position problem, finite displacements problem, velocity analysis and kinematic acceleration analysis, involve at least four phases or steps in what concerns to joint coordinate models:

1. **Assessment of points and vectors from relative coordinates:** the kinematic constraints are usually expressed in terms of positions, velocities or accelerations of points and vectors, and therefore, it is mandatory to update these values prior to any constraint evaluation. Furthermore, points and vectors are as well needed in the recursive relations introduced in 2.2, which are essential for the kinematics evaluation of the each body.

2. **Recursive kinematic relations:** in the current kinematic development, the recursive relations compose the bulk of the kinematics evaluation. These recur-

sive terms describe the relative motion between bodies, and therefore, they are present in every accumulation, assembly and derivative referred to the whole mechanism. In kinematic analyses, these terms are involved in the recursive kinematics of the open-loop system but also in the derivatives of the constraint vector.

3. **Constraints vector and derivatives:** once assessed positions, velocities and/or accelerations of the points and vectors defining the constraints, the constraint vector evaluation is straightforward. On the contrary, constraint derivatives are not as easy to obtain, and a new set of derivatives of positions, velocities and accelerations of points and vectors with respect to relative coordinates have to be resorted to. The derivatives needed in each kinematic problem will be specified in the following sections, but their calculation will be explained in detail in chapter 4.

4. **Composition of the problem and solution:** each constrained kinematic problem requires the constraints vector at position, velocity or acceleration levels along with some partial derivatives, depending on the kinematic problem regarded. In general, since the constrained kinematic problems are expressed in terms of joint-coordinates, only these are updated and a new evaluation of the first step must be executed for points and vectors assessment. Depending on the type of problem, an iterative scheme could be needed, and the validity of the solution will be related to the existence of an error lower than a given tolerance.

In the following lines, the three main kinematic problems are briefly described (the finite displacements problem is assimilated to the initial position problem). The interest of the description of constrained kinematic problems in joint coordinate models is double: first, it manifests that joint coordinate modeling is general enough to pose and solve any dynamic or kinematic problem; and second, some of the listed kinematic problems are required by some dynamic formulations such as the topological semi-recursive Matrix R of section 3.2. The three constrained kinematic problems have been implemented in the MBSLIM general purpose multibody library as general kinematic formulations.

## 3.1.1 Initial position problem

The initial position problem seeks to determine the relative coordinates at position level from a given set of values of the DoF of the mechanism, or which is the same, to establish the vector of dependent coordinates that satisfies the system of nonlinear constraint equations for a given set of DoF. It must be noted that the scheme of solution presented here and implemented in MBSLIM is the method described in [117], but particularized to relative coordinates.

The objective of the initial position problem is to nullify the vector of constraints $\mathbf{\Phi} \in \mathbb{R}^m$ (being $m$ the number of constraint equations) for a given set of values of the independent coordinates $\mathbf{z}^i$:

$$\mathbf{\Phi}\left(\mathbf{q},\mathbf{z},t\right)=\mathbf{0} \tag{3.1}$$

where $\mathbf{q}$ is the vector of natural coordinates, including points and vectors of the model, $\mathbf{z}$ is the vector of dependent relative coordinates and $t$ is the time. Expression (3.1) represents a system of nonlinear equations, which in general cannot be solved analytically due to the complexity of the equations, and has to be computed numerically. One of the most resorted approaches to solve these type of equations is the Newton-Raphson method, widely spread in multitude of nonlinear problems due to its simplicity and its quadratic convergence in the neighborhood of the solution. The Newton-Raphson method is based on the expansion of (3.1) in a Taylor series around a given initial approximated solution $\mathbf{z}^0$:

$$\mathbf{\Phi}\left(\mathbf{q}\left(\mathbf{z}\right),\mathbf{z},t\right)\cong\mathbf{\Phi}\left(\mathbf{q}^0,\mathbf{z}^0\right)+\mathbf{\Phi}_{\hat{\mathbf{z}}}\left(\mathbf{q}^0,\mathbf{z}^0\right)\left(\mathbf{z}-\mathbf{z}^0\right)=\mathbf{0} \tag{3.2}$$

where $\mathbf{\Phi}_{\hat{\mathbf{z}}}\in\mathbb{R}^{m\times n}$ is the Jacobian matrix for the constraint equations:

$$\mathbf{\Phi}_{\hat{\mathbf{z}}}=\frac{\partial\mathbf{\Phi}}{\partial\mathbf{q}}\frac{\partial\mathbf{q}}{\partial\mathbf{z}}+\frac{\partial\mathbf{\Phi}}{\partial\mathbf{z}}=\mathbf{\Phi}_{\mathbf{q}}\mathbf{q}_{\mathbf{z}}+\mathbf{\Phi}_{\mathbf{z}} \tag{3.3}$$

From (3.3) it can be observed that $\mathbf{\Phi}_{\hat{\mathbf{z}}}$ is not exactly a partial derivative, and for this reason the subscript $\hat{\mathbf{z}}$ is used. This notation will be extended in the subsequent sensitivity chapters.

Equation (3.2) can be reformulated as:

$$\mathbf{\Phi}_{\hat{\mathbf{z}}}\left(\mathbf{q}^j,\mathbf{z}^j\right)\Delta\mathbf{z}^{j+1}=-\mathbf{\Phi}\left(\mathbf{q}^j,\mathbf{z}^j\right) \tag{3.4}$$

being $\mathbf{z}^{j+1}$ a better approximation of the solution of (3.1) than $\mathbf{z}^j$, with $j$ and $j+1$ as the iteration numbers[1]. Behold that equation (3.13) cannot be solved because it is rank deficient in the number of degrees of freedom of the system. Moreover, observe that no reference has been made to the imposition of the values of the degrees of freedom yet. For this sake, a new constant matrix $\mathbf{B}\in\mathbb{R}^{d\times n}$ composed of "1"s and "0"s is considered[2] according to [117], being $d$ the number of DoF of the multibody system. This matrix is constant in time, straightforward to calculate with a given set of independent coordinates, and satisfies the following relations:

$$\mathbf{z}^i=\mathbf{B}\mathbf{z} \tag{3.5a}$$

$$\dot{\mathbf{z}}^i=\mathbf{B}\dot{\mathbf{z}} \tag{3.5b}$$

$$\ddot{\mathbf{z}}^i=\mathbf{B}\ddot{\mathbf{z}} \tag{3.5c}$$

With these assumptions, the initial position problem can be finally expressed as:

$$\begin{bmatrix}\mathbf{\Phi}_{\hat{\mathbf{z}}}^j\\ \mathbf{B}\end{bmatrix}\Delta\mathbf{z}^{j+1}=\begin{bmatrix}-\mathbf{\Phi}^j\\ \mathbf{0}\end{bmatrix} \tag{3.6a}$$

$$\mathbf{z}^{j+1}=\mathbf{z}^j+\Delta\mathbf{z}^{j+1} \tag{3.6b}$$

[1]Sometimes, the Newton-Raphson scheme could provide a solution for one iteration that is further from the real solution than the result of the previous iteration. However, these "steps back" do not compromise the convergence of the method.

[2]Do not confuse with $\mathbf{B}_i^v$, an elemental term of the recursive accumulation.

In the MSBLIM implementation, in order to include the possibility of redundant constraints, a least-squares problem is solved. Both sides of (3.6a) are pre-multiplied by the transpose of the leading matrix in (3.6a), generating a non-singular symmetric system matrix:

$$\begin{bmatrix}\mathbf{\Phi}_{\hat{\mathbf{z}}}^{j\mathrm{T}} & \mathbf{B}^{\mathrm{T}}\end{bmatrix}\begin{bmatrix}\mathbf{\Phi}_{\hat{\mathbf{z}}}^{j} \\ \mathbf{B}\end{bmatrix}\Delta\mathbf{z}^{j+1} = \begin{bmatrix}\mathbf{\Phi}_{\hat{\mathbf{z}}}^{j\mathrm{T}} & \mathbf{B}^{\mathrm{T}}\end{bmatrix}\begin{bmatrix}-\mathbf{\Phi}^{j} \\ \mathbf{0}\end{bmatrix} \tag{3.7}$$

and simplifying:

$$\left(\mathbf{\Phi}_{\hat{\mathbf{z}}}^{j\mathrm{T}}\mathbf{\Phi}_{\hat{\mathbf{z}}}^{j} + \mathbf{B}^{\mathrm{T}}\mathbf{B}\right)\Delta\mathbf{z}^{j+1} = -\mathbf{\Phi}_{\hat{\mathbf{z}}}^{j\mathrm{T}}\mathbf{\Phi}^{j} \tag{3.8}$$

The finite displacement problem is very similar to the initial position problem needed to start the dynamics satisfying the constraint equations. In fact, the system of nonlinear equations is identical owing to that both problems are referred to the fulfillment of a series of constraints in positions with a given set of DoF.

The philosophical difference between the finite displacement problem and the initial position problem manifests in kinematic simulations: the initial position problem is regarded as an instant problem, whose purpose is to determine the initial values of the dependent coordinates, while the finite displacement analysis seeks to determine the position of the multibody system at each time step for a given time interval. However, this distinction does not involve any procedure difference, and in fact, both problems are addressed with the same algorithm in MBSLIM.

### 3.1.2 Kinematic velocity analysis

The velocity problem aims to obtain the set of velocities $\dot{\mathbf{z}}$ such that:

$$\dot{\mathbf{\Phi}}\left(\mathbf{q}\left(\mathbf{z}\right), \dot{\mathbf{q}}\left(\mathbf{z}, \dot{\mathbf{z}}\right), \mathbf{z}, \dot{\mathbf{z}}, t\right) = \mathbf{\Phi}_{\hat{\mathbf{z}}}\dot{\mathbf{z}} + \mathbf{\Phi}_{t} = \mathbf{0} \tag{3.9}$$

Differentiating (3.1) with respect to time and completing with equation (3.5b), velocity equations can be rewritten as:

$$\begin{bmatrix}\mathbf{\Phi}_{\hat{\mathbf{z}}} \\ \mathbf{B}\end{bmatrix}\dot{\mathbf{z}} = \begin{bmatrix}-\mathbf{\Phi}_{t} \\ \dot{\mathbf{d}}\end{bmatrix} \equiv \begin{bmatrix}\mathbf{b} \\ \dot{\mathbf{d}}\end{bmatrix} \tag{3.10}$$

with $\dot{\mathbf{d}}$ representing the desired values of the DoF at velocity level, and where the notation $\mathbf{b} \equiv -\mathbf{\Phi}_{t}$ is introduced in accordance with the classical compact notation of this problem[3], as presented in [117]. This notation will be reused also in the dynamic formulations developed in subsequent sections. On this occasion, the velocity problem is solved by means of a linear system of equations and, unlike the position problem, does not need to be iterated.

[3]Do not confuse with $\mathbf{b}_{i}^{v}$, an elemental term of the recursive accumulation

### 3.1.3 Kinematic acceleration analysis

In the acceleration problem, the set of constraints in accelerations have to be fulfilled by $\ddot{\mathbf{z}}$:

$$\ddot{\mathbf{\Phi}}\left(\mathbf{q}\left(\mathbf{z}\right),\dot{\mathbf{q}}\left(\mathbf{z},\dot{\mathbf{z}}\right),\ddot{\mathbf{q}}\left(\mathbf{z},\dot{\mathbf{z}},\ddot{\mathbf{z}}\right),\mathbf{z},\dot{\mathbf{z}},\ddot{\mathbf{z}},t\right)=\mathbf{0} \tag{3.11}$$

Similarly to section 3.1.2, the equations of the acceleration analysis can be found differentiating (3.1) twice with respect to time and completing with equation (3.5c):

$$\begin{bmatrix}\mathbf{\Phi}_{\hat{\mathbf{z}}}\\ \mathbf{B}\end{bmatrix}\ddot{\mathbf{z}}=\begin{bmatrix}-\dot{\mathbf{\Phi}}_t-\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}\dot{\mathbf{z}}\\ \ddot{\mathbf{d}}\end{bmatrix}\equiv\begin{bmatrix}\mathbf{c}\\ \ddot{\mathbf{d}}\end{bmatrix} \tag{3.12}$$

with $\ddot{\mathbf{d}}$ the desired values of the DoF accelerations.

Here again, $\mathbf{c}\equiv-\dot{\mathbf{\Phi}}_t-\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}\dot{\mathbf{z}}$ is introduced to compact notation in this expression and in further dynamic developments. It is important to note that, though no new derivatives appeared in the velocity analysis with respect to the position problems, in the acceleration problem there is a new term that involve derivatives with respect to $\mathbf{z}$, which is $\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}$:

$$\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}=\dot{\mathbf{\Phi}}_{\mathbf{q}}\mathbf{q}_{\mathbf{z}}+\mathbf{\Phi}_{\mathbf{q}}\dot{\mathbf{q}}_{\mathbf{z}}+\dot{\mathbf{\Phi}}_{\mathbf{z}} \tag{3.13}$$

wherein the relation $\mathbf{\Phi}_{\mathbf{q}}=\dot{\mathbf{\Phi}}_{\dot{\mathbf{q}}}$ has been used.

Thus, $\mathbf{c}$ can be rewritten as:

$$\mathbf{c}=-\dot{\mathbf{\Phi}}_t-\left(\dot{\mathbf{\Phi}}_{\mathbf{q}}\mathbf{q}_{\mathbf{z}}+\mathbf{\Phi}_{\mathbf{q}}\dot{\mathbf{q}}_{\mathbf{z}}+\dot{\mathbf{\Phi}}_{\mathbf{z}}\right)\dot{\mathbf{z}} \tag{3.14}$$

In general, $\dot{\mathbf{\Phi}}$ is expressed in terms of positions and velocities of points, vectors, angles, distances and/or the time, and consequently, the assessment of the derivatives $\dot{\mathbf{\Phi}}_{\mathbf{q}}$, $\dot{\mathbf{\Phi}}_{\mathbf{z}}$ and $\dot{\mathbf{\Phi}}_t$ (all with respect to the explicit dependencies of each constraint) are straightforward to obtain. On the contrary, the derivatives of the natural coordinates $\dot{\mathbf{q}}_{\mathbf{z}}$ and $\mathbf{q}_{\mathbf{z}}$ depend on the topology of the mechanism and their evaluation is more challenging, as it will be explained in sections 3.6 and 3.7.

### 3.1.4 Topological kinematics with natural coordinates as degrees of freedom

In closed-loop systems, the degrees of freedom of a mechanism can be identified more directly with a set of Cartesian coordinates rather than joint coordinates. In order to support natural coordinates as DoF (a requirement in MBSLIM models definition), a reformulation of the previous kinematic problems should be faced. For simplicity, hereinafter the dependent relative coordinates are denoted as $\mathbf{z}$ and the independent coordinates (natural or relative) are identified as $\mathbf{z}^i$.

For the kinematic analyses, the constraints vector and its Jacobian matrix have to be evaluated, but also the values of the DoF must be enforced. Following the developments of section 3.1.1, the imposition of DoF can be addressed by means of a new matrix $\mathbf{B}$:

$$\begin{bmatrix}\mathbf{\Phi}_{\hat{\mathbf{z}}}^{\{n\}}\\ \mathbf{B}^{\{n\}}\end{bmatrix}\Delta\mathbf{z}^{\{n+1\}}=\begin{bmatrix}-\mathbf{\Phi}^{\{n\}}\\ \mathbf{d}-\left(\mathbf{z}^i\right)^{\{n\}}\end{bmatrix} \tag{3.15}$$

in which $\Delta\mathbf{z}^{\{n+1\}}$ is the increment in the joint coordinates, $\mathbf{d}$ represents the set of desired positions of the DoF, $(\mathbf{z}^i)^{\{n\}}$ denotes the current value of the DoF at iteration $n$ and $\mathbf{B}^{\{n\}}$ is a matrix describing the relation between joint coordinates and DoF given by (2.191).

If the degrees of freedom are selected from the relative coordinates vector $\mathbf{z}$, the matrix $\mathbf{B}$ will be constant and composed by "1"s and "0"s. Behold that equation (2.191) makes possible to use degrees of freedom out from the relative coordinates vector, but it will result in a more complex matrix $\mathbf{B}$. Knowing that $\mathbf{z}$ defines completely the kinematics of a mechanism, and that any kinematic magnitude can be obtained as a function of these coordinates (applying the theorem of the implicit function), it can be said that:

$$\mathbf{z}^i = f(\mathbf{z}) \tag{3.16}$$

Therefore, the term $\mathbf{B}$ can be generalized as:

$$\mathbf{B} = \frac{\partial f(\mathbf{z})}{\partial \mathbf{z}} \tag{3.17}$$

With this approach, any kinematic magnitude can be used as degree of freedom as long as the matrix $\begin{bmatrix} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \\ \mathbf{B} \end{bmatrix}$ is not singular which is guaranteed if a proper selection of degrees of freedom is made, from the kinematic point of view.

As an example, let us consider the coordinates of a point $\mathbf{r}_j$ as 3 degrees of freedom. In this case, the term $\mathbf{B}$ would be:

$$\mathbf{B} = \frac{\partial \mathbf{r}_j}{\partial \mathbf{z}} \tag{3.18}$$

which can be computed using expressions of section 3.6.

These expressions could be extended to kinematic problems in velocities and accelerations using the same definition of the $\mathbf{B}$ matrix, but in those problems the derivatives of this matrix with respect to time must be considered too.

$$\begin{bmatrix} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \\ \mathbf{B} \end{bmatrix} \dot{\mathbf{z}} = \begin{bmatrix} \mathbf{b} \\ \dot{\mathbf{d}} \end{bmatrix}, \tag{3.19a}$$

$$\begin{bmatrix} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \\ \mathbf{B} \end{bmatrix} \ddot{\mathbf{z}} = \begin{bmatrix} \mathbf{c} \\ \ddot{\mathbf{d}} - \dot{\mathbf{B}}\dot{\mathbf{z}} \end{bmatrix} \tag{3.19b}$$

Note that expressions (3.19a) and (3.19b) are equivalent to (3.10) and (3.12) if the set of DoF is contained in $\mathbf{z}$.

## 3.2 Semi-recursive Matrix R formulations

The formulation presented in this section was originally introduced by García de Jalón and Bayo in [16] and later applied to semi-recursive methods in [45], [46] and more recently in [54], [55] and [118]. As long as the method is profusely described in those works, only the main structure of the formulation is outlined. A new notation is employed here so as to avoid possible misunderstandings between the **R** matrix of the semi-recursive method and the projection matrix $\mathbf{R}^{\boldsymbol{\Phi}}$ of this formulation.

In this approach, a second velocity transformation (Matrix R transformation) is carried out in order to remove some dependent coordinates from the full set of dependent ones. Let us consider a multibody system modeled with $n$ relative coordinates subjected to $m$ constraints and with $d$ DoF. A vector of independent coordinates $\mathbf{z}^i \in \mathbb{R}^d$ can be selected such as the dependent velocities, accelerations and virtual displacements can be expressed in terms of the independent ones by means of matrices $\mathbf{R}^{\boldsymbol{\Phi}} \in \mathbb{R}^{n\times d}$ and $\mathbf{S}^{\boldsymbol{\Phi}} \in \mathbb{R}^{n\times m}$:

$$\dot{\mathbf{z}} = \mathbf{R}^{\boldsymbol{\Phi}}\dot{\mathbf{z}}^i + \mathbf{S}^{\boldsymbol{\Phi}}\mathbf{b} \tag{3.20a}$$

$$\ddot{\mathbf{z}} = \mathbf{R}^{\boldsymbol{\Phi}}\ddot{\mathbf{z}}^i + \mathbf{S}^{\boldsymbol{\Phi}}\mathbf{c} \tag{3.20b}$$

$$\dot{\mathbf{z}}^* = \mathbf{R}^{\boldsymbol{\Phi}}\dot{\mathbf{z}}^{*i} \tag{3.20c}$$

being **b** and **c** the terms related to the temporal constraints derivatives described in sections 3.1.2 and 3.1.3 respectively.

Behold that in the present formulation $\mathbf{z}^i$ represents a set of differential variables with a different meaning than **d** in kinematic problems.

The expressions of the semi-recursive Matrix R formulation can be derived from the virtual power principle applied to a multibody system. Let us recall here the open-loop equation (2.207):

$$\dot{\mathbf{z}}^{*\mathrm{T}}\left[\left(\mathbf{R}^{v\mathrm{T}}\mathbf{M}^v\mathbf{R}^v\right)\ddot{\mathbf{z}} - \mathbf{R}^{v\mathrm{T}}\left(\mathbf{Q}^v - \mathbf{M}^v\dot{\mathbf{R}}^v\dot{\mathbf{z}}\right)\right] = 0 \tag{3.21}$$

which can be rewritten using the notation $\mathbf{M}^d$ and $\mathbf{Q}^d$ introduced in (2.208b) and (2.208c) according to (2.208a) as:

$$\dot{\mathbf{z}}^{*\mathrm{T}}\left[\mathbf{M}^d\ddot{\mathbf{z}} - \mathbf{Q}^d\right] = 0 \tag{3.22}$$

Using (3.20) in (3.22), and taking into account that the virtual velocities $\dot{\mathbf{z}}^{*i}$ are independent, one obtains

$$\left(\mathbf{R}^{\boldsymbol{\Phi}\mathrm{T}}\mathbf{M}^d\mathbf{R}^{\boldsymbol{\Phi}}\right)\ddot{\mathbf{z}}^i = \mathbf{R}^{\boldsymbol{\Phi}\mathrm{T}}\left(\mathbf{Q}^d - \mathbf{M}^d\mathbf{S}^{\boldsymbol{\Phi}}\mathbf{c}\right) \tag{3.23}$$

which are the general Matrix R equations for closed loops or non-minimal joint coordinates.

With an appropriate selection of degrees of freedom, matrices $\mathbf{R}^{\boldsymbol{\Phi}}$ and $\mathbf{S}^{\boldsymbol{\Phi}}$ can be calculated as a part of the inverse of the leading matrix of any of the kinematic problems previously introduced:

$$\begin{bmatrix} \mathbf{S}^{\boldsymbol{\Phi}} & \mathbf{R}^{\boldsymbol{\Phi}} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \\ \mathbf{B} \end{bmatrix} = \mathbf{I}_n \tag{3.24}$$

For the general case of redundant constraints, matrices $\mathbf{R}^{\boldsymbol{\Phi}}$ and $\mathbf{S}^{\boldsymbol{\Phi}}$ can be obtained by means of a least-squares problem:

$$\left[\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}} + \mathbf{B}^{\mathrm{T}}\mathbf{B}\right]\left[\mathbf{S}^{\boldsymbol{\Phi}} \quad \mathbf{R}^{\boldsymbol{\Phi}}\right] = \left[\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \quad \mathbf{B}^{\mathrm{T}}\right] \tag{3.25}$$

### 3.2.1 Non-constant B matrix

In the previous Matrix R developments, matrix **B** was assumed to be constant, which indicates that the vector of independent coordinates is contained in the dependents one. As it was introduced in section 3.1.4, the selection of the degrees of freedom of a mechanism is one of the most entangled parts in a kinematics analysis, and this complexity is translated to independent-coordinates forward dynamics formulations such as the Matrix R formulation. As commented above, in some cases a degree of freedom of a mechanism is better described by a coordinate of a point or a vector than by a joint coordinate, especially in closed-loop systems. For this reason, a generalization of the equations of the previous section is devised, bearing in mind that the use of natural coordinates as degrees of freedom may difficult and slow down the computation of the dynamic problem.

If matrix **B** is considered as variable and dependent on the kinematics of the system, the calculation of the acceleration of the joint coordinates with the matrices $\mathbf{R}^{\boldsymbol{\Phi}}$ and $\mathbf{S}^{\boldsymbol{\Phi}}$ requires additional terms. Considering the equations of the kinematics in accelerations for a non-constant **B** matrix (3.19b) and the expressions of the acceleration of the dependent coordinates in terms of $\mathbf{R}^{\boldsymbol{\Phi}}$ and $\mathbf{S}^{\boldsymbol{\Phi}}$ (3.20b), the following relation is reached:

$$\ddot{\mathbf{z}} = \mathbf{R}^{\boldsymbol{\Phi}}\left(\ddot{\mathbf{z}}^{i} - \dot{\mathbf{B}}\dot{\mathbf{z}}\right) + \mathbf{S}^{\boldsymbol{\Phi}}\mathbf{c} \tag{3.26}$$

Moreover, the non-constant **B** matrix induces changes in the general dynamic expressions (3.23):

$$\left(\mathbf{R}^{\boldsymbol{\Phi}\mathrm{T}}\mathbf{M}^{d}\mathbf{R}^{\boldsymbol{\Phi}}\right)\ddot{\mathbf{z}}^{i} = \mathbf{R}^{\boldsymbol{\Phi}\mathrm{T}}\left(\mathbf{Q}^{d} - \mathbf{M}^{d}\left(\mathbf{S}^{\boldsymbol{\Phi}}\mathbf{c} - \mathbf{R}^{\boldsymbol{\Phi}}\dot{\mathbf{B}}\dot{\mathbf{z}}\right)\right) \tag{3.27}$$

Comparing (3.23) and (3.27), it can be deduced that the first expression is preferable and computationally more efficient, but the possibility of selecting the DoF from a wider number of variables makes the second approach especially interesting for the MBSLIM implementation, in which the multibody models as well as their DoF are described mainly in terms of points and vectors (natural coordinates). Behold also that (3.23) represents a particular case of the more general expression (3.27).

## 3.3 Semi-recursive ALI3-P formulations

The Augmented Lagrangian Index-3 formulation with velocity and acceleration projections (ALI3-P) constitutes a forward dynamics formulation which combines high accuracy at position, velocity and acceleration levels with a reduced computational time. From the seminal work by Bayo and Ledesma [119], the ALI3-P equations have been applied in the framework of global [53, 120] and topological formulations [54, 60, 120] ( [60] constitutes a result from the present work) in the field of rigid body dynamics. Moreover, these formalisms have been extended to flexible multibody systems [121–123], and its capabilities have been exploited for real time simulation [123, 124]. In this section, the application of the ALI3-P formalism to a semi-recursive accumulation scheme in the framework of joint coordinate models is studied as an update of the method presented in [120].

In ALI3-P formulations, the equations of motion are formulated with dependent coordinates, in contrast with the semi-recursive Matrix R formulation of section 3.2. This dependent-coordinate formulation eliminates the need of a selection of degrees of freedom along with the evaluation of the matrix $\mathbf{B}$, but it has also its shortcomings. Firstly, a set of Lagrange multipliers $\boldsymbol{\lambda} \in \mathbb{R}^m$ has to be added, increasing the number of variables of the problem. Secondly, the resulting expressions constitute a set of Differential Algebraic Equations (DAE) instead of the set of Ordinary Differential Equations (ODE) yielded by the semi-recursive Matrix R formulation.

The ALI3-P scheme is based upon the union of an index-3 augmented Lagrangian scheme (with imposition of constraints at position level) with one projection of state velocities onto the manifold $\dot{\boldsymbol{\Phi}} = \mathbf{0}$ and another acceleration projection onto $\ddot{\boldsymbol{\Phi}} = \mathbf{0}$. Even though the augmented Lagrangian part of the algorithm guarantees the fulfillment of the constraints at position level, if it is evaluated alone without any numerical dissipation, it will introduce increasing errors and instabilities due to the violation of the constraints at velocity and accelerations levels. It should be highlighted that the ALI3 equations of motion (without projections) are equivalent to the classical index-3 DAE, thus their instabilities are the same. The purpose of projections in this case is double since they limit the violation of the time derivatives of the constraints and, as a consequence, they stabilize the index-3 DAE, allowing long simulations.

The first step of this formulation is the solution of the augmented Lagrangian system in the form:

$$\mathbf{M}^d \ddot{\mathbf{z}}^* + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \left( \boldsymbol{\lambda}^{*\{i+1\}} + \boldsymbol{\alpha}\boldsymbol{\Phi} \right) = \mathbf{Q}^d \tag{3.28a}$$

$$\boldsymbol{\lambda}^{*\{i+1\}} = \boldsymbol{\lambda}^{*\{i\}} + \boldsymbol{\alpha}\boldsymbol{\Phi}^{\{i+1\}};\ i > 0 \tag{3.28b}$$

where $i = 0, 1, 2, ...$, $\boldsymbol{\alpha} \in \mathbb{R}^{m \times m}$ is a diagonal matrix that contains the penalty factors associated with the constraints, $i$ is the iteration index of the approximate Lagrange multipliers $\boldsymbol{\lambda}^* \in \mathbb{R}^m$, $\mathbf{M}^d \in \mathbb{R}^{n \times n}$ is the mass matrix referred to joint-coordinates, $\mathbf{Q}^d \in \mathbb{R}^n$ denotes the vector of generalized forces, $\boldsymbol{\Phi} \in \mathbb{R}^m$ is the constraint vector and $\boldsymbol{\Phi}_{\hat{\mathbf{z}}} \in \mathbb{R}^{m \times n}$ its Jacobian matrix.

The Lagrange multipliers of the augmented Lagrangian formulation converge to the values of the Lagrange multipliers of the original index-3 DAE for $i \to \infty$. However, as soon as the constraint violation is low enough, both sets of Lagrange multipliers become almost identical.

The initialization of the Lagrange multipliers at each time step can be faced with different schemes. One option is to start from null values, as indicated in [125]. Another option, which is the one regarded for the MBSLIM implementation, consists in using the Lagrange multiplier values of the previous time step to initiate the iteration in the current time step. This last update method has been described in [120], and as the authors declare, it usually offers fewer iterations than the former method. Despite that, this method has an important shortcoming related to redundant constraints. If there are redundant constraints, some of the Lagrange multipliers would be undetermined, and it was experimentally observed that the upgrading scheme could lead to increasing values of the Lagrange multipliers which could eventually entail a numerical ill conditioning of the problem. In order to limit the increase in the values of the Lagrange multipliers, the initialization of the Lagrange multipliers has been substituted by:

$$\boldsymbol{\lambda}_j^{*\{0\}} = \varpi \boldsymbol{\lambda}_{j-1}^{*} : \ \varpi < 1.0 \tag{3.29}$$

being $j$ the index of the time step, and $\varpi$ a scaling factor that delivers good results for values from 0.95 to 0.99[4].

The first combination of an augmented Lagrangian scheme with projection methods is attributed to Bayo and Ledesma in their celebrated work [119]. In this paper, the authors presented a different version of the augmented Lagrangian problem than the one revisited here, which involved a term composed of the weighted combination of the constrains vector at position, velocity and acceleration levels. Moreover, they introduced three different projections at position, velocity and acceleration levels.

Projection techniques are founded on a minimization problem of the form:

$$min_{\mathbf{x}} V = \frac{1}{2}\left(\mathbf{x}-\mathbf{x}^*\right)^{\mathrm{T}} \bar{\mathbf{P}} \left(\mathbf{x}-\mathbf{x}^*\right) \ s.\,t.\ \mathbf{g}\left(\mathbf{x},t\right) = \mathbf{0} \tag{3.30}$$

wherein $\mathbf{x}$ are the projected magnitudes, $\mathbf{x}^*$ the unprojected ones, $\bar{\mathbf{P}}$ a projection matrix and $\mathbf{g}$ a set of constraints which specify the manifold in which $\mathbf{x}^*$ is projected.

This type of problem can be tackled applying the augmented Lagrangian method for the combination of the objective function being minimized and the constraints to which the problem is subjected, leading to an unconstrained minimization problem with the form:

$$min_{\mathbf{x}} V^* = \frac{1}{2}\left(\mathbf{x}-\mathbf{x}^*\right)^{\mathrm{T}} \bar{\mathbf{P}} \left(\mathbf{x}-\mathbf{x}^*\right) + \frac{1}{2}\mathbf{g}^{\mathrm{T}}\alpha\mathbf{g} + \mathbf{g}^{\mathrm{T}}\boldsymbol{\tau} \tag{3.31}$$

in which $\mathbf{g}_{\mathbf{x}}$ is the Jacobian of $\mathbf{g}$ with respect to $\mathbf{x}$, $\alpha$ is a penalty factor and $\boldsymbol{\tau}$ is a set of Lagrange multipliers.

Now, differentiating (3.31) and equating to zero (first-order necessary condition), the following equation emerges:

$$\bar{\mathbf{P}}\left(\mathbf{x}-\mathbf{x}^*\right) + \mathbf{g}_{\mathbf{x}}^{\mathrm{T}}\alpha\mathbf{g} + \mathbf{g}_{\mathbf{x}}^{\mathrm{T}}\boldsymbol{\tau} = \mathbf{0} \tag{3.32}$$

which regarding the dependencies of $\mathbf{g}$, (3.32) represents in general a nonlinear system of equations. In the cases regarded in the ALI3-P formulation, the set of equations generated during velocity and acceleration projections are linear, which significantly reduces the computational effort devoted to that formulation stage.

For the sake of clarity, let us consider now velocity projections. For projections onto the manifold $\dot{\boldsymbol{\Phi}} = \mathbf{0}$ in joint coordinate models, the minimization problem becomes:

$$min_{\dot{\mathbf{z}}} V = \frac{1}{2}\left(\dot{\mathbf{z}}-\dot{\mathbf{z}}^*\right)^{\mathrm{T}} \bar{\mathbf{P}} \left(\dot{\mathbf{z}}-\dot{\mathbf{z}}^*\right) \ s.\,t.\ \dot{\boldsymbol{\Phi}}\left(\mathbf{z},\dot{\mathbf{z}},t\right) = \mathbf{0} \tag{3.33}$$

[4]Acknowledgments to Dr. Emilio Sanjurjo Maroño, which detected the issue and proposed the new updating method.

The application of the augmented Lagrangian method to (3.33) according to (3.32) yields:

$$\bar{\mathbf{P}}\left(\dot{\mathbf{z}}-\dot{\mathbf{z}}^{*}\right)+\mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\varsigma\boldsymbol{\alpha}\dot{\mathbf{\Phi}}+\mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\sigma}=\mathbf{0} \tag{3.34}$$

in which $\varsigma$ is a new penalty term associated to the penalty matrix $\boldsymbol{\alpha}$ of the augmented Lagrangian part of the ALI3-P formulation (introduced in equation (3.28)). The reason of adding a new penalty factor can be justified regarding that some terms can be directly reused from the augmented Lagrangian stage of the ALI3-P formulation.

As described in [119], there are different options to solve (3.34):

- Augmented Lagrangian method:

  The application of the augmented Lagrangian method to the equation (3.34) yields an iterative solution algorithm with the form:

$$\left(\bar{\mathbf{P}}+\varsigma\mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\alpha}\mathbf{\Phi}_{\hat{\mathbf{z}}}\right)\dot{\mathbf{z}}=\bar{\mathbf{P}}\dot{\mathbf{z}}^{*}-\mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\left(\boldsymbol{\sigma}^{\{i+1\}}+\varsigma\boldsymbol{\alpha}\mathbf{b}\right) \tag{3.35a}$$

$$\boldsymbol{\sigma}^{\{i+1\}}=\boldsymbol{\sigma}^{\{i\}}+\varsigma\boldsymbol{\alpha}\dot{\mathbf{\Phi}} \tag{3.35b}$$

  with $\mathbf{b}$ described in section 3.1.2. Behold that this scheme of solution is especially robust, supporting the use of redundant constraints or semi-definite positive matrices without any reformulation. Moreover, the solution is efficient, since the system matrix only has to be evaluated and factorized once per projection.

- Classical Lagrange method:

  The solution of (3.34) can be expressed in the form of the classical Lagrange method, with:

$$\begin{bmatrix}\bar{\mathbf{P}} & \mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\\ \mathbf{\Phi}_{\hat{\mathbf{z}}} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\dot{\mathbf{z}}\\ \boldsymbol{\sigma}\end{bmatrix}=\begin{bmatrix}\bar{\mathbf{P}}\left(\dot{\mathbf{z}}-\dot{\mathbf{z}}^{*}\right)-\mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\left(\boldsymbol{\sigma}+\varsigma\boldsymbol{\alpha}\mathbf{\Phi}_{t}\right)\\ \mathbf{0}\end{bmatrix} \tag{3.36}$$

  Behold that, although the iterative nature of (3.35) is avoided with (3.36), this system of equations involve more equations and numerical problems related to redundancy.

- Penalty method:

  Since the purpose of velocity projections is to keep the fulfillment of $\dot{\mathbf{\Phi}}=\mathbf{0}$ under determined acceptable orders (constraints cannot be exactly compelled with a projection technique [126]), this can be achieved with a penalty formulation with an appropriate selection of the projection penalty factor $\varsigma$:

$$\left(\bar{\mathbf{P}}+\varsigma\mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\alpha}\mathbf{\Phi}_{\hat{\mathbf{z}}}\right)\dot{\mathbf{z}}=\bar{\mathbf{P}}\dot{\mathbf{z}}^{*}-\mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\varsigma\boldsymbol{\alpha}\mathbf{b} \tag{3.37}$$

In the MBSLIM multibody library, both penalty and augmented Lagrangian methods have been implemented.

The process for the achievement of the acceleration projection equations is analog to the one unfolded for velocity projections. The minimization problem related to a projection onto the manifold $\ddot{\mathbf{\Phi}} = \mathbf{0}$ takes the form:

$$min_{\ddot{\mathbf{z}}} V = \frac{1}{2} \left(\ddot{\mathbf{z}} - \ddot{\mathbf{z}}^*\right)^{\mathrm{T}} \bar{\mathbf{P}} \left(\ddot{\mathbf{z}} - \ddot{\mathbf{z}}^*\right) \;\; s.\, t.\; \ddot{\mathbf{\Phi}} \left(\mathbf{z}, \dot{\mathbf{z}}, \ddot{\mathbf{z}}, t\right) = \mathbf{0} \tag{3.38}$$

The solution of (3.38) by means of the augmented Lagrangian method yields:

$$\left(\bar{\mathbf{P}} + \varsigma \mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \boldsymbol{\alpha} \mathbf{\Phi}_{\hat{\mathbf{z}}}\right) \ddot{\mathbf{z}} = \bar{\mathbf{P}} \ddot{\mathbf{z}}^* - \mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \left(\boldsymbol{\kappa}^{\{i+1\}} + \varsigma \boldsymbol{\alpha} \mathbf{c}\right) \tag{3.39a}$$

$$\boldsymbol{\kappa}^{\{i+1\}} = \boldsymbol{\kappa}^{\{i\}} + \varsigma \boldsymbol{\alpha} \ddot{\mathbf{\Phi}} \tag{3.39b}$$

being $\mathbf{c}$ a sum of terms appearing in second derivative of the constraint vector with respect to time, already commented in section 3.1.3.

If a penalty formulation is used for the projection instead of an augmented Lagrangian one, a computationally less demanding non-iterative scheme can be obtained. For non-iterative velocity projections, equation (3.37) apply, while non-iterative acceleration projection become:

$$\left(\bar{\mathbf{P}} + \varsigma \mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \boldsymbol{\alpha} \mathbf{\Phi}_{\hat{\mathbf{z}}}\right) \ddot{\mathbf{z}} = \bar{\mathbf{P}} \ddot{\mathbf{z}}^* - \mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \mathbf{c} \tag{3.40}$$

Observe that the notation is deliberately compacted, using $\mathbf{\Phi}_{\hat{\mathbf{z}}}$, $\mathbf{b}$ and $\mathbf{c}$ instead of its expanded expressions. With this notation, each term of the semi-recursive projections can be directly related to the equivalent term in Cartesian coordinate formulations [119,120]. It should be pointed out that despite the similarities between relative and Cartesian coordinates, velocity and acceleration projections in joint coordinate models involve in general less constraint equations but more complex derivatives.

This formulation has been implemented in the MBSLIM multibody library with the possibilities of selection of different penalty factors $\boldsymbol{\alpha}$ and $\varsigma$, and different schemes of projection solution (iterative or augmented Lagrangian projections and non-iterative or penalty projections). Nonetheless, only the mass matrix has been considered as projection matrix, primarily due to the fact that it is already computed in the solution of the augmented Lagrangian system.

Moreover, mass matrices have a series of properties that make them ideal for projection purposes. First, they are symmetric and semi-definite positive, which opens the possibility of using efficient symmetric solvers. And second, the resulting projections are unconditionally dissipative, i.e. they do not incorporate spurious energy, as it was demonstrated in [127] for positive definite mass matrices. Behold that this dissipative condition can be easily extended to semi-definite mass matrices since a variation on the additional “massless” variables do not change the kinetic energy of the system.

In general, velocity projections are sufficient to stabilize an index-3 formulation, but numerical experiments proved that acceleration projections improve the convergence rate at a minimum additional expense, regarding that the factorization of the velocity projection system matrix can be directly reused in the acceleration problem. Additionally, projections can be executed only under certain conditions of the constraint time derivatives violation, which contributes to minimize the computational effort without damaging accuracy or stability.

## 3.4 Semi-recursive penalty formulation

The solution of ALI3-P formulations using an implicit numerical integrator such as Newmark requires an initial value of the state accelerations in order to compute the corrector terms and to predict the set of positions, velocities and accelerations at the following instant of time. However, the ALI3-P scheme is not well posed to be solved at acceleration level, thus it is convenient to resort to other formulations for initialization purposes. Moreover, an initialization of the Lagrange multipliers, though it is not essential, is convenient to reduce the number of iterations of the augmented Lagrangian scheme during its first evaluation or to have an estimation of the Lagrange multipliers if needed, for example, for constraints reactions calculation.

The penalty formulation proposed by Bayo et al. [125] presents a solution for the dynamics of a multibody system by substituting the Lagrange multipliers of the classical Lagrange's index-1 formulation by a penalized term including the constraints vector in positions, velocities and accelerations. The resulting system takes the form of an ODE system (opposed to the classical Lagrange's formulation) with the state accelerations as unique unknowns. This problem is particularly convenient for the initialization of ALI3-P formulations due to its simplicity, its good fulfillment of constraints in positions, velocities and accelerations, and due to the fact that the Lagrange multipliers can be approximated from the constraints penalty term. It should be remarked that this initialization of the Lagrange multipliers is a good estimation, even though it does not provide the exact Lagrange multipliers of the augmented Lagrangian formulation.

Because the penalty formulation is exclusively intended here for initialization purposes, it will not be described in detail as the previous formulations, but just outlined. This formulation can be easily applied to relative coordinate models, yielding the following expression:

$$\mathbf{M}^d\ddot{\mathbf{z}} + \boldsymbol{\Phi}_{\mathbf{z}}^{\mathrm{T}}\boldsymbol{\alpha}\left(\ddot{\boldsymbol{\Phi}} + 2\Omega\xi\dot{\boldsymbol{\Phi}} + \Omega^2\boldsymbol{\Phi}\right) = \mathbf{Q}^d \tag{3.41}$$

Unlike the notation presented in [125], here the damping coefficient originally denoted as $\mu$ is renamed as $\xi$ in order to eliminate possible misunderstandings with the set of adjoint variables described in the following chapters. The ODE (3.41) can be directly solved using a fixed point scheme in accelerations. Behold that the ODE system (3.41) is transformed into a set of algebraic equations in initial acceleration problems, where the dependent coordinates vector is already known at position and velocity levels thanks to the kinematic initial position and velocity problems.

In fact, the use of a penalty formulation to initialize the accelerations is very similar to an index-1 scheme according to the imposition of constraints in the formulation, since both approaches at this particular initial time are basically incorporating the satisfaction of $\ddot{\mathbf{\Phi}}$. Besides, the penalty formulation is equal to the first iteration of an index-1 formulation if the index-1 system of equations is solved via an augmented Lagrangian scheme with the same penalty factor $\boldsymbol{\alpha}$, provided that the kinematic problems in positions and velocities are already solved.

Equation (3.41) can be reformulated so as to simplify notation by means of:

$$\breve{\mathbf{M}}\ddot{\mathbf{z}} = \breve{\mathbf{Q}} \tag{3.42}$$

in which:

$$\breve{\mathbf{M}} = \mathbf{M}^{d} + \mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\alpha}\mathbf{\Phi}_{\hat{\mathbf{z}}} \tag{3.43a}$$

$$\breve{\mathbf{Q}} = \mathbf{Q}^{d} - \mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\alpha}\left(\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}\dot{\mathbf{z}} + \dot{\mathbf{\Phi}}_{t} + 2\Omega\xi\dot{\mathbf{\Phi}} + \Omega^{2}\mathbf{\Phi}\right) \tag{3.43b}$$

where the leading matrix of the system $\breve{\mathbf{M}}$ is symmetric and always has inverse. The value of $\boldsymbol{\alpha}$ used in the MBSLIM implementation is equal to the one used for ALI3-P, while the other coefficients are $\Omega = 10$ and $\xi = 1$ by default.

The penalty formulation does not explicitly involve Lagrange multipliers, but the approximated index-3 augmented Lagrangian multipliers can be estimated with the help of the following expression, introduced in [125]:

$$\boldsymbol{\lambda}^{*} \approx \boldsymbol{\alpha}\left(\ddot{\mathbf{\Phi}} + 2\Omega\xi\dot{\mathbf{\Phi}} + \Omega^{2}\mathbf{\Phi}\right) \tag{3.44}$$

An improved method for the solution of this penalty formulation included as well in [125], based on the explicit incorporation of the Lagrange multipliers into the equations of motion through an iterative augmented Lagrangian scheme in order to eliminate the round-off errors caused by the non-infinite penalty term $\boldsymbol{\alpha}$, has not been considered here for two reasons: first, the round-off error is usually very small and only becomes relevant when it is accumulated in time, hence if the problem is solved exclusively at the initial time, the error generated can be neglected; and second, augmented Lagrangian iterations are sought to be avoided in order to simplify the subsequent sensitivity analyses and made initialization procedures as straightforward as possible without any loss of accuracy.

## 3.5 Topological constraints

In joint coordinate models, it is common to express constraint equations as a relation of points and vectors rather than joint coordinates. Let us consider for instance a constraint of coincidence of two points belonging to different bodies. The constraint equation in terms of Cartesian coordinates is forthright, while to express it in terms of joint coordinates is infeasible in a general manner. For this reason, an additional set of coordinates comprising positions of points and vectors is regarded as intermediate dependencies of a general constraint equation. Following the developments of chapter 2, it was established that joint-coordinates along with the recursive joint-dependent kinematic relations allow to determine the kinematics of any rigid body, thus the coordinates of a point or a vector belonging to it.

Regarding the particular implementation of joint coordinate models in MBSLIM, the use of this intermediate set of coordinates, in addition to the simplification of constraints, allows us to reuse the constraint equations already defined in MBSLIM in terms of Cartesian coordinates, and what is more important, to take advantage of the constraint derivatives already developed and tested.

Back into the constraint derivatives (3.3) and (3.13) required in the dynamics and kinematic analyses presented in former sections, it can be observed that the evaluation of constraint derivatives with respect to joint coordinates can be addressed by the direct application of the chain rule. In those expressions, constraint derivatives can be decomposed in partial derivatives of the constraint vector with respect to positions of points and vectors ($\mathbf{\Phi_q}$, $\dot{\mathbf{\Phi}}_\mathbf{q}$) and joint-coordinates ($\mathbf{\Phi_z}$, $\dot{\mathbf{\Phi}}_\mathbf{z}$), and other terms describing the derivative of the state of points and vectors with respect to the joint coordinates ($\mathbf{q_z}$, $\dot{\mathbf{q}}_\mathbf{z}$). Those last terms correspond to topological derivatives as long as they depend exclusively on the topology of the mechanism, and they are independent of the constraint equations. A first direct evaluation of $\mathbf{q_z}$ and $\dot{\mathbf{q}}_\mathbf{z}$ will be presented in section 3.6, while a more efficient evaluation will be tackled in section 4.5.

In relative coordinate models, there is a set of constraint equations that are directly related to the generation of the model, namely loop-closure constraints and the Euler parameters normalization constraint. In this section, the analytical expression of these constraint equations along with their partial derivatives $\mathbf{\Phi_q}$, $\dot{\mathbf{\Phi}}_\mathbf{q}$, $\mathbf{\Phi_z}$, $\dot{\mathbf{\Phi}}_\mathbf{z}$ are presented. The loop-closure constraint equations for each type of joint eliminated along with the normalization constraint of the Euler parameters have been implemented in the MBLSIM multibody library in the form they are presented below.

### 3.5.1 Euler parameters normalization constraint

The problem of singular configurations of any orientation or rotation parameterization by means of three coordinates is an issue which even today is investigated by the multibody community. In order to get rid of singular configurations and the reparameterization problems they imply, other orientation coordinates can be resorted to. In the present document and in the MBSLIM implementation, Euler parameters are used.

Euler parameters conform a four-coordinate parameterization of a rotation, and since only three out of four coordinates are independent, they have to comply with an additional constraint. In this case, this constraint subjects the vector of four parameters to have unitary norm, or which is the same, to be normalized.

$$\mathbf{\Phi} = \bar{\mathbf{p}}^{\mathrm{T}}\bar{\mathbf{p}} - 1 = 0 \tag{3.45}$$

Observe that this constraint depends exclusively on $\mathbf{z}$, therefore $\mathbf{\Phi_q} = \mathbf{0}$, $\dot{\mathbf{\Phi}}_\mathbf{q} = \mathbf{0}$ and the only nonzero derivatives are:

$$\mathbf{\Phi_z} = 2\bar{\mathbf{p}}^{\mathrm{T}} \tag{3.46}$$

$$\dot{\mathbf{\Phi}}_\mathbf{z} = 2\dot{\bar{\mathbf{p}}}^{\mathrm{T}} \tag{3.47}$$

For each spherical or floating joint present in a multibody model, in the case they are parameterized with Euler parameters, an additional normalization constraint shall be incorporated to the model.

## 3.5.2 Spherical joint: loop-closure constraint

An ideal spherical joint is equivalent to two bodies sharing one point. In the case of an opening procedure of a closed loop resulting with a cut of an spherical joint, the removed spherical joint must be replaced by a set of constraints that reproduce its behavior. Since this joint allows three degrees of freedom, it is logical to impose three independent constraint equations.

Let us consider that the spherical joint depicted in Figure 2.5 has to be removed and substituted by a loop-closure constraint. In that case, the shared point $j$ will be regarded as two different points belonging to two different bodies, whose positions will be $\mathbf{r}_j^{i-1}$ and $\mathbf{r}_j^i$. The constraint in this case will measure the deviation between the coordinates of these two points:

$$\mathbf{\Phi} = \mathbf{r}_j^i - \mathbf{r}_j^{i-1} \tag{3.48}$$

Observe that these constraints directly depend on $\mathbf{q}$, therefore $\mathbf{\Phi_z} = \mathbf{0}$, $\dot{\mathbf{\Phi}}_\mathbf{z} = \mathbf{0}$, and

$$\mathbf{\Phi_q} = \begin{bmatrix} \mathbf{I}_3 & -\mathbf{I}_3 \end{bmatrix} \tag{3.49}$$

$$\dot{\mathbf{\Phi}}_\mathbf{q} = \mathbf{0} \tag{3.50}$$

considering $\mathbf{q} = \begin{bmatrix} \mathbf{r}_j^i & \mathbf{r}_j^{i-1} \end{bmatrix}$.

## 3.5.3 Revolute joint: loop-closure constraint

The revolute joint, as it is displayed in Figure 2.8, is defined by one point and one vector shared between two bodies. In the case of a revolute joint removal resulting from an opening-of-the-closed-loop process, some additional constraints must enforce the coincidence of these points ($\mathbf{\Phi}_1$) and vectors ($\mathbf{\Phi}_2$). The proposed 6 constraint equations take the form:

$$\mathbf{\Phi}_1 = \mathbf{r}_j^i - \mathbf{r}_j^{i-1} \tag{3.51}$$

$$\mathbf{\Phi}_2 = \mathbf{v}_j^i - \mathbf{v}_j^{i-1} \tag{3.52}$$

In the case of (3.52), only two out of the three equations are independent since the constraint imposes the parallelism between two unit vectors. It should be pointed out that the possibility of redundant constraints is supported in every calculation developed and implemented in the present work, including kinematics, dynamics and sensitivities. Therefore, redundancy does not represent a problem, but it allows the use of simpler expressions with straightforward derivatives.

The constraints depend directly on $\mathbf{q}$, therefore $\mathbf{\Phi_z} = \mathbf{0}$, $\dot{\mathbf{\Phi}}_\mathbf{z} = \mathbf{0}$ and the only nonzero derivatives are,

$$(\mathbf{\Phi}_1)_\mathbf{q} = \begin{bmatrix}\mathbf{I}_3 & -\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3\end{bmatrix} \tag{3.53}$$

$$(\mathbf{\Phi}_2)_\mathbf{q} = \begin{bmatrix}\mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & -\mathbf{I}_3\end{bmatrix} \tag{3.54}$$

$$\left(\dot{\mathbf{\Phi}}_1\right)_\mathbf{q} = \mathbf{0} \tag{3.55}$$

$$\left(\dot{\mathbf{\Phi}}_2\right)_\mathbf{q} = \mathbf{0} \tag{3.56}$$

wherein $\mathbf{q} = \begin{bmatrix}\mathbf{r}_j^i & \mathbf{r}_j^{i-1} & \mathbf{v}_j^i & \mathbf{v}_j^{i-1}\end{bmatrix}$.

### 3.5.4 Cylindrical joint: loop-closure constraint

A cylindrical joint is equivalent to two bodies sharing a vector and with a condition of alignment of two points along this vector, each one belonging to each body. Looking at Figure 2.4 where this joint type is displayed, the shared vector is $\mathbf{w}_j$ and the points aligned $\mathbf{r}_j^i$ and $\mathbf{r}_{j-1}^{i-1}$. When this type of joint has to be eliminated, one constraint of vector coincidence ($\mathbf{\Phi}_1$) and another one of point alignment along this vector($\mathbf{\Phi}_2$) have to be added to the model.

$$\mathbf{\Phi}_1 = \mathbf{w}_j^i - \mathbf{w}_j^{i-1} \tag{3.57}$$

$$\mathbf{\Phi}_2 = \mathbf{w}_j^i \wedge \left(\mathbf{r}_j^i - \mathbf{r}_{j-1}^{i-1}\right) \tag{3.58}$$

Regard that (3.57) and (3.58) are expressed in terms of points and vectors, thus $\mathbf{\Phi_z} = \mathbf{0}$, $\dot{\mathbf{\Phi}}_\mathbf{z} = \mathbf{0}$ and:

$$(\mathbf{\Phi}_1)_\mathbf{q} = \begin{bmatrix}\mathbf{I}_3 & -\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3\end{bmatrix} \tag{3.59}$$

$$(\mathbf{\Phi}_2)_\mathbf{q} = \begin{bmatrix}\left(\mathbf{r}_j^i - \mathbf{r}_{j-1}^{i-1}\right)^\mathrm{T} & \mathbf{0}_3 & \left(\mathbf{w}_j^i\right)^\mathrm{T} & -\left(\mathbf{w}_j^i\right)^\mathrm{T}\end{bmatrix} \tag{3.60}$$

$$\left(\dot{\mathbf{\Phi}}_1\right)_\mathbf{q} = \mathbf{0} \tag{3.61}$$

$$\left(\dot{\mathbf{\Phi}}_2\right)_\mathbf{q} = \begin{bmatrix}\left(\dot{\mathbf{r}}_j^i - \dot{\mathbf{r}}_{j-1}^{i-1}\right)^\mathrm{T} & \mathbf{0}_3 & \left(\dot{\mathbf{w}}_j^i\right)^\mathrm{T} & -\left(\dot{\mathbf{w}}_j^i\right)^\mathrm{T}\end{bmatrix} \tag{3.62}$$

in which $\mathbf{q} = \begin{bmatrix}\mathbf{w}_j^i & \mathbf{w}_j^{i-1} & \mathbf{r}_j^i & \mathbf{r}_{j-1}^{i-1}\end{bmatrix}$.

### 3.5.5 Prismatic joint: loop-closure constraint

The prismatic joint can be seen as a particularization of the cylindrical joint, where the rotation around the joint axis is impeded. In addition to cylindrical joint constraints, the rotation must be prevented, and there are different possibilities for that. In this case, the rotation is avoided by means of a coincidence constraint between two additional vectors not aligned with the axis of the joint, each one contained in each of the bodies related by the joint.

In brief, the prismatic joint can by substituted by 3 constraint sets: one $\mathbf{\Phi}_1$ of coincidence of the vectors (one belonging to each body) defining the axis of the joint; a second constraint $\mathbf{\Phi}_2$ of alignment of two points along the vector of the joint; and a third one $\mathbf{\Phi}_3$ of coincidence of other two vectors used to define the constant rotation matrix $\mathbf{A}^i_{i+1}$ (see equation (2.31)).

$$\mathbf{\Phi}_1 = \mathbf{u}^i_j - \mathbf{u}^{i-1}_j \tag{3.63}$$

$$\mathbf{\Phi}_2 = \mathbf{u}^i_j \wedge \left(\mathbf{r}^i_j - \mathbf{r}^{i-1}_{j-1}\right) \tag{3.64}$$

$$\mathbf{\Phi}_3 = \mathbf{v}^i_j - \mathbf{v}^{i-1}_j \tag{3.65}$$

For these constraints, $\mathbf{\Phi}_\mathbf{z} = \mathbf{0}$, $\dot{\mathbf{\Phi}}_\mathbf{z} = \mathbf{0}$ and:

$$(\mathbf{\Phi}_1)_\mathbf{q} = \begin{bmatrix}\mathbf{I}_3 & -\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3\end{bmatrix} \tag{3.66}$$

$$(\mathbf{\Phi}_2)_\mathbf{q} = \begin{bmatrix}\left(\mathbf{r}^i_j - \mathbf{r}^{i-1}_{j-1}\right)^\mathrm{T} & \mathbf{0}_3 & \left(\mathbf{u}^i_j\right)^\mathrm{T} & -\left(\mathbf{u}^i_j\right)^\mathrm{T} & \mathbf{0}_3 & \mathbf{0}_3\end{bmatrix} \tag{3.67}$$

$$(\mathbf{\Phi}_3)_\mathbf{q} = \begin{bmatrix}\mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & -\mathbf{I}_3\end{bmatrix} \tag{3.68}$$

$$\left(\dot{\mathbf{\Phi}}_1\right)_\mathbf{q} = \mathbf{0} \tag{3.69}$$

$$\left(\dot{\mathbf{\Phi}}_2\right)_\mathbf{q} = \begin{bmatrix}\left(\dot{\mathbf{r}}^i_j - \dot{\mathbf{r}}^{i-1}_{j-1}\right)^\mathrm{T} & \mathbf{0}_3 & \left(\dot{\mathbf{u}}^i_j\right)^\mathrm{T} & -\left(\dot{\mathbf{u}}^i_j\right)^\mathrm{T} & \mathbf{0}_3 & \mathbf{0}_3\end{bmatrix} \tag{3.70}$$

$$\left(\dot{\mathbf{\Phi}}_3\right)_\mathbf{q} = \mathbf{0} \tag{3.71}$$

considering $\mathbf{q} = \begin{bmatrix}\mathbf{u}^i_j & \mathbf{u}^{i-1}_j & \mathbf{r}^i_j & \mathbf{r}^{i-1}_{j-1} & \mathbf{v}^i_j & \mathbf{v}^{i-1}_j\end{bmatrix}$.

### 3.5.6 Cardan joint: loop-closure constraint

The Cardan or universal joint constraint equations stem from the perpendicularity of the two axis defining the joint and from the sharing of a point (see section 2.1.3). Regarding the notation introduced in Figure 2.3, the perpendicular vectors are denoted as $\mathbf{w}_j$ and $\mathbf{w}_{j+1}$ and the two points generated by the shared point, once the joint is eliminated, are identified as $\mathbf{r}^i_j$ and $\mathbf{r}^{i-1}_j$.

$$\mathbf{\Phi}_1 = \mathbf{r}^i_j - \mathbf{r}^{i-1}_j \tag{3.72}$$

$$\mathbf{\Phi}_2 = \mathbf{w}^\mathrm{T}_j \mathbf{w}_{j+1} \tag{3.73}$$

These constraints depend explicitly on points and vectors, therefore $\mathbf{\Phi_z} = \mathbf{0}$, $\dot{\mathbf{\Phi}}_\mathbf{z} = \mathbf{0}$ and:

$$(\mathbf{\Phi}_1)_\mathbf{q} = \begin{bmatrix}\mathbf{I}_3 & -\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3\end{bmatrix} \tag{3.74}$$

$$(\mathbf{\Phi}_2)_\mathbf{q} = \begin{bmatrix}\mathbf{0}_3 & \mathbf{0}_3 & \mathbf{w}_{j+1}^{\mathrm{T}} & \mathbf{w}_j^{\mathrm{T}}\end{bmatrix} \tag{3.75}$$

$$\left(\dot{\mathbf{\Phi}}_1\right)_\mathbf{q} = \mathbf{0} \tag{3.76}$$

$$\left(\dot{\mathbf{\Phi}}_2\right)_\mathbf{q} = \begin{bmatrix}\mathbf{0}_3 & \mathbf{0}_3 & \dot{\mathbf{w}}_{j+1}^{\mathrm{T}} & \dot{\mathbf{w}}_j^{\mathrm{T}}\end{bmatrix} \tag{3.77}$$

$$\tag{3.78}$$

wherein $\mathbf{q} = \begin{bmatrix}\mathbf{r}_j^i & \mathbf{r}_j^{i-1} & \mathbf{w}_j & \mathbf{w}_{j+1}\end{bmatrix}$.

### 3.5.7 Planar joint: loop-closure constraint

The planar joint, depicted in Figure 2.7, is defined by a vector shared between two bodies along with a point-in-plane constraint. The constraints needed to describe the joint allowed motion are, accordingly, a constraint of coincidence for the vectors and a condition of perpendicularity between the vector of the joint and the vector defined by the point belonging to the plane in the first body and the point of the second body moving in the plane, so these two points are forced to be in a plane normal to the vector of the joint.

$$\mathbf{\Phi}_1 = \mathbf{w}_j^i - \mathbf{w}_j^{i-1} \tag{3.79}$$

$$\mathbf{\Phi}_2 = \mathbf{w}_j^{i\mathrm{T}}\left(\mathbf{r}_j^i - \mathbf{r}_{j-1}^{i-1}\right) \tag{3.80}$$

The constraints are calculated in terms of $\mathbf{q}$, so $\mathbf{\Phi_z} = \mathbf{0}$, $\dot{\mathbf{\Phi}}_\mathbf{z} = \mathbf{0}$ and:

$$(\mathbf{\Phi}_1)_\mathbf{q} = \begin{bmatrix}\mathbf{I}_3 & -\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3\end{bmatrix} \tag{3.81}$$

$$(\mathbf{\Phi}_2)_\mathbf{q} = \begin{bmatrix}\left(\mathbf{r}_j^i - \mathbf{r}_{j-1}^{i-1}\right)^{\mathrm{T}} & \mathbf{0}_3 & \mathbf{w}_j^{i\mathrm{T}} & -\mathbf{w}_j^{i\mathrm{T}}\end{bmatrix} \tag{3.82}$$

$$\left(\dot{\mathbf{\Phi}}_1\right)_\mathbf{q} = \mathbf{0} \tag{3.83}$$

$$\left(\dot{\mathbf{\Phi}}_2\right)_\mathbf{q} = \begin{bmatrix}\left(\dot{\mathbf{r}}_j^i - \dot{\mathbf{r}}_{j-1}^{i-1}\right)^{\mathrm{T}} & \mathbf{0}_3 & \dot{\mathbf{w}}_j^{i\mathrm{T}} & -\dot{\mathbf{w}}_j^{i\mathrm{T}}\end{bmatrix} \tag{3.84}$$

with $\mathbf{q} = \begin{bmatrix}\mathbf{w}_j^i & \mathbf{w}_j^{i-1} & \mathbf{r}_j^i & \mathbf{r}_{j-1}^{i-1}\end{bmatrix}$.

## 3.6 Topological derivatives: $\mathbf{q_z}$

The most direct evaluation of the derivatives of the position of a point or a vector with respect to the relative coordinates in positions involve the consideration of the following identity:

$$\frac{\partial \mathbf{r}_j}{\partial \mathbf{z}} = \frac{\partial \dot{\mathbf{r}}_j}{\partial \dot{\mathbf{z}}} \tag{3.85}$$

This equivalence can be easy proved considering the dependencies of positions and velocities of any point or vector. Let us consider a point $j$, whose velocity can be calculated as:

$$\dot{\mathbf{r}}_j = \frac{\mathrm{d}\mathbf{r}_j}{\mathrm{d}t} = \frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}\dot{\mathbf{z}} \tag{3.86}$$

Observe that the position of a point does not depend explicitly on time. If the previous expression is differentiated with respect to $\dot{\mathbf{z}}$:

$$\frac{\partial \dot{\mathbf{r}}_j}{\partial \dot{\mathbf{z}}} = \frac{\partial \mathbf{r}_j}{\partial \mathbf{z}} \tag{3.87}$$

An analog derivation is valid for the equivalent derivative of vectors.

Using (3.85) and the distribution of velocities in a rigid body, (2.11) and (2.12),

$$\frac{\partial \mathbf{r}_j}{\partial \mathbf{z}} = \frac{\partial \dot{\mathbf{r}}_j}{\partial \dot{\mathbf{z}}} = \frac{\partial \dot{\mathbf{r}}_i}{\partial \dot{\mathbf{z}}} + \frac{\partial \boldsymbol{\omega}_i}{\partial \dot{\mathbf{z}}} \wedge \left(\mathbf{r}_j - \mathbf{r}_i\right) = \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_j\end{bmatrix} \frac{\partial \mathbf{V}_i}{\partial \dot{\mathbf{z}}} \tag{3.88}$$

$$\frac{\partial \mathbf{u}_j}{\partial \mathbf{z}} = \frac{\partial \dot{\mathbf{u}}_j}{\partial \dot{\mathbf{z}}} = \frac{\partial \boldsymbol{\omega}_i}{\partial \dot{\mathbf{z}}} \wedge \mathbf{u}_j = \begin{bmatrix}\mathbf{0} & -\tilde{\mathbf{u}}_j\end{bmatrix} \frac{\partial \mathbf{V}_i}{\partial \dot{\mathbf{z}}} \tag{3.89}$$

where $\mathbf{r}_i$ is the reference point in body $i$ and $\mathbf{V}_i = \begin{bmatrix}\dot{\mathbf{r}}_i^{\mathrm{T}} & \boldsymbol{\omega}_i{}^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}}$.

Now, recalling here the recursive relations (2.204):

$$\mathbf{V} = \mathbf{R}^v\dot{\mathbf{z}} \Rightarrow \mathbf{V}_i = \mathbf{R}_i^v\dot{\mathbf{z}} \Rightarrow \frac{\partial \mathbf{V}_i}{\partial \dot{\mathbf{z}}} = \mathbf{R}_i^v \tag{3.90}$$

where $\mathbf{R}_i^v$ are the six rows of the full matrix $\mathbf{R}^v$ corresponding to body $i$.

Then, the kinematic term $\mathbf{q_z}$ can be built from the derivatives $\frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}$ of points and $\frac{\partial \mathbf{u}_j}{\partial \mathbf{z}}$ of vectors, by applying the following expressions:

$$\frac{\partial \mathbf{r}_j}{\partial \mathbf{z}} = \frac{\partial \dot{\mathbf{r}}_j}{\partial \dot{\mathbf{z}}} = \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_j\end{bmatrix} \mathbf{R}_i^v \tag{3.91}$$

$$\frac{\partial \mathbf{u}_j}{\partial \mathbf{z}} = \frac{\partial \dot{\mathbf{u}}_j}{\partial \dot{\mathbf{z}}} = \begin{bmatrix}\mathbf{0} & -\tilde{\mathbf{u}}_j\end{bmatrix} \mathbf{R}_i^v \tag{3.92}$$

Behold that, since equations (3.91) and (3.92) depend explicitly on the reference point selected, they have different expressions for the versions RTdyn0 and RTdyn1. For the RTdyn0 formulation, equations (3.91) and (3.92) become:

$$\frac{\partial \mathbf{r}_j}{\partial \mathbf{z}} = \frac{\partial \dot{\mathbf{r}}_j}{\partial \dot{\mathbf{z}}} = \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_j\end{bmatrix} \mathbf{R}_i^y \tag{3.93}$$

$$\frac{\partial \mathbf{u}_j}{\partial \mathbf{z}} = \frac{\partial \dot{\mathbf{u}}_j}{\partial \dot{\mathbf{z}}} = \begin{bmatrix}\mathbf{0} & -\tilde{\mathbf{u}}_j\end{bmatrix} \mathbf{R}_i^y \tag{3.94}$$

In the case of reference points coincident with the origin of coordinates (RTdyn1), equations (3.91) and (3.92) take the form:

$$\frac{\partial \mathbf{r}_j}{\partial \mathbf{z}} = \frac{\partial \dot{\mathbf{r}}_j}{\partial \dot{\mathbf{z}}} = \begin{bmatrix}\mathbf{I} & -\tilde{\mathbf{r}}_j\end{bmatrix} \mathbf{R}_i^z \tag{3.95}$$

$$\frac{\partial \mathbf{u}_j}{\partial \mathbf{z}} = \frac{\partial \dot{\mathbf{u}}_j}{\partial \dot{\mathbf{z}}} = \begin{bmatrix}\mathbf{0} & -\tilde{\mathbf{u}}_j\end{bmatrix} \mathbf{R}_i^z \tag{3.96}$$

A special case is the derivative of the position of the reference point with respect to the relative coordinates in the RTdyn1 formulation. Since this position is always constant and equal to the origin of coordinates, it does not vary with any value of the joint coordinates, thus its derivative is always null. Behold that, in this case, the expression presented in (3.91) is incorrect for this type of point:

$$\frac{\partial \mathbf{r}_i^z}{\partial \mathbf{z}} \neq \frac{\partial \dot{\mathbf{r}}_i^z}{\partial \dot{\mathbf{z}}} \tag{3.97}$$

In section 4.5.1, a more efficient evaluation of derivatives of positions of points and vectors with respect to relative coordinates will be developed. However, these simplified expressions will be derived from the ones presented in this section, so both approaches and derivative expressions will be equally valid.

## 3.7 Topological derivatives: $\dot{\mathbf{q}}_\mathbf{z}$

The evaluation of $\dot{\mathbf{q}}_\mathbf{z}$ can be carried out by simply taking temporal derivatives of equations (3.91) and (3.92).

$$\frac{\partial \dot{\mathbf{r}}_j}{\partial \mathbf{z}} = \begin{bmatrix}\mathbf{0} & \dot{\tilde{\mathbf{r}}}_i - \dot{\tilde{\mathbf{r}}}_j\end{bmatrix} \mathbf{R}_i^v + \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_j\end{bmatrix} \dot{\mathbf{R}}_i^v \tag{3.98}$$

$$\frac{\partial \dot{\mathbf{u}}_j}{\partial \mathbf{z}} = \begin{bmatrix}\mathbf{0} & -\dot{\tilde{\mathbf{u}}}_j\end{bmatrix} \mathbf{R}_i^v + \begin{bmatrix}\mathbf{0} & -\tilde{\mathbf{u}}_j\end{bmatrix} \dot{\mathbf{R}}_i^v \tag{3.99}$$

It turns out the need for an extra term, not derived before, $\dot{\mathbf{R}}^v$, whose composition will be attained in section 3.8.

Note that the temporal derivative of $\mathbf{q}_\mathbf{z}$ is equivalent to the partial derivative of $\dot{\mathbf{q}}$ with respect to $\mathbf{z}$. In order to explain this relation, let us differentiate with respect to time the derivative of the position of a point with respect to $\mathbf{z}$:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}\right) = \frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}\right)\dot{\mathbf{z}} + \frac{\partial}{\partial t}\left(\frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}\right) = \frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}\right)\dot{\mathbf{z}} \tag{3.100}$$

in which $\frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}$ does not depend explicitly on time. Now, let us consider the derivative of the velocity of a point with respect to $\mathbf{z}$:

$$\frac{\partial \dot{\mathbf{r}}_j}{\partial \mathbf{z}} = \frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}\dot{\mathbf{z}}\right) = \frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}\right)\dot{\mathbf{z}} \tag{3.101}$$

Expressions (3.100) and (3.101) can be easily extended to vectors too, resulting equivalent relations. From eq (3.100) and (3.101), the following identity can be deduced:

$$\dot{\mathbf{q}}_{\mathbf{z}} = \frac{\mathrm{d}\mathbf{q}_{\mathbf{z}}}{\mathrm{d}t} \tag{3.102}$$

This relation will be extremely useful in the differentiation of the equations of motion, tackled in the sensitivity chapters 4 and 5.

A second strategy for calculating $\dot{\mathbf{q}}_{\mathbf{z}}$ consist in taking partial derivatives in the equations of distribution of velocities in a rigid body, (2.11) and (2.12),

$$\frac{\partial \dot{\mathbf{r}}_j}{\partial \mathbf{z}} = \begin{bmatrix} \mathbf{0} & \frac{\partial \tilde{\mathbf{r}}_i}{\partial \mathbf{z}} - \frac{\partial \tilde{\mathbf{r}}_j}{\partial \mathbf{z}} \end{bmatrix} \mathbf{V}_i + \begin{bmatrix} \mathbf{I} & \tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_j \end{bmatrix} \frac{\partial \mathbf{V}_i}{\partial \mathbf{z}} \tag{3.103}$$

$$\frac{\partial \dot{\mathbf{u}}_j}{\partial \mathbf{z}} = \begin{bmatrix} \mathbf{0} & -\frac{\partial \tilde{\mathbf{u}}_j}{\partial \mathbf{z}} \end{bmatrix} \mathbf{V}_i + \begin{bmatrix} \mathbf{0} & -\tilde{\mathbf{u}}_j \end{bmatrix} \frac{\partial \mathbf{V}_i}{\partial \mathbf{z}} \tag{3.104}$$

Both strategies should be equivalent and lead to the same expressions.

The expressions above are valid for any point or vector contained in a body (fixed in the local reference frame), but cannot be applied to other floating entities, like the reference point in RTdyn1 formulations. In this case, since (3.91) is incorrect for points fixed in the global frame, it cannot be differentiated with respect to time for that point types.

Likewise in the calculation of $\mathbf{q}_{\mathbf{z}}$, an expression independent of the reference points can be reached for $\dot{\mathbf{q}}_{\mathbf{z}}$. Those simplified and more efficient expressions will be set forth in section 4.5.2.

## 3.8 Evaluation of $\dot{\mathbf{R}}$

In section 3.7, the need of $\dot{\mathbf{R}}^v$ has been made apparent for the assessment of the derivatives of Cartesian velocities with respect to the relative coordinates in positions. Even though this derivative can be solved without the computation of $\dot{\mathbf{R}}^v$, as it will be explained in section 4.5.2, this matrix will come up in other several occasions, and consequently, it will need to be evaluated.

Let us consider here again the mechanism of figure 2.10, whose matrix $\mathbf{R}^v$ was previously assembled in (2.211). The matrix $\dot{\mathbf{R}}^v$ corresponds to its time derivative, thus:

$$\dot{\mathbf{R}}^v = \begin{bmatrix} \dot{\mathbf{b}}^v_{1,1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \dot{\mathbf{b}}^v_{2,1} & \dot{\mathbf{b}}^v_{2,2} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \dot{\mathbf{b}}^v_{3,1} & \dot{\mathbf{b}}^v_{3,2} & \dot{\mathbf{b}}^v_{3,3} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \dot{\mathbf{b}}^v_{4,1} & \mathbf{0} & \mathbf{0} & \dot{\mathbf{b}}^v_{4,4} & \mathbf{0} & \mathbf{0} \\ \dot{\mathbf{b}}^v_{5,1} & \mathbf{0} & \mathbf{0} & \dot{\mathbf{b}}^v_{5,4} & \dot{\mathbf{b}}^v_{5,5} & \mathbf{0} \\ \dot{\mathbf{b}}^v_{6,1} & \mathbf{0} & \mathbf{0} & \dot{\mathbf{b}}^v_{6,4} & \mathbf{0} & \dot{\mathbf{b}}^v_{6,6} \end{bmatrix} \tag{3.105}$$

where the terms $\dot{\mathbf{b}}^v_{i,j}$ can be obtained recursively with the same logic applied to the calculation of the $\mathbf{b}^v_{i,j}$ terms:

$$\dot{\mathbf{b}}^v_{i,i} = \dot{\mathbf{b}}^v_i \tag{3.106a}$$

$$\dot{\mathbf{b}}^v_{i,j} = \dot{\mathbf{B}}^v_i \mathbf{b}^v_{h,j} + \mathbf{B}^v_i \dot{\mathbf{b}}^v_{h,j};\ i > j \tag{3.106b}$$

$$\dot{\mathbf{b}}^v_{i,j} = \mathbf{0};\ i < j \tag{3.106c}$$

where $h$ is the parent body of $i$, i.e. the preceding body in the kinematic chain. These relations are calculated recursively, traversing the rows of the matrix, from the root to the leaves of the mechanism.

The previous assembly (3.105) is valid for any of the semi-recursive methods presented, but it can be further simplified for RTdyn1:

$$\dot{\mathbf{R}}^z = \begin{bmatrix} \dot{\mathbf{b}}^z_{1,1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \dot{\mathbf{b}}^z_{2,1} & \dot{\mathbf{b}}^z_{2,2} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \dot{\mathbf{b}}^z_{3,1} & \dot{\mathbf{b}}^z_{3,2} & \dot{\mathbf{b}}^z_{3,3} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \dot{\mathbf{b}}^z_{4,1} & \mathbf{0} & \mathbf{0} & \dot{\mathbf{b}}^z_{4,4} & \mathbf{0} & \mathbf{0} \\ \dot{\mathbf{b}}^z_{5,1} & \mathbf{0} & \mathbf{0} & \dot{\mathbf{b}}^z_{5,4} & \dot{\mathbf{b}}^z_{5,5} & \mathbf{0} \\ \dot{\mathbf{b}}^z_{6,1} & \mathbf{0} & \mathbf{0} & \dot{\mathbf{b}}^z_{6,4} & \mathbf{0} & \dot{\mathbf{b}}^z_{6,6} \end{bmatrix} \tag{3.107}$$

where $\dot{\mathbf{b}}^z_{i,j}$ can be obtained recursively as:

$$\dot{\mathbf{b}}^z_{i,i} = \dot{\mathbf{b}}^z_i \tag{3.108a}$$

$$\dot{\mathbf{b}}^z_{i,j} = \dot{\mathbf{B}}^z_i \mathbf{b}^z_{h,j} + \dot{\mathbf{b}}^z_{h,j};\ i > j \tag{3.108b}$$

$$\dot{\mathbf{b}}^z_{i,j} = \mathbf{0};\ i < j \tag{3.108c}$$

in which, again, $h$ is the parent body of $i$.

It is important to remark that the complete evaluation and assembly of both $\mathbf{R}^v$ or $\dot{\mathbf{R}}^v$ is avoided in the MBSLIM implementation. As it was presented in section 2.4.1, the mass matrix and generalized forces vector can be easily assembled without computing these two matrices, and with regard to the point and vector derivatives described in previous sections, these matrices are not needed in their totality, but only a few rows of them.

# Chapter 4

# Sensitivity analysis of unconstrained open-loop systems

In this chapter, the sensitivity analysis of the unconstrained open-loop dynamic formulations described in chapter 2 is addressed using the direct differentiation method. As a result, a semi-recursive and a fully-recursive forward sensitivity formulations have been achieved. First, the general form of the semi-recursive topological sensitivity formulation is introduced and then the derivatives of the mass matrix and generalized forces vector are presented. In fully-recursive sensitivity formulations, the focus is placed upon recursivity and the accumulation process.

Both semi-recursive and fully-recursive formulations have been differentiated considering an arbitrary selection of reference points, thus the expressions presented in this chapter are general in this regard. In addition, each derivative involved in these sensitivity formulations has been also particularized for the cases of RTdyn0 and RTdyn1, delivering more direct and simplified expressions.

Special attention has been paid to the derivatives of the recursive kinematic relations, which are described for each of the joint types included in chapter 2. Moreover, the derivatives of Cartesian coordinates have been deeply studied in order to enhance the computational efficiency of both semi-recursive and fully-recursive sensitivity formulations.

As it has been made apparent in chapter 2, semi-recursive and fully-recursive dynamic formulations require a series of accumulation and assembly operations that involve products of different kinematic and dynamic magnitudes. An expression involving a concatenation of products entails an explosion of terms in its derivatives (applying the chain rule of differentiation), which have to be carefully handled and inspected in order to avoid repetitions and maximize efficiency. In the current chapter, differentiation is accomplished according to this criterion.

Every derivative expression included in this chapter has been analytically obtained and then implemented and tested in the MBSLIM general purpose multibody library [5].

## 4.1 Introduction to sensitivity analysis on joint coordinates

First of all, let us consider a vector of objective functions[1] $\boldsymbol{\psi} \in \mathbb{R}^o$ expressed in terms of the relative coordinates $\mathbf{z}$, $\dot{\mathbf{z}}$ and $\ddot{\mathbf{z}}$, a set of natural coordinates $\mathbf{q}$, $\dot{\mathbf{q}}$, $\ddot{\mathbf{q}}$ and a set of parameters $\boldsymbol{\rho} \in \mathbb{R}^p$, being $p$ the number of parameters of the system:

$$\boldsymbol{\psi} = \int_{t_0}^{t_F} \mathbf{g}\left(\mathbf{z}, \dot{\mathbf{z}}, \ddot{\mathbf{z}}, \mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \boldsymbol{\rho}\right) \mathrm{d}t \tag{4.1}$$

wherein $\mathbf{g} \in \mathbb{R}^o$ is a known vector of functions dependent on the parameters and the instant values of the states.

Considering the explicit dependencies of (4.1), the gradient of the vector of objective functions takes the form:

$$\boldsymbol{\psi}' = \nabla\boldsymbol{\psi}^T = \int_{t_0}^{t_F} \left(\mathbf{g}_{\mathbf{z}}\mathbf{z}' + \mathbf{g}_{\dot{\mathbf{z}}}\dot{\mathbf{z}}' + \mathbf{g}_{\ddot{\mathbf{z}}}\ddot{\mathbf{z}}' + \mathbf{g}_{\mathbf{q}}\mathbf{q}' + \mathbf{g}_{\dot{\mathbf{q}}}\dot{\mathbf{q}}' + \mathbf{g}_{\ddot{\mathbf{q}}}\ddot{\mathbf{q}}' + \mathbf{g}_{\boldsymbol{\rho}}\right) \mathrm{d}t \tag{4.2}$$

where the derivatives of $\mathbf{g}$ are known, $\mathbf{z}'$, $\dot{\mathbf{z}}'$ and $\ddot{\mathbf{z}}'$ represent the sensitivities of the relative coordinates and $\mathbf{q}'$, $\dot{\mathbf{q}}'$ and $\ddot{\mathbf{q}}'$ are the sensitivities of the natural coordinates on which the objective function depends.

Since the model is completely defined by the set of relative coordinates, any natural coordinate (i.e. the position, velocity and acceleration of any point or vector of the model) can be computed using relative coordinates. Therefore:

$$\mathbf{q}' = \frac{\mathrm{d}\mathbf{q}}{\mathrm{d}\boldsymbol{\rho}} = \frac{\partial\mathbf{q}}{\partial\mathbf{z}}\frac{\partial\mathbf{z}}{\partial\boldsymbol{\rho}} + \frac{\partial\mathbf{q}}{\partial\boldsymbol{\rho}} = \mathbf{q}_{\mathbf{z}}\mathbf{z}' + \mathbf{q}_{\boldsymbol{\rho}} \tag{4.3a}$$

$$\dot{\mathbf{q}}' = \frac{\mathrm{d}\dot{\mathbf{q}}}{\mathrm{d}\boldsymbol{\rho}} = \frac{\partial\dot{\mathbf{q}}}{\partial\mathbf{z}}\frac{\partial\mathbf{z}}{\partial\boldsymbol{\rho}} + \frac{\partial\dot{\mathbf{q}}}{\partial\dot{\mathbf{z}}}\frac{\partial\dot{\mathbf{z}}}{\partial\boldsymbol{\rho}} + \frac{\partial\dot{\mathbf{q}}}{\partial\boldsymbol{\rho}} = \dot{\mathbf{q}}_{\mathbf{z}}\mathbf{z}' + \mathbf{q}_{\mathbf{z}}\dot{\mathbf{z}}' + \dot{\mathbf{q}}_{\boldsymbol{\rho}} \tag{4.3b}$$

$$\ddot{\mathbf{q}}' = \frac{\mathrm{d}\ddot{\mathbf{q}}}{\mathrm{d}\boldsymbol{\rho}} = \frac{\partial\ddot{\mathbf{q}}}{\partial\mathbf{z}}\frac{\partial\mathbf{z}}{\partial\boldsymbol{\rho}} + \frac{\partial\ddot{\mathbf{q}}}{\partial\dot{\mathbf{z}}}\frac{\partial\dot{\mathbf{z}}}{\partial\boldsymbol{\rho}} + \frac{\partial\ddot{\mathbf{q}}}{\partial\ddot{\mathbf{z}}}\frac{\partial\ddot{\mathbf{z}}}{\partial\boldsymbol{\rho}} + \frac{\partial\mathbf{q}}{\partial\boldsymbol{\rho}} = \ddot{\mathbf{q}}_{\mathbf{z}}\mathbf{z}' + 2\dot{\mathbf{q}}_{\mathbf{z}}\dot{\mathbf{z}}' + \mathbf{q}_{\mathbf{z}}\ddot{\mathbf{z}}' + \ddot{\mathbf{q}}_{\boldsymbol{\rho}} \tag{4.3c}$$

where the following identities have been considered:

$$\dot{\mathbf{q}}_{\dot{\mathbf{z}}} = \mathbf{q}_{\mathbf{z}} \tag{4.4a}$$

$$\ddot{\mathbf{q}}_{\ddot{\mathbf{z}}} = \mathbf{q}_{\mathbf{z}} \tag{4.4b}$$

$$\ddot{\mathbf{q}}_{\dot{\mathbf{z}}} = 2\dot{\mathbf{q}}_{\mathbf{z}} \tag{4.4c}$$

Expressions (4.4a) and (4.4b) are almost straightforward, but (4.4c) needs further explanation. The natural coordinates accelerations are the time derivative of the

[1]Not all of them need to be objective functions, some of them could be design or optimization constraints for which the gradient with respect to the parameters is also needed.

velocities and the second time derivative of the positions:

$$\dot{\mathbf{q}}(\mathbf{z},\dot{\mathbf{z}}) = \frac{\mathrm{d}}{\mathrm{d}t}(\mathbf{q}(\mathbf{z})) = \frac{\partial \mathbf{q}}{\partial \mathbf{z}}\dot{\mathbf{z}} \tag{4.5a}$$

$$\ddot{\mathbf{q}}(\mathbf{z},\dot{\mathbf{z}},\ddot{\mathbf{z}}) = \frac{\mathrm{d}^2}{\mathrm{d}t^2}(\mathbf{q}(\mathbf{z})) = \left(\frac{\partial^2 \mathbf{q}}{\partial \mathbf{z}^2}\dot{\mathbf{z}}\right)\dot{\mathbf{z}} + \left(\frac{\partial \mathbf{q}}{\partial \mathbf{z}}\right)\ddot{\mathbf{z}} \tag{4.5b}$$

Now, differentiating (4.5a) with respect to $\mathbf{z}$ and (4.5b) with respect to $\dot{\mathbf{z}}$:

$$\frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{z}} = \frac{\partial^2 \mathbf{q}}{\partial \mathbf{z}^2}\dot{\mathbf{z}} \tag{4.6a}$$

$$\frac{\partial \ddot{\mathbf{q}}}{\partial \dot{\mathbf{z}}} = 2\frac{\partial^2 \mathbf{q}}{\partial \mathbf{z}^2}\dot{\mathbf{z}} \tag{4.6b}$$

Equation (4.4c) is followed from (4.6b).

Another important conclusion is that $\mathbf{q}$, this is, points and vectors, can depend directly on different parameters, as the local coordinates of a point or a length of a body, for instance, and accordingly, a set of partial derivatives with respect to $\boldsymbol{\rho}$ will appear during the sensitivity analysis. However, the derivatives $\mathbf{q}_{\boldsymbol{\rho}}$, $\dot{\mathbf{q}}_{\boldsymbol{\rho}}$ and $\ddot{\mathbf{q}}_{\boldsymbol{\rho}}$ have known expressions based on the rigid body equations and the recursive relations between bodies, hence their calculation is straightforward.

Once identified all the dependencies and derivatives of the natural coordinates with respect to relative, expressions (4.3) can be substituted in (4.2), and regrouping:

$$\boldsymbol{\psi}' = \int_{t_0}^{t_F} \left(\mathbf{g}_{\hat{\mathbf{z}}}\mathbf{z}' + \mathbf{g}_{\hat{\dot{\mathbf{z}}}}\dot{\mathbf{z}}' + \mathbf{g}_{\hat{\ddot{\mathbf{z}}}}\ddot{\mathbf{z}}' + \mathbf{g}_{\hat{\boldsymbol{\rho}}}\right)\mathrm{d}t \tag{4.7}$$

where:

$$\mathbf{g}_{\hat{\mathbf{z}}} = \mathbf{g}_{\mathbf{z}} + \mathbf{g}_{\mathbf{q}}\mathbf{q}_{\mathbf{z}} + \mathbf{g}_{\dot{\mathbf{q}}}\dot{\mathbf{q}}_{\mathbf{z}} + \mathbf{g}_{\ddot{\mathbf{q}}}\ddot{\mathbf{q}}_{\mathbf{z}} \tag{4.8a}$$

$$\mathbf{g}_{\hat{\dot{\mathbf{z}}}} = \mathbf{g}_{\dot{\mathbf{z}}} + \mathbf{g}_{\dot{\mathbf{q}}}\mathbf{q}_{\mathbf{z}} + 2\mathbf{g}_{\ddot{\mathbf{q}}}\dot{\mathbf{q}}_{\mathbf{z}} \tag{4.8b}$$

$$\mathbf{g}_{\hat{\ddot{\mathbf{z}}}} = \mathbf{g}_{\ddot{\mathbf{z}}} + \mathbf{g}_{\ddot{\mathbf{q}}}\mathbf{q}_{\mathbf{z}} \tag{4.8c}$$

$$\mathbf{g}_{\hat{\boldsymbol{\rho}}} = \mathbf{g}_{\mathbf{q}}\mathbf{q}_{\boldsymbol{\rho}} + \mathbf{g}_{\dot{\mathbf{q}}}\dot{\mathbf{q}}_{\boldsymbol{\rho}} + \mathbf{g}_{\ddot{\mathbf{q}}}\ddot{\mathbf{q}}_{\boldsymbol{\rho}} + \mathbf{g}_{\boldsymbol{\rho}} \tag{4.8d}$$

It must be pointed out that all the terms appearing in (4.8) have known expressions, since the derivatives of $\mathbf{g}$ are known and $\mathbf{q}_{\mathbf{z}}$, $\dot{\mathbf{q}}_{\mathbf{z}}$ and $\ddot{\mathbf{q}}_{\mathbf{z}}$ can be computed following the topology of the mechanism as explained in sections 3.6 and 3.7. Therefore, in (4.7), $\mathbf{z}'$, $\dot{\mathbf{z}}'$ and $\ddot{\mathbf{z}}'$ are the unique unknown matrices, which can be obtained differentiating the equations of motion with respect to the set of parameters.

## 4.2 Forward sensitivity of semi-recursive EoM for open-loop systems

In the sensitivity analysis of the dynamics, specially in topological models, several different derivatives and concatenations of products appear, which can make the resulting expressions difficult to understand. For the sake of clarity, a new rule of differentiation and notation is used. Given a function $f$ dependent on $\mathbf{z}$, $\dot{\mathbf{z}}$, $\ddot{\mathbf{z}}$, $\mathbf{q}$, $\dot{\mathbf{q}}$, $\ddot{\mathbf{q}}$ and $\boldsymbol{\rho}$, the following derivative is considered.

$$f_{\hat{\mathbf{x}}} = \frac{\partial f}{\partial \mathbf{q}}\frac{\partial \mathbf{q}}{\partial \mathbf{x}} + \frac{\partial f}{\partial \dot{\mathbf{q}}}\frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{x}} + \frac{\partial f}{\partial \ddot{\mathbf{q}}}\frac{\partial \ddot{\mathbf{q}}}{\partial \mathbf{x}} + \frac{\partial f}{\partial \mathbf{x}} = f_{\mathbf{q}}\mathbf{q}_{\mathbf{x}} + f_{\dot{\mathbf{q}}}\dot{\mathbf{q}}_{\mathbf{x}} + f_{\ddot{\mathbf{q}}}\ddot{\mathbf{q}}_{\mathbf{x}} + f_{\mathbf{x}} \tag{4.9}$$

being $\mathbf{x}$ any of the dependencies of $f$. This derivative involves all the derivatives of the function $f$ with respect to the natural coordinates as well as their derivatives with respect to the set of variables $\mathbf{x}$ being considered. Thus, for instance, the equivalent derivatives with respect to the positions, velocities and accelerations of the relative coordinates, and the derivatives with respect to the set of parameters of this function $f$ will be:

$$f_{\hat{\mathbf{z}}} = f_{\mathbf{q}}\mathbf{q}_{\mathbf{z}} + f_{\dot{\mathbf{q}}}\dot{\mathbf{q}}_{\mathbf{z}} + f_{\ddot{\mathbf{q}}}\ddot{\mathbf{q}}_{\mathbf{z}} + f_{\mathbf{z}} \tag{4.10a}$$

$$f_{\hat{\dot{\mathbf{z}}}} = f_{\mathbf{q}}\mathbf{q}_{\dot{\mathbf{z}}} + f_{\dot{\mathbf{q}}}\dot{\mathbf{q}}_{\dot{\mathbf{z}}} + f_{\ddot{\mathbf{q}}}\ddot{\mathbf{q}}_{\dot{\mathbf{z}}} + f_{\dot{\mathbf{z}}} = f_{\dot{\mathbf{q}}}\dot{\mathbf{q}}_{\dot{\mathbf{z}}} + f_{\ddot{\mathbf{q}}}\ddot{\mathbf{q}}_{\dot{\mathbf{z}}} + f_{\dot{\mathbf{z}}} \tag{4.10b}$$

$$f_{\hat{\ddot{\mathbf{z}}}} = f_{\mathbf{q}}\mathbf{q}_{\ddot{\mathbf{z}}} + f_{\dot{\mathbf{q}}}\dot{\mathbf{q}}_{\ddot{\mathbf{z}}} + f_{\ddot{\mathbf{q}}}\ddot{\mathbf{q}}_{\ddot{\mathbf{z}}} + f_{\ddot{\mathbf{z}}} = f_{\ddot{\mathbf{q}}}\ddot{\mathbf{q}}_{\ddot{\mathbf{z}}} + f_{\ddot{\mathbf{z}}} \tag{4.10c}$$

$$f_{\hat{\boldsymbol{\rho}}} = f_{\mathbf{q}}\mathbf{q}_{\boldsymbol{\rho}} + f_{\dot{\mathbf{q}}}\dot{\mathbf{q}}_{\boldsymbol{\rho}} + f_{\ddot{\mathbf{q}}}\ddot{\mathbf{q}}_{\boldsymbol{\rho}} + f_{\boldsymbol{\rho}} \tag{4.10d}$$

Observe that with this notation, the total derivative can be expressed as:

$$\frac{\mathrm{d}f}{\mathrm{d}\mathbf{x}} = f_{\hat{\mathbf{x}}} + f_{\hat{\mathbf{z}}}\mathbf{z}_{\mathbf{x}} + f_{\hat{\dot{\mathbf{z}}}}\dot{\mathbf{z}}_{\mathbf{x}} + f_{\hat{\ddot{\mathbf{z}}}}\ddot{\mathbf{z}}_{\mathbf{x}} + f_{\hat{\boldsymbol{\rho}}}\boldsymbol{\rho}_{\mathbf{x}} \tag{4.11}$$

The notation introduced in (4.9) can be extended to derivatives of vectors, matrices or tensors of any dimension. In this case, a tensor $\mathbf{F} \in \mathbb{R}^{n_1 \times n_2 \times \ldots \times n_r}$ of $r$ dimensions is considered. Let us define the derivative of this tensor in accordance with (4.9) as:

$$\mathbf{F}_{\hat{\mathbf{x}}} = \frac{\partial \mathbf{F}}{\partial \mathbf{q}}\frac{\partial \mathbf{q}}{\partial \mathbf{x}} + \frac{\partial \mathbf{F}}{\partial \dot{\mathbf{q}}}\frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{x}} + \frac{\partial \mathbf{F}}{\partial \ddot{\mathbf{q}}}\frac{\partial \ddot{\mathbf{q}}}{\partial \mathbf{x}} + \frac{\partial \mathbf{F}}{\partial \mathbf{x}} = \mathbf{F}_{\mathbf{q}}\mathbf{q}_{\mathbf{x}} + \mathbf{F}_{\dot{\mathbf{q}}}\dot{\mathbf{q}}_{\mathbf{x}} + \mathbf{F}_{\ddot{\mathbf{q}}}\ddot{\mathbf{q}}_{\mathbf{x}} + \mathbf{F}_{\mathbf{x}} \tag{4.12}$$

where the following tensor products appear: $\mathbf{F}_{\mathbf{q}}\mathbf{q}_{\mathbf{x}} = \mathbf{F}_{\mathbf{q}} \otimes_{r+1} \mathbf{q}_{\mathbf{x}}$, $\mathbf{F}_{\dot{\mathbf{q}}}\dot{\mathbf{q}}_{\mathbf{x}} = \mathbf{F}_{\dot{\mathbf{q}}} \otimes_{r+1} \dot{\mathbf{q}}_{\mathbf{x}}$ and $\mathbf{F}_{\ddot{\mathbf{q}}}\ddot{\mathbf{q}}_{\mathbf{x}} = \mathbf{F}_{\ddot{\mathbf{q}}} \otimes_{r+1} \ddot{\mathbf{q}}_{\mathbf{x}}$ [2].

Let us start here from the compact expression of the equations of motion (2.209). Considering a set of parameters $\boldsymbol{\rho} \in \mathbb{R}^p$ determining the dynamic behavior of a system, the effect of them in the equations of motion can be computed by differentiating (2.209) with respect to the set of parameters:

$$\left(\mathbf{M}^d_{\hat{\mathbf{z}}}\mathbf{z}' + \mathbf{M}^d_{\hat{\boldsymbol{\rho}}}\right)\ddot{\mathbf{z}} + \mathbf{M}^d\ddot{\mathbf{z}}' - \left(\mathbf{Q}^d_{\hat{\mathbf{z}}}\mathbf{z}' + \mathbf{Q}^d_{\hat{\dot{\mathbf{z}}}}\dot{\mathbf{z}}' + \mathbf{Q}^d_{\hat{\boldsymbol{\rho}}}\right) = \mathbf{0} \tag{4.13}$$

in which the following are tensor products: $\left(\mathbf{M}^d_{\hat{\mathbf{z}}}\mathbf{z}'\right)\ddot{\mathbf{z}} = \left(\mathbf{M}^d_{\hat{\mathbf{z}}} \otimes_3 \mathbf{z}'\right) \otimes_2 \ddot{\mathbf{z}}$ and $\mathbf{M}^d_{\hat{\boldsymbol{\rho}}}\ddot{\mathbf{z}} = \mathbf{M}^d_{\hat{\boldsymbol{\rho}}} \otimes_2 \ddot{\mathbf{z}}$.

Gathering terms,

$$\left(\mathbf{M}^d_{\hat{\mathbf{z}}}\ddot{\mathbf{z}} + \mathbf{K}\right)\mathbf{z}' + \mathbf{C}\dot{\mathbf{z}}' + \mathbf{M}^d\ddot{\mathbf{z}}' = \mathbf{Q}^d_{\hat{\boldsymbol{\rho}}} - \mathbf{M}^d_{\hat{\boldsymbol{\rho}}}\ddot{\mathbf{z}} \tag{4.14}$$

[2]Operator $\otimes_{r+1}$ represents a tensor product of mode $r+1$, i.e the product is executed on the $r+1$ dimension of the tensor.

wherein

$$\mathbf{K} = -\mathbf{Q}^d_{\hat{\mathbf{z}}} \tag{4.15a}$$

$$\mathbf{C} = -\mathbf{Q}^d_{\hat{\dot{\mathbf{z}}}} \tag{4.15b}$$

Matrices $\mathbf{K}$ and $\mathbf{C}$ are respectively the equivalent stiffness and the damping matrices of the mechanism. Similarly to subscript $\hat{\mathbf{z}}$, the new terms with the subscript $\hat{\boldsymbol{\rho}}$ gather derivatives with respect to $\boldsymbol{\rho}$ and with respect to natural coordinates.

The final expression obtained (4.14) is an ODE system, like the system of equations from which it has been derived. Consequently, it can be solved by means of any of the techniques used for this type of systems. In the current implementation of MBSLIM, the system is solved using a Newmark's family integrator, with the relative positions sensitivities, $\mathbf{z}'$, as the main variables.

The methodology used to solve the sensitivity analysis of the dynamics of a multibody system has to take into account the initialization process which involves the determination of positions, velocities and accelerations of the states along with their sensitivities at the initial instant of time. If joint coordinates are selected as degrees of freedom in unconstrained open-loop systems, there is no need to solve any initialization problem at position and velocity levels since the values of the degrees of freedom match the values of the coordinates vector. Accordingly, the sensitivities of the states at position and velocity levels are equal to the sensitivities of the degrees of freedom. Both initial accelerations and their sensitivities can be computed using the same set of equations valid for any other instant of time, which are (2.209) for the dynamics and (4.14) for the sensitivities.

The algorithm implemented in MBSLIM to evaluate the sensitivity of the unconstrained open-loop semi-recursive formulation is outlined below. For a given time step:

1. Dynamic analysis by means of (2.209).

2. Assessment of the derivatives of masses $\mathbf{M}^d_{\hat{\mathbf{z}}}$ and $\mathbf{M}^d_{\hat{\boldsymbol{\rho}}}$ and forces $\mathbf{K}$, $\mathbf{C}$ and $\mathbf{Q}^d_{\hat{\boldsymbol{\rho}}}$ with the expressions of sections 4.2.1 and 4.2.2.

3. Computation of the integrator correction terms. In the case of a Newmark's integrator, correction terms can be obtained from the extension of (2.223) to sensitivities:

$$\hat{\dot{\mathbf{z}}}'_n = -\left(\frac{\gamma}{\beta h}\mathbf{z}'_n + \left(\frac{\gamma}{\beta} - 1\right)\dot{\mathbf{z}}'_n + \left(\frac{\gamma}{2\beta} - 1\right)h\ddot{\mathbf{z}}'_n\right) \tag{4.16a}$$

$$\hat{\ddot{\mathbf{z}}}'_n = -\left(\frac{1}{\beta h^2}\mathbf{z}'_n + \frac{1}{\beta h}\dot{\mathbf{z}}'_n + \left(\frac{1}{2\beta} - 1\right)\ddot{\mathbf{z}}'_n\right) \tag{4.16b}$$

being $h$ the time step and $\gamma$ and $\beta$ two parameters of the integrators of the Newmark family. The equations of the implicit trapezoidal rule can be obtained as a particularization of the Newmark integrator with $\gamma = 0.5$ and $\beta = 0.25$.

4. Composition of the sensitivity system applying the numerical integrator to (4.14). In the case o the Newmark integrator:

$$\left(\mathbf{M}^d + \gamma h \mathbf{C}^d + \beta h^2 \left(\mathbf{M}^d_{\mathbf{z}} \ddot{\mathbf{z}} + \mathbf{K}^d\right)\right) \mathbf{z}' = \beta h^2 \left(\mathbf{Q}^d_{\hat{\boldsymbol{\rho}}} - \mathbf{M}^d_{\hat{\boldsymbol{\rho}}} \ddot{\mathbf{z}} - \mathbf{C}^d \hat{\dot{\mathbf{z}}}' - \mathbf{M}^d \hat{\ddot{\mathbf{z}}}'\right) \tag{4.17}$$

5. Solution of (4.17) for $\mathbf{z}'$.

6. Numerical differentiation of velocity and acceleration sensitivities with:

$$\dot{\mathbf{z}}'_{n+1} = \frac{\gamma}{\beta h} \mathbf{z}'_{n+1} + \hat{\dot{\mathbf{z}}}'_n \tag{4.18a}$$

$$\ddot{\mathbf{z}}'_{n+1} = \frac{1}{\beta h^2} \mathbf{z}'_{n+1} + \hat{\ddot{\mathbf{z}}}'_n \tag{4.18b}$$

7. Evaluation of the instant value $\dot{\boldsymbol{\psi}}'(t_i)$ of the objective function gradient in expression (4.7), using the sensitivities of the states already computed.

8. Numerical integration of the gradient. Considering the trapezoidal rule as time integrator, the integral (4.7) can be discretized by means of:

$$\boldsymbol{\psi}'(t_i) = \boldsymbol{\psi}'(t_{i-1}) + \frac{h}{2} \left(\dot{\boldsymbol{\psi}}'(t_{i-1}) + \dot{\boldsymbol{\psi}}'(t_i)\right) \tag{4.19}$$

Behold that (4.14) constitutes a set of $p$ systems of equations, all with the same leading matrix. Despite the fact that the leading matrix has to be factorized only once per time step, for a large number of parameters the direct differentiation method would be inefficient, and the use of the adjoint variable formulations, presented in chapter 5, is suggested. In that chapter, the adjoint method will be addressed for the sensitivity analysis of constrained multibody systems, and the developments presented there will be also applicable to the open-loop adjoint sensitivity equations. For the sake of brevity and clarity, the adjoint problem will be omitted for open-loop systems.

## 4.2.1 Semi-recursive mass matrix derivatives

As described in section 2.4.1, masses and inertia referred to the reference point of each body are accumulated from the tips of the mechanism to the base following the topology of the multibody model. The result is the general mass matrix of the mechanism, which depends on both the joint coordinates and on a set of parameters that remain constant during the dynamic simulation (masses, inertia referred to the local reference frame, local coordinates, etc.). Considering the implicit dependencies of the mass matrix in a given instant of time, its derivatives with respect to the parameters can be obtained as:

$$\frac{\mathrm{d}\mathbf{M}^d}{\mathrm{d}\boldsymbol{\rho}} = \mathbf{M}^d_{\mathbf{z}} \mathbf{z}' + \mathbf{M}^d_{\hat{\boldsymbol{\rho}}} \tag{4.20}$$

Observing (2.208b), the mass matrix can be regarded as the projection of $\mathbf{M}^v$ onto the joint-coordinates manifold. However, matrix products of the mass matrix by the topological matrix $\mathbf{R}$ are not really accomplished since a more efficient partial accumulation procedure (2.217) is employed instead. The derivative of $\mathbf{M}^d$ with respect to $\mathbf{z}$ or $\boldsymbol{\rho}$ using (4.9) can be obtained as the assembly of the derivatives of each of its internal blocks, with the form:

$$\left(\mathbf{M}^d(i,j)\right)_{\hat{\mathbf{x}}} = \left(\mathbf{b}_i^{v\mathrm{T}}\right)_{\hat{\mathbf{x}}} \mathbf{M}_i^\Sigma \mathbf{b}_{i,j}^v + \mathbf{b}_i^{v\mathrm{T}} \left(\mathbf{M}_i^\Sigma\right)_{\hat{\mathbf{x}}} \mathbf{b}_{i,j}^v + \mathbf{b}_i^{v\mathrm{T}} \mathbf{M}_i^\Sigma \left(\mathbf{b}_{i,j}^v\right)_{\hat{\mathbf{x}}}; \qquad i > j, \tag{4.21a}$$

$$\left(\mathbf{M}^d(i,i)\right)_{\hat{\mathbf{x}}} = \left(\mathbf{b}_i^{v\mathrm{T}}\right)_{\hat{\mathbf{x}}} \mathbf{M}_i^\Sigma \mathbf{b}_i^v + \mathbf{b}_i^{v\mathrm{T}} \left(\mathbf{M}_i^\Sigma\right)_{\hat{\mathbf{x}}} \mathbf{b}_i^v + \mathbf{b}_i^{v\mathrm{T}} \mathbf{M}_i^\Sigma \left(\mathbf{b}_i^v\right)_{\hat{\mathbf{x}}}, \tag{4.21b}$$

$$\left(\mathbf{M}^d(i,j)\right)_{\hat{\mathbf{x}}} = \left(\mathbf{b}_{j,i}^{v\mathrm{T}}\right)_{\hat{\mathbf{x}}} \mathbf{M}_i^\Sigma \mathbf{b}_i^v + \mathbf{b}_{j,i}^{v\mathrm{T}} \left(\mathbf{M}_i^\Sigma\right)_{\hat{\mathbf{x}}} \mathbf{b}_i^v + \mathbf{b}_{j,i}^{v\mathrm{T}} \mathbf{M}_i^\Sigma \left(\mathbf{b}_i^v\right)_{\hat{\mathbf{x}}}; \qquad i < j. \tag{4.21c}$$

with

$$\begin{aligned}\left(\mathbf{M}_i^\Sigma\right)_{\hat{\mathbf{x}}} &= \left(\mathbf{M}_i^v + \sum_{j=1}^{n_s^i} \left(\mathbf{B}_s^{v\mathrm{T}} \mathbf{M}_s^{v\Sigma} \mathbf{B}_s^v\right)\right)_{\hat{\mathbf{x}}} = \\ &= \left(\mathbf{M}_i^v\right)_{\hat{\mathbf{x}}} + \sum_{s=1}^{n_s^i} \left(\left(\mathbf{B}_s^{v\mathrm{T}}\right)_{\hat{\mathbf{x}}} \mathbf{M}_s^{v\Sigma} \mathbf{B}_s^v + \mathbf{B}_s^{v\mathrm{T}} \left(\mathbf{M}_s^{v\Sigma}\right)_{\hat{\mathbf{x}}} \mathbf{B}_s^v + \mathbf{B}_s^{v\mathrm{T}} \mathbf{M}_s^{v\Sigma} \left(\mathbf{B}_s^v\right)_{\hat{\mathbf{x}}}\right)\end{aligned} \tag{4.22}$$

being $\mathbf{x}$ the dependency considered (relative coordinates, $\mathbf{z}$, or parameters, $\boldsymbol{\rho}$), $n_s^i$ is the number of children of body $i$ (according to section 2.4.1), $\mathbf{B}_s^v \in \mathbb{R}^{6\times6}$ is the transformation matrix from body $s$ to body $s-1$ defined in (2.115), and

$$\left(\mathbf{b}_{i,i}^v\right)_{\hat{\mathbf{x}}} = \left(\mathbf{b}_i^v\right)_{\hat{\mathbf{x}}} \tag{4.23a}$$

$$\left(\mathbf{b}_{i,j}^v\right)_{\hat{\mathbf{x}}} = \left(\mathbf{B}_i^v\right)_{\hat{\mathbf{x}}} \mathbf{b}_{h,j}^v + \mathbf{B}_i^v \left(\mathbf{b}_{h,j}^v\right)_{\hat{\mathbf{x}}}; \; i > j \tag{4.23b}$$

$$\left(\mathbf{b}_{i,j}^v\right)_{\hat{\mathbf{x}}} = \mathbf{0}; \; i < j \tag{4.23c}$$

wherein $h$ is the parent body of $i$, i.e. the preceding body in the kinematic chain (see section 2.4.1), and the derivatives of $\mathbf{b}_i^v$ and $\mathbf{B}_i^v$ with respect to relative coordinates and parameters (the two possible dependencies of these terms), will be tackled in sections 4.4.3 and 4.4.6 for $\mathbf{b}_i^v$ and 4.4.8 and 4.4.11 for $\mathbf{B}_i^v$ respectively.

The derivative $\left(\mathbf{M}^d(i,j)\right)_{\hat{\mathbf{x}}}$ involves a set of matrix and tensor products that need to be computed at each time step, at each iteration and $n_j \times n_j$ times per iteration, being $n_j$ the number of kinematic joints. In fact, this brings about the derivative of $\mathbf{M}^d$ to be one of the most time consuming terms during the sensitivity analysis. For this reason, some simplifications should be applied.

Considering the symmetry of the general mass matrix $\mathbf{M}^d$, and the consequent symmetry with respect to the first and second dimension of its derivatives, it can be established that:

$$\left\{\left(\mathbf{M}^d\right)_{\hat{\mathbf{x}}}\right\}(i,j,k) = \left\{\left(\mathbf{M}^d\right)_{\hat{\mathbf{x}}}^{\mathrm{T}}\right\}(j,i,k) \tag{4.24}$$

being $j, i, k$ indices of joints. Using this relation, only $(n_j-1)\times n_j/2$ terms $\left\{\left(\mathbf{M}^d\right)_{\hat{\mathbf{x}}}\right\}_{i,j,k}$ need to be calculated, and the rest could be directly assembled with the transpose of the symmetric term.

From (4.21) and (4.22), it can be seen that the derivatives involved in the differentiation of the mass matrix can be gathered into derivatives of recursive kinematic relations $\left(\mathbf{b}^v_{i,j}\right)_{\hat{\mathbf{x}}}$, $\left(\mathbf{B}^v_s\right)_{\hat{\mathbf{x}}}$ and derivatives of elemental mass matrices. The derivative of the recursive kinematic terms will be set forth in section 4.4, while elemental mass matrices derivatives will be tackled here.

In a general formulation for any reference point, the elemental mass matrix of a body can be expressed as the product of a mass matrix referred to the CoM by a correction matrix $\mathbf{D}^v_i$, as it was introduced in (2.200). Applying the chain rule for differentiation, the tensor $\left(\mathbf{M}^v_i\right)_{\hat{\mathbf{z}}} \in \mathbb{R}^{6\times 6\times n}$ , result of the application of the differentiation rule explained in (4.9), to the elemental mass matrix expression (2.200) with respect to the relative coordinates, can be calculated as:

$$\left(\mathbf{M}^v_i\right)_{\hat{\mathbf{z}}} = m_i\left(\left(\mathbf{D}^{v\mathrm{T}}_i\right)_{\hat{\mathbf{z}}}\mathbf{D}^v_i + \left(\mathbf{D}^v_i\right)^{\mathrm{T}}\left(\mathbf{D}^v_i\right)_{\hat{\mathbf{z}}}\right) + \begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{0} & \left(\mathbf{J}^G_i\right)_{\hat{\mathbf{z}}}\end{bmatrix} \tag{4.25}$$

where

$$\left(\mathbf{J}^G_i\right)_{\hat{\mathbf{z}}} = \left(\mathbf{A}_i\right)_{\mathbf{z}}\bar{\mathbf{J}}^G_i\mathbf{A}^{\mathrm{T}}_i + \mathbf{A}_i\bar{\mathbf{J}}^G_i\left(\mathbf{A}^{\mathrm{T}}_i\right)_{\mathbf{z}} \tag{4.26a}$$

$$\left(\mathbf{D}^v_i\right)_{\hat{\mathbf{z}}} = \begin{bmatrix}\mathbf{0} & \dfrac{\partial\tilde{\mathbf{r}}_i}{\partial\mathbf{z}} - \dfrac{\partial\tilde{\mathbf{r}}^i_G}{\partial\mathbf{z}}\\ \mathbf{0} & \mathbf{0}\end{bmatrix} \tag{4.26b}$$

in which partial derivatives of rotation matrices will be described in section 4.4.1 and partial derivatives of points with respect to joint coordinates have been unfolded in section 3.6. Behold that, in the case of position derivatives of points and vectors, the differentiation rule $\left(\mathbf{r}_i\right)_{\hat{\mathbf{z}}}$ is equivalent to $\dfrac{\partial\mathbf{r}_i}{\partial\mathbf{z}}$, thus both notations will be considered along the document according to what is more descriptive in each case.

Observe in (4.25) that some simplifications are considered, as the decomposition of the mass matrix into mass and inertia terms, and the elimination of the derivatives of the terms not dependent on the relative coordinates.

Considering now the differentiation with respect to the parameters of the system, the tensor $\left(\mathbf{M}^v_i\right)_{\hat{\boldsymbol{\rho}}} \in \mathbb{R}^{6\times 6\times p}$ can be calculated as:

$$\left(\mathbf{M}^v_i\right)_{\hat{\boldsymbol{\rho}}} = \frac{\partial m_i}{\partial\boldsymbol{\rho}}\left(\left(\mathbf{D}^v_i\right)^{\mathrm{T}}\mathbf{D}^v_i\right) + m_i\left(\left(\mathbf{D}^{v\mathrm{T}}_i\right)_{\hat{\boldsymbol{\rho}}}\mathbf{D}^v_i + \left(\mathbf{D}^v_i\right)^{\mathrm{T}}\left(\mathbf{D}^v_i\right)_{\hat{\boldsymbol{\rho}}}\right) + \left(\begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{0} & \left(\mathbf{J}^G_i\right)_{\hat{\boldsymbol{\rho}}}\end{bmatrix}\right) \tag{4.27}$$

in which $\left(\mathbf{J}^G_i\right)_{\hat{\boldsymbol{\rho}}}$ is determined by differentiating (2.196):

$$\left(\mathbf{J}^G_i\right)_{\hat{\boldsymbol{\rho}}} = \left(\mathbf{A}_i\right)_{\boldsymbol{\rho}}\bar{\mathbf{J}}^G_i\mathbf{A}^{\mathrm{T}}_i + \mathbf{A}_i\left(\bar{\mathbf{J}}^G_i\right)_{\hat{\boldsymbol{\rho}}}\mathbf{A}^{\mathrm{T}}_i + \mathbf{A}_i\bar{\mathbf{J}}^G_i\left(\mathbf{A}^{\mathrm{T}}_i\right)_{\boldsymbol{\rho}} \tag{4.28}$$

with $\left(\mathbf{A}_i\right)_{\boldsymbol{\rho}}$ studied in section 4.4.2, and $\left(\mathbf{D}_i^v\right)_{\hat{\boldsymbol{\rho}}}$ can be reached by taking derivatives on the expression of $\mathbf{D}_i^v$ from (2.197):

$$\left(\mathbf{D}_i^v\right)_{\hat{\boldsymbol{\rho}}} = \begin{bmatrix} \mathbf{0} & \dfrac{\partial \tilde{\mathbf{r}}_i}{\partial \boldsymbol{\rho}} - \dfrac{\partial \tilde{\mathbf{r}}_G^i}{\partial \boldsymbol{\rho}} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{4.29}$$

wherein point position derivatives $\dfrac{\partial \mathbf{r}_i}{\partial \boldsymbol{\rho}}$ and $\dfrac{\partial \mathbf{r}_G^i}{\partial \boldsymbol{\rho}}$ will be addressed in section 4.5.7.

The evaluation of the mass matrix derivative can take different forms depending on the set of reference points selected for the accumulation process and for the description of the recursive kinematic relations. It should be pointed out that the mass matrix is independent of the set of reference points selected although, since the accumulation process relies on them, the composition of the mass matrix (and its derivatives) can differ. Considering the derivative of the mass matrix in the sense of the RTdyn0 formulation described in 2.4.3.1, with the CoM as reference point for each body, the following simplifications apply:

$$\mathbf{D}_i^y = \mathbf{I} \tag{4.30a}$$

$$\left(\mathbf{D}_i^y\right)_{\hat{\mathbf{z}}} = \mathbf{0} \tag{4.30b}$$

$$\left(\mathbf{D}_i^y\right)_{\hat{\boldsymbol{\rho}}} = \mathbf{0} \tag{4.30c}$$

The RTdyn1 approach, however, yields a different set of simplifications:

$$\mathbf{B}_i^z = \mathbf{I} \tag{4.31a}$$

$$\left(\mathbf{B}_i^z\right)_{\hat{\mathbf{z}}} = \mathbf{0} \tag{4.31b}$$

$$\left(\mathbf{B}_i^z\right)_{\hat{\boldsymbol{\rho}}} = \mathbf{0} \tag{4.31c}$$

Behold that the RTdyn0 approach simplifies the derivatives of each elemental mass matrix (4.25), while the RTdyn1 version increases the efficiency of the accumulation of (4.22). Although the simplifications delivered by the two approaches are similar, the efficient accumulation of RTdyn1 makes this approach less computationally demanding in this context.

The need to compute the derivative of the mass matrix in a sensitivity analysis constitutes one of biggest drawbacks of joint coordinate modeling when it is compared with models based on other set of coordinates, such as natural coordinates. A natural or fully-Cartesian coordinates model leads to a constant mass matrix [14], thus its derivative is null with respect to the states and constant with respect to the parameters. This fact could imply that joint coordinate modeling is less appropriate for sensitivity analysis than natural coordinate modeling, but the reduced number of coordinates and the efficiency of the dynamics can make this analytical sensitivity competitive in terms of computational time.

### 4.2.2 Semi-recursive generalized forces derivatives

Taking derivatives of the generalized forces vector with respect to relative coordinates or with respect to the system parameters is a particularly complex task in semi-recursive formulations due to the number of terms involved, the variety of forces and the fact that the elemental derivatives of each force are usually expressed in terms of natural coordinates, and they must be transformed to relative coordinates.

First of all, the elemental force vector for each body can be decomposed in 2 terms, one depending on each external force applied to the body and another one including the velocity dependent forces of the body related to its inertia. Recalling expression (2.201), the elemental force can be reformulated as:

$$\mathbf{Q}_i^v = \sum_{j=1}^{n_f^i} \mathbf{Q}_{i,j}^{v\ (e)} + \mathbf{Q}_i^{v(I)} \tag{4.32a}$$

$$\mathbf{Q}_{i,j}^{v\ (e)} = \begin{bmatrix} \mathbf{f}_j \\ \mathbf{n}_j^G + \left(\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_i\right)\mathbf{f}_j \end{bmatrix} \tag{4.32b}$$

$$\mathbf{Q}_i^{v(I)} = \begin{bmatrix} -m_i\tilde{\boldsymbol{\omega}}_i\left(\tilde{\boldsymbol{\omega}}_i\left(\mathbf{r}_G^i - \mathbf{r}_i\right)\right) \\ -\tilde{\boldsymbol{\omega}}_i\mathbf{J}_i^G\boldsymbol{\omega}_i - \left(\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_i\right)\left(m_i\tilde{\boldsymbol{\omega}}_i\left(\tilde{\boldsymbol{\omega}}_i\left(\mathbf{r}_G^i - \mathbf{r}_i\right)\right)\right) \end{bmatrix} \tag{4.32c}$$

being $n_f^i$ the number of external forces applied to the body $i$. Behold that vector products have been substituted by matrix products with the skew-symmetric operator $\tilde{(\cdot)}$ (see appendix B) in order to compact notation.

Thanks to this division, the terms related to each external force $\mathbf{f}_j$ can be evaluated separately from the inertial component of the elemental force, which only has to be calculated once per iteration for each body.

Finally, the previous terms have to be assembled in order to build the vector of generalized forces $\mathbf{Q}^d$:

$$\mathbf{Q}_i^d = \mathbf{R}_i^{v\mathrm{T}}\left(\mathbf{Q}^v - \mathbf{M}^v\dot{\mathbf{R}}^v\dot{\mathbf{z}}\right) = \mathbf{b}_i^{v\mathrm{T}}\mathbf{Q}_i^{v\Sigma}, \tag{4.33a}$$

$$\mathbf{Q}_i^{v\Sigma} = \mathbf{Q}_i^v - \mathbf{M}_i^v\mathbf{d}_i^{v\Sigma} + \sum_{s=1}^{n_s^i}\mathbf{B}_s^{v\mathrm{T}}\mathbf{Q}_s^{v\Sigma} \tag{4.33b}$$

where $n_s^i$ is again the number of children of body $i$.

#### 4.2.2.1 Stiffness matrix K

The equivalent stiffness matrix is defined as the derivative $\mathbf{K}_i = -\left(\mathbf{Q}_i^d\right)_{\hat{\mathbf{z}}}$, delivering a measure of the variation of the forces with the variation of the position of the mechanism.

It can be derived following the aforementioned procedure of division of the generalized forces vector. Taking derivatives on (4.33a) and (4.33b):

$$\mathbf{K}_i = -\left(\mathbf{Q}_i^d\right)_{\hat{\mathbf{z}}} = \mathbf{b}_i^{v\mathrm{T}}\mathbf{K}_i^{v\Sigma} - \left(\mathbf{b}_i^{v\mathrm{T}}\right)_{\hat{\mathbf{z}}}\mathbf{Q}_i^{v\Sigma}, \tag{4.34a}$$

$$\mathbf{K}_i^{v\Sigma} = -\left(\mathbf{Q}_i^{v\Sigma}\right)_{\hat{\mathbf{z}}} = \mathbf{K}_i{}^{v} + \left(\mathbf{M}_i^v\right)_{\hat{\mathbf{z}}}\mathbf{d}_i^{v\Sigma} + \mathbf{M}_i^v\left(\mathbf{d}_i^{v\Sigma}\right)_{\hat{\mathbf{z}}} + \sum_{s=1}^{n_s^i}\left(\mathbf{B}_s^{v\mathrm{T}}\mathbf{K}_s^{v\Sigma} - \left(\mathbf{B}_s^{v\mathrm{T}}\right)_{\hat{\mathbf{z}}}\mathbf{Q}_s^{v\Sigma}\right) \tag{4.34b}$$

where $n_s^i$ is again the number of children of body $i$. The derivative of the accumulated kinematic term $\mathbf{d}_i^{v\Sigma}$ involved in (4.34b) will be described in detail in section 4.4.13.

Before addressing the analytical expressions of the elemental force derivatives involved in (4.34b), it is convenient to introduce the derivatives of the position and velocity of a point and the angular velocity. The derivative of the position and velocity of a point and the derivative of the angular velocity of a body can be calculated making use of the recursive relations presented in (2.204). Decomposing the grouped term $\mathbf{V}$ into linear and angular velocities and particularizing it to a body $i$:

$$\mathbf{V}_i = \begin{bmatrix}\dot{\mathbf{r}}_i^v \\ \boldsymbol{\omega}_i\end{bmatrix} = \mathbf{R}_i^v\dot{\mathbf{z}} \tag{4.35}$$

and hence:

$$\dot{\mathbf{r}}_i^v = \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\mathbf{R}_i^v\dot{\mathbf{z}} \tag{4.36a}$$

$$\boldsymbol{\omega}_i = \begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix}\mathbf{R}_i^v\dot{\mathbf{z}} \tag{4.36b}$$

Using (4.36b), the derivatives of the angular velocity with respect to the relative coordinates in positions, can be easily obtained as:

$$\left(\boldsymbol{\omega}_i\right)_{\hat{\mathbf{z}}} = \begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix}\left(\mathbf{R}_i^v\right)_{\hat{\mathbf{z}}}\dot{\mathbf{z}} \tag{4.37}$$

Developing and expanding (4.37) (see appendix A), the final expression can be highly simplified avoiding the calculation of tensors of order 3:

$$\left(\boldsymbol{\omega}_i\right)_{\hat{\mathbf{z}}} = \begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix}\left(\mathbf{R}_i^v\right)_{\hat{\mathbf{z}}}\dot{\mathbf{z}} = \begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix}\dot{\mathbf{R}}_i^v - \begin{bmatrix}\mathbf{0} & \tilde{\boldsymbol{\omega}}_i\end{bmatrix}\mathbf{R}_i^v \tag{4.38}$$

Observe that expression(4.38) is valid for both the RTdyn0 and RTdyn1 EoM considering the corresponding $\mathbf{R}_i^y$ and $\dot{\mathbf{R}}_i^y$, or $\mathbf{R}_i^z$ and $\dot{\mathbf{R}}_i^z$ matrices instead of $\mathbf{R}_i^v$ and $\dot{\mathbf{R}}_i^v$. Behold also that $\dfrac{\partial \mathbf{R}_i^v}{\partial \mathbf{z}}\dot{\mathbf{z}} \neq \dot{\mathbf{R}}_i^v$. In the appendix A, the development of this derivative is accomplished.

Besides, the expressions for the derivatives of points and vectors considered in the current development are the ones already introduced in section 3.6.

The definition of a force is usually addressed in terms of positions and velocities of points, vectors, angles and distances. The two latest entities can be assimilated to joint-coordinates, but points and vectors require a special treatment, this is, a conversion to joint coordinates.

In order to simplify notation, let us first consider the derivative of a force and a torque with respect to the joint-coordinates vector by means of the differentiation rule (4.9):

$$\left(\mathbf{f}_j\right)_{\hat{\mathbf{z}}} = \frac{\partial \mathbf{f}_j}{\partial \mathbf{q}}\mathbf{q}_{\mathbf{z}} + \frac{\partial \mathbf{f}_j}{\partial \dot{\mathbf{q}}}\dot{\mathbf{q}}_{\mathbf{z}} + \frac{\partial \mathbf{f}_j}{\partial \mathbf{z}} \tag{4.39a}$$

$$\left(\mathbf{f}_j\right)_{\hat{\dot{\mathbf{z}}}} = \frac{\partial \mathbf{f}_j}{\partial \dot{\mathbf{q}}}\mathbf{q}_{\mathbf{z}} + \frac{\partial \mathbf{f}_j}{\partial \dot{\mathbf{z}}} \tag{4.39b}$$

$$\left(\mathbf{n}_j^G\right)_{\hat{\mathbf{z}}} = \frac{\partial \mathbf{n}_j^G}{\partial \mathbf{q}}\mathbf{q}_{\mathbf{z}} + \frac{\partial \mathbf{n}_j^G}{\partial \dot{\mathbf{q}}}\dot{\mathbf{q}}_{\mathbf{z}} + \frac{\partial \mathbf{n}_j^G}{\partial \mathbf{z}} \tag{4.39c}$$

$$\left(\mathbf{n}_j^G\right)_{\hat{\dot{\mathbf{z}}}} = \frac{\partial \mathbf{n}_j^G}{\partial \dot{\mathbf{q}}}\mathbf{q}_{\mathbf{z}} + \frac{\partial \mathbf{n}_j^G}{\partial \dot{\mathbf{z}}} \tag{4.39d}$$

where $\mathbf{q}$ and $\dot{\mathbf{q}}$ represent positions and velocities of the points and vectors in which force $\mathbf{f}_j$ depends. Behold that derivatives with respect to joint coordinates at position level involve the derivatives of the force with respect to velocities of points and vectors, whereas the derivative $\left(\mathbf{f}_j\right)_{\hat{\dot{\mathbf{z}}}}$ is more direct.

Once defined the elemental derivatives of all the terms appearing in (4.32), the derivative of $\mathbf{Q}_i^v$ with respect to $\mathbf{z}$ can be addressed. For the brevity, some intermediate operations are eliminated, and only the simplified result is presented.

$$\mathbf{K}_i{}^v = -\left(\mathbf{Q}_i^v\right)_{\hat{\mathbf{z}}} = \sum_{j=1}^{n_f^i} \mathbf{K}_{i,j}^v{}^{(e)} + \mathbf{K}_i^{v(I)} \tag{4.40a}$$

$$\mathbf{K}_{i,j}^v{}^{(e)} = \begin{bmatrix} -\left(\mathbf{f}_j\right)_{\hat{\mathbf{z}}} \\ \tilde{\mathbf{f}}_j\left(\left[\mathbf{I} \quad \tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_j\right]\mathbf{R}_i^v - \dfrac{\partial \mathbf{r}_i}{\partial \mathbf{z}}\right) - \left(\mathbf{n}_j^G\right)_{\hat{\mathbf{z}}} + \left(\tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_j\right)\left(\mathbf{f}_j\right)_{\hat{\mathbf{z}}} \end{bmatrix} \tag{4.40b}$$

$$\begin{aligned} \mathbf{K}_i^{v(I)} = & \begin{bmatrix} \mathbf{t}_{\hat{\mathbf{z}}} \\ \left(\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_i\right)\mathbf{t}_{\hat{\mathbf{z}}} - \tilde{\mathbf{t}}\left(\left[\mathbf{I} \quad \tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_G^i\right]\mathbf{R}_i^v - \dfrac{\partial \mathbf{r}_i}{\partial \mathbf{z}}\right) \end{bmatrix} \\ & + \begin{bmatrix} \mathbf{0} \\ \left(\tilde{\boldsymbol{\omega}}_i \mathbf{J}_i^G - \widetilde{\mathbf{J}_i^G \boldsymbol{\omega}_i}\right)\left(\boldsymbol{\omega}_i\right)_{\hat{\mathbf{z}}} + \tilde{\boldsymbol{\omega}}_i\left(\mathbf{J}_i^G\right)_{\hat{\mathbf{z}}}\boldsymbol{\omega}_i \end{bmatrix} \end{aligned} \tag{4.40c}$$

where

$$\mathbf{t} = m_i \tilde{\boldsymbol{\omega}}_i \left(\tilde{\boldsymbol{\omega}}_i \left(\mathbf{r}_G^i - \mathbf{r}_i\right)\right) \tag{4.41a}$$

$$\mathbf{t}_{\hat{\mathbf{z}}} = m_i \tilde{\boldsymbol{\omega}}_i \tilde{\boldsymbol{\omega}}_i \left(\left[\mathbf{I} \quad \tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_G^i\right]\mathbf{R}_i^v - \frac{\partial \mathbf{r}_i}{\partial \mathbf{z}}\right) - m_i\left(\tilde{\mathbf{h}} - \tilde{\boldsymbol{\omega}}_i\left(\tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_G^i\right)\right)\left(\boldsymbol{\omega}_i\right)_{\hat{\mathbf{z}}} \tag{4.41b}$$

$$\mathbf{h} = \tilde{\boldsymbol{\omega}}_i\left(\mathbf{r}_G^i - \mathbf{r}_i\right) \tag{4.41c}$$

Once calculated the elemental stiffness matrices for each body, they have to be assembled following the same scheme of the dynamics, already presented in (4.33a) and (4.33b). In the previous expressions, all the force-dependent terms are known, and the derivatives of the recursive kinematic terms $\left(\mathbf{b}_i^v\right)_{\hat{\mathbf{z}}}$, $\left(\mathbf{B}_s^{v\mathrm{T}}\right)_{\hat{\mathbf{z}}}$ and $\left(\mathbf{d}_i^{v\Sigma}\right)_{\hat{\mathbf{z}}}$ will be described in detail in sections 4.4.3, 4.4.8 and 4.4.13, respectively. The stiffness evaluation proposed involves elemental mass matrix derivatives, which could entail a repetition of calculations. Consequently, efficiency could be improved with a storage and reuse of these terms.

The stiffness matrix is particularly time demanding in semi-recursive methods. Similarly to the evaluation of the generalized forces vector, the particular versions RTdyn0 and RTdyn1 encompass a series of simplifications that imply a reduction in the stiffness matrix computational expense. First, let us study the RTdyn0 approach. As presented in section 2.4.3.1, the composition of the elemental generalized forces vector has the simplest possible form (2.249), thus the elemental stiffness matrix for each body becomes:

$$\mathbf{K}_i{}^y = -\left(\mathbf{Q}_i^d\right)_{\hat{\mathbf{z}}} = \sum_{j=1}^{n_f^i} \mathbf{K}_{i,j}^{y\ (e)} + \mathbf{K}_i^{y(I)} \tag{4.42a}$$

$$\mathbf{K}_{i,j}^{y\ (e)} = \begin{bmatrix} -\left(\mathbf{f}_j\right)_{\hat{\mathbf{z}}} \\ \tilde{\mathbf{f}}_j \begin{bmatrix}\mathbf{0} & \tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_j\end{bmatrix} \mathbf{R}_i^y - \left(\mathbf{n}_j^G\right)_{\hat{\mathbf{z}}} + \left(\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_j\right)\left(\mathbf{f}_j\right)_{\hat{\mathbf{z}}} \end{bmatrix} \tag{4.42b}$$

$$\mathbf{K}_i^{y(I)} = \begin{bmatrix} \mathbf{0} \\ \left(\tilde{\boldsymbol{\omega}}_i \mathbf{J}_i^G - \widetilde{\mathbf{J}_i^G \boldsymbol{\omega}_i}\right)\left(\boldsymbol{\omega}_i\right)_{\hat{\mathbf{z}}} + \tilde{\boldsymbol{\omega}}_i \left(\mathbf{J}_i^G\right)_{\hat{\mathbf{z}}} \boldsymbol{\omega}_i \end{bmatrix} \tag{4.42c}$$

Comparing (4.42) and (4.40), a significant reduction in the computational cost of the inertial stiffness term can be observed. The assembly equations of the general version of the stiffness matrix (4.34a) and (4.34b) are analog to the RTdyn0 version described by:

$$\mathbf{K}_i = \mathbf{b}_i^{y\mathrm{T}} \mathbf{K}_i^{y\Sigma} - \left(\mathbf{b}_i^{v\mathrm{T}}\right)_{\hat{\mathbf{z}}} \mathbf{Q}_i^{y\Sigma}, \tag{4.43a}$$

$$\mathbf{K}_i^{y\Sigma} = \mathbf{K}_i{}^y + \left(\mathbf{M}_i^y\right)_{\hat{\mathbf{z}}} \mathbf{d}_i^{y\Sigma} + \mathbf{M}_i^y \left(\mathbf{d}_i^{y\Sigma}\right)_{\hat{\mathbf{z}}} + \sum_{s=1}^{n_s^i} \left(\mathbf{B}_s^{y\mathrm{T}} \mathbf{K}_s^{y\Sigma} - \left(\mathbf{B}_s^{y\mathrm{T}}\right)_{\hat{\mathbf{z}}} \mathbf{Q}_s^{y\Sigma}\right) \tag{4.43b}$$

being $n_s^i$ is the number of children of body $i$.

In the RTdyn1 version, the stiffness matrix assessment is slightly different due to the particular type of point that is used as reference point. A reference point of a body coincident with the origin of coordinates at each time step has a series of properties:

- The position $\mathbf{r}_0^i$ in the global reference frame is fixed.
- The position $\bar{\mathbf{r}}_0^i$ in the local reference frame of body $i$ varies over time.

- Any derivative of the global position is always null.
- The assumptions for point derivatives presented in section 3.6 are invalid for this point type:

$$\frac{\partial \mathbf{r}_0^i}{\partial \mathbf{z}} \neq \frac{\partial \dot{\mathbf{r}}_0^i}{\partial \dot{\mathbf{z}}} \tag{4.44}$$

Taking derivatives on (2.261) following the scheme of division into external and inertial components of the elemental generalized forces vector introduced in (4.32), the elemental stiffness expressions for RTdyn1 become:

$$\mathbf{K}_i{}^z = -\left(\mathbf{Q}^d\right)_{\hat{\mathbf{z}}} = \sum_{j=1}^{n_f^i} \mathbf{K}_{i,j}^z{}^{(e)} + \mathbf{K}_i^{z(I)} \tag{4.45a}$$

$$\mathbf{K}_{i,j}^z{}^{(e)} = \begin{bmatrix} -\left(\mathbf{f}_j\right)_{\hat{\mathbf{z}}} \\ \tilde{\mathbf{f}}_j \left[\mathbf{I} \quad -\tilde{\mathbf{r}}_j\right] \mathbf{R}_i^z - \left(\mathbf{n}_j^G\right)_{\hat{\mathbf{z}}} - \tilde{\mathbf{r}}_j \left(\mathbf{f}_j\right)_{\hat{\mathbf{z}}} \end{bmatrix} \tag{4.45b}$$

$$\mathbf{K}_i^{z(I)} = \begin{bmatrix} \mathbf{t}_{\hat{\mathbf{z}}} \\ \tilde{\mathbf{r}}_G^i \mathbf{t}_{\hat{\mathbf{z}}} + \tilde{\mathbf{t}} \left[-\mathbf{I} \quad \tilde{\mathbf{r}}_G^i\right] \mathbf{R}_i^z + \left(\tilde{\boldsymbol{\omega}}_i \mathbf{J}_i^G - \widetilde{\mathbf{J}_i^G \boldsymbol{\omega}_i}\right) \left(\boldsymbol{\omega}_i\right)_{\hat{\mathbf{z}}} + \tilde{\boldsymbol{\omega}}_i \left(\mathbf{J}_i^G\right)_{\hat{\mathbf{z}}} \boldsymbol{\omega}_i \end{bmatrix} \tag{4.45c}$$

where

$$\mathbf{t} = m_i \tilde{\boldsymbol{\omega}}_i \left(\tilde{\boldsymbol{\omega}}_i \mathbf{r}_G^i\right) \tag{4.46a}$$

$$\mathbf{t}_{\hat{\mathbf{z}}} = m_i \tilde{\boldsymbol{\omega}}_i \tilde{\boldsymbol{\omega}}_i \left[\mathbf{I} \quad -\tilde{\mathbf{r}}_G^i\right] \mathbf{R}_i^v - m_i \left(\widetilde{\tilde{\boldsymbol{\omega}}_i \mathbf{r}_G^i} + \tilde{\boldsymbol{\omega}}_i \tilde{\mathbf{r}}_G^i\right) \left(\boldsymbol{\omega}_i\right)_{\hat{\mathbf{z}}} \tag{4.46b}$$

The application of the relation $\mathbf{B}_i^z = \mathbf{I}$, derived from the definition of the RTdyn1 approach, to the assembly equations of the stiffness matrix (4.34a) and (4.34b) brings about important simplifications:

$$\mathbf{K}_i = \mathbf{b}_i^{z\mathrm{T}} \mathbf{K}_i^{z\Sigma} - \left(\mathbf{b}_i^{z\mathrm{T}}\right)_{\hat{\mathbf{z}}} \mathbf{Q}_i^{z\Sigma}, \tag{4.47a}$$

$$\mathbf{K}_i^{z\Sigma} = \mathbf{K}_i{}^z + \left(\mathbf{M}_i^z\right)_{\hat{\mathbf{z}}} \mathbf{d}_i^{z\Sigma} + \mathbf{M}_i^z \left(\mathbf{d}_i^{z\Sigma}\right)_{\hat{\mathbf{z}}} + \sum_{s=1}^{n_s^i} \mathbf{K}_s^{z\Sigma} \tag{4.47b}$$

Both RTdyn0 and RTdyn1 approaches entail a series of particularizations that allow a significant reduction in the computational expense of the stiffness matrix evaluation comparing with the general expressions for any reference point (4.40), (4.34a) and (4.34b).

In addition to the different expressions reached for both versions, there are a set of forces that are worth to be considered separately: gravitational forces. For general multibody systems with constant mass and constant gravitational acceleration, partial derivatives of gravitational forces with respect to the position or velocity states of the model are null. However, the momentum in the reference point generated by the gravitational force applied in the CoM of a body can vary even though the force is constant since the length of the arm is different. If the reference point belongs to the body, which means that the point is fixed in the local reference frame of body $i$, then the length $\| \mathbf{r}_i - \mathbf{r}_G^i \|$ is constant. This is the case of RTdyn0, where even this distance becomes 0. On the contrary, in RTdyn1 the reference point is fixed in the global reference frame, but it changes its position in the local one, which causes that the arm of the force increases or decreases its length when the relative coordinates in positions change. The additional term required in RTdyn1 for gravitational forces can be readily obtained as a particularization of (4.45) with $\left(\mathbf{f}_j\right)_{\hat{\mathbf{z}}} = \mathbf{0}$ and $\left(\mathbf{n}_j^G\right)_{\hat{\mathbf{z}}} = \mathbf{0}$.

#### 4.2.2.2 Damping matrix C

The variation of the generalized forces vector with respect to joint-coordinate velocities can be gathered in a matrix $\mathbf{C} \in \mathbb{R}^{n \times n}$ equivalent to the damping matrix of structural problems. Hereinafter, for the sake of clearness, this matrix will be denoted as damping matrix.

The evaluation of the damping matrix will be firstly described for a general semi-recursive formulation with an arbitrary selection of reference points, and then the particular expressions for RTdyn0 and RTdyn1 will be presented.

Let us begin with the derivative of the angular velocity with respect to joint-coordinate velocities. Recalling the angular velocity recursive expression (4.36b), its derivative can be immediately determined as:

$$\left(\boldsymbol{\omega}_i\right)_{\hat{\dot{\mathbf{z}}}} = \begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix} \mathbf{R}_i^v \tag{4.48}$$

Besides, recalling the partial derivatives of forces and torques (4.39) and taking derivatives on (4.32) with respect to joint-coordinate velocities:

$$\mathbf{C}_i{}^v = -\left(\mathbf{Q}_i^d\right)_{\hat{\dot{\mathbf{z}}}} = \sum_{j=1}^{n_f^i} \mathbf{C}_{i,j}^{v}{}^{(e)} + \mathbf{C}_i^{v(I)} \tag{4.49a}$$

$$\mathbf{C}_{i,j}^{v}{}^{(e)} = -\begin{bmatrix}\mathbf{I} \\ \tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i\end{bmatrix}\left(\mathbf{f}_j\right)_{\hat{\dot{\mathbf{z}}}} - \begin{bmatrix}\mathbf{0} \\ \left(\mathbf{n}_j^G\right)_{\hat{\dot{\mathbf{z}}}}\end{bmatrix} \tag{4.49b}$$

$$\mathbf{C}_i^{v(I)} = \begin{bmatrix}\mathbf{0} & -m_i\left(\tilde{\mathbf{h}} - \tilde{\boldsymbol{\omega}}_i\left(\tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_G^i\right)\right) \\ \mathbf{0} & \tilde{\boldsymbol{\omega}}_i\mathbf{J}_i^G - \widetilde{\mathbf{J}_i^G\boldsymbol{\omega}_i} - m_i\left(\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_i\right)\left(\tilde{\mathbf{h}} - \tilde{\boldsymbol{\omega}}_i\left(\tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_G^i\right)\right)\end{bmatrix}\mathbf{R}_i^v \tag{4.49c}$$

with

$$\mathbf{h} = \tilde{\boldsymbol{\omega}}_i\left(\mathbf{r}_G^i - \mathbf{r}_i\right) \tag{4.50}$$

The elemental damping matrices of each body have now to be assembled in order to build the damping matrix of the mechanism. Recalling the scheme of forces assembly introduced in (4.33a) and (4.33b), the general damping matrix is obtained as:

$$\mathbf{C}_i = \mathbf{b}_i^{v\mathrm{T}} \mathbf{C}_i^{v\Sigma}, \tag{4.51a}$$

$$\mathbf{C}_i^{v\Sigma} = \mathbf{C}_i{}^v + \mathbf{M}_i^v \left(\mathbf{d}_i^{v\Sigma}\right)_{\hat{\mathbf{z}}} + \sum_{s=1}^{n_s^i} \mathbf{B}_s^{v\mathrm{T}} \mathbf{C}_s^{v\Sigma} \tag{4.51b}$$

being $n_s^i$ the number of children of body $i$. In this case, the unique recursive kinematic derivative required is $\left(\mathbf{d}_i^{v\Sigma}\right)_{\hat{\mathbf{z}}}$, which will be introduced in section 4.4.14.

Equations (4.49), (4.51a) and (4.51b) describe the damping matrix of the equations of motion of a system composed through a semi-recursive method with an arbitrary selection of reference points. However, efficiency can be improved for the two specific formulations reviewed in this document, RTdyn0 and RTdyn1.

Considering the CoM of each body as reference point (RTdyn0), (4.49) becomes:

$$\mathbf{C}_i{}^y = -\left(\mathbf{Q}_i^d\right)_{\hat{\mathbf{z}}} = \sum_{j=1}^{n_f^i} \mathbf{C}_{i,j}^{y\ (e)} + \mathbf{C}_i^{y(I)} \tag{4.52a}$$

$$\mathbf{C}_{i,j}^{y\ (e)} = -\begin{bmatrix} \mathbf{I} \\ \tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_G^i \end{bmatrix} \left(\mathbf{f}_j\right)_{\hat{\mathbf{z}}} - \begin{bmatrix} \mathbf{0} \\ \left(\mathbf{n}_j^G\right)_{\hat{\mathbf{z}}} \end{bmatrix} \tag{4.52b}$$

$$\mathbf{C}_i^{y(I)} = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \tilde{\boldsymbol{\omega}}_i \mathbf{J}_i^G - \widetilde{\mathbf{J}_i^G \boldsymbol{\omega}_i} \end{bmatrix} \mathbf{R}_i^y \tag{4.52c}$$

Analogously to what happens with the stiffness matrix, in RTdyn0 the resulting elemental damping matrices related to the inertial terms are notably simplified. The assembly procedure for this specific approach takes the form:

$$\mathbf{C}_i = \mathbf{b}_i^{y\mathrm{T}} \mathbf{C}_i^{y\Sigma}, \tag{4.53a}$$

$$\mathbf{C}_i^{y\Sigma} = \mathbf{C}_i{}^y + \mathbf{M}_i^y \left(\mathbf{d}_i^{y\Sigma}\right)_{\hat{\mathbf{z}}} + \sum_{s=1}^{n_s^i} \mathbf{B}_s^{y\mathrm{T}} \mathbf{C}_s^{y\Sigma} \tag{4.53b}$$

in which $n_s^i$ denotes the number of children of body $i$.

In the RTdyn1 approach, the elemental damping matrices take the form:

$$\mathbf{C}_i{}^z = -\left(\mathbf{Q}_i^d\right)_{\hat{\mathbf{z}}} = \sum_{j=1}^{n_f^i} \mathbf{C}_{i,j}^{z\ (e)} + \mathbf{C}_i^{z(I)} \tag{4.54a}$$

$$\mathbf{C}_{i,j}^{z\ (e)} = -\begin{bmatrix} \mathbf{I} \\ \tilde{\mathbf{r}}_j \end{bmatrix} \left(\mathbf{f}_j\right)_{\hat{\mathbf{z}}} - \begin{bmatrix} \mathbf{0} \\ \left(\mathbf{n}_j^G\right)_{\hat{\mathbf{z}}} \end{bmatrix} \tag{4.54b}$$

$$\mathbf{C}_i^{z(I)} = \begin{bmatrix} \mathbf{0} & -m_i \left(\widetilde{\tilde{\boldsymbol{\omega}}_i \mathbf{r}_G^i} + \tilde{\boldsymbol{\omega}}_i \tilde{\mathbf{r}}_G^i\right) \\ \mathbf{0} & \tilde{\boldsymbol{\omega}}_i \mathbf{J}_i^G - \widetilde{\mathbf{J}_i^G \boldsymbol{\omega}_i} - m_i \tilde{\mathbf{r}}_G^i \left(\widetilde{\tilde{\boldsymbol{\omega}}_i \mathbf{r}_G^i} + \tilde{\boldsymbol{\omega}}_i \tilde{\mathbf{r}}_G^i\right) \end{bmatrix} \mathbf{R}_i^z \tag{4.54c}$$

In this case, the assembly is also simplified:

$$\mathbf{C}_i = \mathbf{b}_i^{z\mathrm{T}} \mathbf{C}_i^{z\Sigma}, \tag{4.55a}$$

$$\mathbf{C}_i^{z\Sigma} = \mathbf{C}_i{}^z + \mathbf{M}_i^z \left( \mathbf{d}_i^{z\Sigma} \right)_{\hat{\mathbf{z}}} + \sum_{s=1}^{n_s^i} \mathbf{C}_s^{z\Sigma} \tag{4.55b}$$

The enhancement in the computational effort required to evaluate the damping matrix in RTdyn0 or RTdyn1 approaches highly depends on the model, and it is difficult to determine which approach is more efficient for any multibody model. Despite the important simplification on the assembly process brought by the RTdyn1 formulation, its higher complexity in the evaluation of elemental damping matrices referred to each body could entail a loss of efficiency compared to the RTdyn0 approach. In general, numerical results have proved both RTdyn0 and RTdyn1 versions to yield similar results in terms of computational time for damping matrix assessments, being the RTdyn0 accumulation method slightly more efficient.

#### 4.2.2.3 Evaluation of $\mathbf{Q}^d_{\hat{\boldsymbol{\rho}}}$

Partial derivatives with respect to any set of parameters are conditioned by the type of parameters and the type of forces considered. Moreover, the computational effort is strongly affected by the type of parameters selected, depending on whether they affect or not the topology of the mechanism, this is, the sequence, type and definition of each joint.

If a parameter is selected such as it only affects a coefficient of a force, for example, only the partial derivative of this force and its assembly will be needed for the evaluation of $\mathbf{Q}^d_{\hat{\boldsymbol{\rho}}}$. The opposite case consists in the selection of a local coordinate of a point or vector as parameter defining one joint, in which case the evaluation of the derivatives of each one of the points and vectors used to compute each force will be required , but also the derivatives of the assembly itself, involving the accumulation of derivatives of the joint-dependent terms, such as $\mathbf{b}_i^v$ or $\mathbf{B}_i^v$, for instance. Furthermore, the derivatives of the inertial terms of the forces will be required as well, entailing an important computational effort, specially in RTdyn1 formulations.

Similarly to (4.39), notation can be clarified by using the compact notation presented at the beginning of section 4.2. Thus, consider the derivative of a force and a torque with respect to a parameters array $\boldsymbol{\rho} \in \mathbb{R}^p$ by means of the differentiation rule (4.9):

$$\left( \mathbf{f}_j \right)_{\hat{\boldsymbol{\rho}}} = \frac{\partial \mathbf{f}_j}{\partial \mathbf{q}} \mathbf{q}_{\boldsymbol{\rho}} + \frac{\partial \mathbf{f}_j}{\partial \dot{\mathbf{q}}} \dot{\mathbf{q}}_{\boldsymbol{\rho}} + \frac{\partial \mathbf{f}_j}{\partial \boldsymbol{\rho}} \tag{4.56a}$$

$$\left( \mathbf{n}_j^G \right)_{\hat{\boldsymbol{\rho}}} = \frac{\partial \mathbf{n}_j^G}{\partial \mathbf{q}} \mathbf{q}_{\boldsymbol{\rho}} + \frac{\partial \mathbf{n}_j^G}{\partial \dot{\mathbf{q}}} \dot{\mathbf{q}}_{\boldsymbol{\rho}} + \frac{\partial \mathbf{n}_j^G}{\partial \boldsymbol{\rho}} \tag{4.56b}$$

Applying now the rule of differentiation defined in (4.9) to the generalized forces vector expressions (4.32):

$$\left(\mathbf{Q}_i^d\right)_{\hat{\boldsymbol{\rho}}} = \sum_{j=1}^{n_f^i} \left(\mathbf{Q}_{i,j}^{v\ (e)}\right)_{\hat{\boldsymbol{\rho}}} + \left(\mathbf{Q}_i^{v(I)}\right)_{\hat{\boldsymbol{\rho}}} \tag{4.57a}$$

$$\left(\mathbf{Q}_{i,j}^{v\ (e)}\right)_{\hat{\boldsymbol{\rho}}} = \begin{bmatrix} \left(\mathbf{f}_j\right)_{\hat{\boldsymbol{\rho}}} \\ \tilde{\mathbf{f}}_j \left(\dfrac{\partial \mathbf{r}_j}{\partial \boldsymbol{\rho}} - \dfrac{\partial \mathbf{r}_i}{\partial \boldsymbol{\rho}}\right) + \left(\mathbf{n}_j^G\right)_{\hat{\boldsymbol{\rho}}} + \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i\right)\left(\mathbf{f}_j\right)_{\hat{\boldsymbol{\rho}}} \end{bmatrix} \tag{4.57b}$$

$$\left(\mathbf{Q}_i^{v(I)}\right)_{\hat{\boldsymbol{\rho}}} = \begin{bmatrix} \mathbf{t}_{\hat{\boldsymbol{\rho}}} \\ \left(\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_i\right)\mathbf{t}_{\hat{\boldsymbol{\rho}}} + \tilde{\mathbf{t}}\left(\dfrac{\partial \mathbf{r}_i}{\partial \boldsymbol{\rho}} - \dfrac{\partial \mathbf{r}_G^i}{\partial \boldsymbol{\rho}}\right) - \left(\tilde{\boldsymbol{\omega}}_i \mathbf{J}_i^G - \widetilde{\mathbf{J}_i^G \boldsymbol{\omega}_i}\right)\left(\boldsymbol{\omega}_i\right)_{\hat{\boldsymbol{\rho}}} - \tilde{\boldsymbol{\omega}}_i \left(\mathbf{J}_i^G\right)_{\hat{\boldsymbol{\rho}}} \boldsymbol{\omega}_i \end{bmatrix} \tag{4.57c}$$

where

$$\mathbf{t} = -m_i \tilde{\boldsymbol{\omega}}_i \left(\tilde{\boldsymbol{\omega}}_i \left(\mathbf{r}_G^i - \mathbf{r}_i\right)\right) \tag{4.58a}$$

$$\mathbf{t}_{\hat{\boldsymbol{\rho}}} = -m_i \tilde{\boldsymbol{\omega}}_i \tilde{\boldsymbol{\omega}}_i \left(\frac{\partial \mathbf{r}_G^i}{\partial \boldsymbol{\rho}} - \frac{\partial \mathbf{r}_i}{\partial \boldsymbol{\rho}}\right) - m_i \left(\tilde{\mathbf{h}} - \tilde{\boldsymbol{\omega}}_i \left(\tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_G^i\right)\right)\left(\boldsymbol{\omega}_i\right)_{\hat{\boldsymbol{\rho}}} \tag{4.58b}$$

$$\mathbf{h} = \tilde{\boldsymbol{\omega}}_i \left(\mathbf{r}_G^i - \mathbf{r}_i\right) \tag{4.58c}$$

Partial derivatives of forces can take different expressions depending on the type of parameter or force. Nevertheless, partial derivatives of the position of points $j$ and $i$, $\dfrac{\partial \mathbf{r}_j}{\partial \boldsymbol{\rho}}$ and $\dfrac{\partial \mathbf{r}_i}{\partial \boldsymbol{\rho}}$, with respect to the sensitivity parameters can be determined following the topology of the system, and regarding that the unique set of parameters that can influence the position of a point are local coordinates of points and vectors affecting the topology or the own point definition. These cases will be studied in detail in section 4.5.7.

Furthermore, derivative $\left(\mathbf{J}_i^G\right)_{\hat{\boldsymbol{\rho}}}$ can be directly determined from (4.28), and $\left(\boldsymbol{\omega}_i\right)_{\hat{\boldsymbol{\rho}}}$ can be calculated taking derivatives on the semi-recursive expression (4.36b):

$$\left(\boldsymbol{\omega}_i\right)_{\hat{\boldsymbol{\rho}}} = \begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix} \left(\mathbf{R}_i^v\right)_{\hat{\boldsymbol{\rho}}} \dot{\mathbf{z}} \tag{4.59}$$

Regarding the composition of the topological matrix $\mathbf{R}$ in terms of elemental blocks (2.212), its derivative can be directly expressed in terms of partial derivatives of the kinematic recursive terms $\mathbf{b}_i^v$ and $\mathbf{B}_i^v$:

$$\left(\mathbf{b}_{i,i}^v\right)_{\hat{\boldsymbol{\rho}}} = \left(\mathbf{b}_i^v\right)_{\hat{\boldsymbol{\rho}}} \tag{4.60a}$$

$$\left(\mathbf{b}_{i,j}^v\right)_{\hat{\boldsymbol{\rho}}} = \left(\mathbf{B}_i^v\right)_{\hat{\boldsymbol{\rho}}} \mathbf{b}_{h,j}^v + \mathbf{B}_i^v \left(\mathbf{b}_{h,j}^v\right)_{\hat{\boldsymbol{\rho}}};\ i > j \tag{4.60b}$$

$$\left(\mathbf{b}_{i,j}^v\right)_{\hat{\boldsymbol{\rho}}} = \mathbf{0};\ i < j \tag{4.60c}$$

in which derivatives $\left(\mathbf{b}_i^v\right)_{\hat{\boldsymbol{\rho}}}$ and $\left(\mathbf{B}_i^v\right)_{\hat{\boldsymbol{\rho}}}$ will be respectively described in sections 4.4.6 and 4.4.11.

Following the assembly of the dynamics, presented in (4.33a) and (4.33b), the complete derivative of the generalized forces vector takes the form:

$$\left(\mathbf{Q}_i^d\right)_{\hat{\boldsymbol{\rho}}} = \mathbf{b}_i^{v\mathrm{T}}\left(\mathbf{Q}_i^{v\Sigma}\right)_{\hat{\boldsymbol{\rho}}} + \left(\mathbf{b}_i^{v\mathrm{T}}\right)_{\hat{\boldsymbol{\rho}}}\mathbf{Q}_i^{v\Sigma}, \tag{4.61a}$$

$$\left(\mathbf{Q}_i^{v\Sigma}\right)_{\hat{\boldsymbol{\rho}}} = \left(\mathbf{Q}_i^{v}\right)_{\hat{\boldsymbol{\rho}}} + \left(\mathbf{M}_i^{v}\right)_{\hat{\boldsymbol{\rho}}}\mathbf{d}_i^{v\Sigma} + \mathbf{M}_i^{v}\left(\mathbf{d}_i^{v\Sigma}\right)_{\hat{\boldsymbol{\rho}}} + \sum_{s=1}^{n_s^i}\left(\mathbf{B}_s^{v\mathrm{T}}\left(\mathbf{Q}_s^{v\Sigma}\right)_{\hat{\boldsymbol{\rho}}} + \left(\mathbf{B}_s^{v\mathrm{T}}\right)_{\hat{\boldsymbol{\rho}}}\mathbf{Q}_s^{v\Sigma}\right) \tag{4.61b}$$

Comparing this derivative with the expressions of the stiffness matrix, several analogies can be observed, both in the elemental derivatives and in the assembly. Harnessing these similarities, the structure of computation of the stiffness matrix can be easily extended or reused to compute the derivatives with respect to any parameter, and therefore, the implementation effort can be significantly reduced.

Due to the wide range of types of parameters that could determine a dynamic simulation, a semi-analytical approach may be convenient in some cases, i.e. to compute derivatives with respect to parameters by means of numerical methods or automatic differentiation. In the MBSLIM implementation, analytical methods are the preferred option and only a few derivatives related to splines have been programmed with automatic differentiation.

## 4.3 Forward sensitivity of fully-recursive EoM for open-loop systems

The fully-recursive approach for the dynamics solution of open-loop systems with no additional constraints is specially well suited for large concatenations of bodies in a single chain. In that case, if the semi-recursive accumulation is used, the resulting mass matrix will be fully dense, and with a rank equal to the number of joint coordinates. On the contrary, the fully-recursive approach dodges the assembly and factorization of this matrix by the recursive accumulation and dynamic evaluation of each joint separately.

The advantages of this formulation of the EoM can be extended to sensitivity analysis, where there is no need to differentiate matrices of the size of the system, but only elemental terms related to each body and each joint.

The general expressions (2.247) including the possibility of forces applied directly on joint coordinates are used for the development of the sensitivity expressions. The application of the direct differentiation method to (2.247) has to follow the dynamics solution procedure. Remember that recursive dynamic formulations involve three stages (see stages 2 to 4 of flowchart 2.12), one related to the forward evaluation of the kinematics, a second stage for the backward accumulation of masses and forces, and a third one for the forward solution of the equations of motion. In this case, the kinematic derivatives do not have to be assembled recursively but on demand, and only the second and third stages have to be differentiated. Therefore, the forward fully-recursive sensitivity formulation can be divided in 2 phases:

1. **Accumulation of forces and masses, from the leaves to the root.**

   Firstly, the masses and forces must be accumulated from the leaves of the mechanism to the base, incorporating the effects of these forces and inertia in each of the bodies of the kinematic chain. Taking derivatives in (2.247), the following accumulation of sensitivities is obtained:

$$\begin{aligned}\left(\hat{\mathbf{M}}^v_{i-1}\right)_{\hat{\mathbf{z}}} = \left(\mathbf{M}^v_{i-1}\right)_{\hat{\mathbf{z}}} + \left(\mathbf{B}^{v\mathrm{T}}_i\right)_{\hat{\mathbf{z}}}\mathbf{K}_i\hat{\mathbf{M}}^v_i\mathbf{B}^v_i + \mathbf{B}^{v\mathrm{T}}_i\left(\mathbf{K}_i\right)_{\hat{\mathbf{z}}}\hat{\mathbf{M}}^v_i\mathbf{B}^v_i \\ +\mathbf{B}^{v\mathrm{T}}_i\mathbf{K}_i\left(\hat{\mathbf{M}}^v_i\right)_{\hat{\mathbf{z}}}\mathbf{B}^v_i + \mathbf{B}^{v\mathrm{T}}_i\mathbf{K}_i\hat{\mathbf{M}}^v_i\left(\mathbf{B}^v_i\right)_{\hat{\mathbf{z}}}\end{aligned} \tag{4.62a}$$

$$\begin{aligned}\left(\hat{\mathbf{Q}}^v_{i-1}\right)_{\hat{\mathbf{z}}} = \left(\mathbf{Q}^v_{i-1}\right)_{\hat{\mathbf{z}}} + \left(\mathbf{B}^{v\mathrm{T}}_i\right)_{\hat{\mathbf{z}}}\left(\mathbf{K}_i\left(\hat{\mathbf{Q}}^v_i - \hat{\mathbf{M}}^v_i\mathbf{d}^v_i\right) + \mathbf{K}^{\mathbf{z}}_i\mathbf{Q}^{\mathbf{z}}_i\right) \\ +\mathbf{B}^{v\mathrm{T}}_i\left(\mathbf{K}_i\right)_{\hat{\mathbf{z}}}\left(\hat{\mathbf{Q}}^v_i - \hat{\mathbf{M}}^v_i\mathbf{d}^v_i\right) + \mathbf{B}^{v\mathrm{T}}_i\left(\mathbf{K}^{\mathbf{z}}_i\right)_{\hat{\mathbf{z}}}\mathbf{Q}^{\mathbf{z}}_i \\ +\mathbf{B}^{v\mathrm{T}}_i\left(\mathbf{K}_i\left(\left(\hat{\mathbf{Q}}^v_i\right)_{\hat{\mathbf{z}}} - \left(\hat{\mathbf{M}}^v_i\right)_{\hat{\mathbf{z}}}\mathbf{d}^v_i - \hat{\mathbf{M}}^v_i\left(\mathbf{d}^v_i\right)_{\hat{\mathbf{z}}}\right) + \mathbf{K}^{\mathbf{z}}_i\left(\mathbf{Q}^{\mathbf{z}}_i\right)_{\hat{\mathbf{z}}}\right)\end{aligned} \tag{4.62b}$$

   being the derivatives of the accumulation matrices:

$$\left(\mathbf{K}_i\right)_{\hat{\mathbf{z}}} = \left(\mathbf{K}^{\mathbf{z}}_i\right)_{\hat{\mathbf{z}}}\mathbf{b}^{v\mathrm{T}}_i + \mathbf{K}^{\mathbf{z}}_i\left(\mathbf{b}^{v\mathrm{T}}_i\right)_{\hat{\mathbf{z}}} \tag{4.63a}$$

$$\begin{aligned}\left(\mathbf{K}^{\mathbf{z}}_i\right)_{\hat{\mathbf{z}}} = -\left(\hat{\mathbf{M}}^v_i\right)_{\hat{\mathbf{z}}}\mathbf{b}^v_i\left[\mathbf{b}^{v\mathrm{T}}_i\hat{\mathbf{M}}^v_i\mathbf{b}^v_i\right]^{-1} - \hat{\mathbf{M}}^v_i\left(\mathbf{b}^v_i\right)_{\hat{\mathbf{z}}}\left[\mathbf{b}^{v\mathrm{T}}_i\hat{\mathbf{M}}^v_i\mathbf{b}^v_i\right]^{-1} \\ -\hat{\mathbf{M}}^v_i\mathbf{b}^v_i\left(-\left[\mathbf{b}^{v\mathrm{T}}_i\hat{\mathbf{M}}^v_i\mathbf{b}^v_i\right]^{-1}\left\{\left(\mathbf{b}^{v\mathrm{T}}_i\right)_{\hat{\mathbf{z}}}\hat{\mathbf{M}}^v_i\mathbf{b}^v_i + \mathbf{b}^{v\mathrm{T}}_i\left(\hat{\mathbf{M}}^v_i\right)_{\hat{\mathbf{z}}}\mathbf{b}^v_i\right.\right. \\ \left.\left. +\mathbf{b}^{v\mathrm{T}}_i\hat{\mathbf{M}}^v_i\left(\mathbf{b}^v_i\right)_{\hat{\mathbf{z}}}\right\}\left[\mathbf{b}^{v\mathrm{T}}_i\hat{\mathbf{M}}^v_i\mathbf{b}^v_i\right]^{-1}\right)\end{aligned} \tag{4.63b}$$

   Regrouping terms on (4.63b) and identifying recursive expressions from (2.247), $\left(\mathbf{K}^{\mathbf{z}}_i\right)_{\hat{\mathbf{z}}}$ can be reformulated as:

$$\left(\mathbf{K}^{\mathbf{z}}_i\right)_{\hat{\mathbf{z}}} = -\left\{\mathbf{K}^{\mathbf{z}}_i\left(\mathbf{b}^{v\mathrm{T}}_i\right)_{\hat{\mathbf{z}}}\hat{\mathbf{M}}^v_i\mathbf{b}^v_i + \mathbf{K}_i\left(\hat{\mathbf{M}}^v_i\right)_{\hat{\mathbf{z}}}\mathbf{b}^v_i + \mathbf{K}_i\hat{\mathbf{M}}^v_i\left(\mathbf{b}^v_i\right)_{\hat{\mathbf{z}}}\right\}\left[\mathbf{b}^{v\mathrm{T}}_i\hat{\mathbf{M}}^v_i\mathbf{b}^v_i\right]^{-1} \tag{4.64}$$

   The derivatives with respect to velocities yield:

$$\left(\hat{\mathbf{M}}^v_{i-1}\right)_{\dot{\hat{\mathbf{z}}}} = \mathbf{0} \tag{4.65a}$$

$$\left(\hat{\mathbf{Q}}^v_{i-1}\right)_{\dot{\hat{\mathbf{z}}}} = \left(\mathbf{Q}^v_{i-1}\right)_{\dot{\hat{\mathbf{z}}}} + \mathbf{B}^{v\mathrm{T}}_i\left(\mathbf{K}_i\left(\left(\hat{\mathbf{Q}}^v_i\right)_{\dot{\hat{\mathbf{z}}}} - \hat{\mathbf{M}}^v_i\left(\mathbf{d}^v_i\right)_{\dot{\hat{\mathbf{z}}}}\right) + \mathbf{K}^{\mathbf{z}}_i\left(\mathbf{Q}^{\mathbf{z}}_i\right)_{\dot{\hat{\mathbf{z}}}}\right) \tag{4.65b}$$

$$\left(\mathbf{K}_i\right)_{\dot{\hat{\mathbf{z}}}} = \mathbf{0} \tag{4.65c}$$

The accumulated masses and forces are independent of the accelerations of the states, so these derivatives are always zero.

Observe that all the terms are presented directly after differentiation (except $\left(\mathbf{K}_i^{\mathbf{z}}\right)_{\hat{\mathbf{z}}}$ in (4.64)), but the expressions can be further simplified regarding the symmetry and the internal structure of the matrices, and also by grouping and reusing terms.

Considering that since a parameter could be any mass, inertia, local coordinate or any force coefficient, the derivative of the accumulated masses and forces with respect to them can be obtained as:

$$\begin{aligned}\left(\hat{\mathbf{M}}_{i-1}^v\right)_{\hat{\boldsymbol{\rho}}} = \left(\mathbf{M}_{i-1}^v\right)_{\hat{\boldsymbol{\rho}}} + \left(\mathbf{B}_i^{v\mathrm{T}}\right)_{\hat{\boldsymbol{\rho}}}\mathbf{K}_i\hat{\mathbf{M}}_i^v\mathbf{B}_i^v + \mathbf{B}_i^{v\mathrm{T}}\left(\mathbf{K}_i\right)_{\hat{\boldsymbol{\rho}}}\hat{\mathbf{M}}_i^v\mathbf{B}_i^v \\ +\mathbf{B}_i^{v\mathrm{T}}\mathbf{K}_i\left(\hat{\mathbf{M}}_i^v\right)_{\hat{\boldsymbol{\rho}}}\mathbf{B}_i^v + \mathbf{B}_i^{v\mathrm{T}}\mathbf{K}_i\hat{\mathbf{M}}_i^v\left(\mathbf{B}_i^v\right)_{\hat{\boldsymbol{\rho}}}\end{aligned} \tag{4.66a}$$

$$\begin{aligned}\left(\hat{\mathbf{Q}}_{i-1}^v\right)_{\hat{\boldsymbol{\rho}}} = \left(\mathbf{Q}_{i-1}^v\right)_{\hat{\boldsymbol{\rho}}} + \left(\mathbf{B}_i^{v\mathrm{T}}\right)_{\hat{\boldsymbol{\rho}}}\left(\mathbf{K}_i\left(\hat{\mathbf{Q}}_i^v - \hat{\mathbf{M}}_i^v\mathbf{d}_i^v\right) + \mathbf{K}_i^{\mathbf{z}}\mathbf{Q}_i^{\mathbf{z}}\right) \\ +\mathbf{B}_i^{v\mathrm{T}}\left(\mathbf{K}_i\right)_{\hat{\boldsymbol{\rho}}}\left(\hat{\mathbf{Q}}_i^v - \hat{\mathbf{M}}_i^v\mathbf{d}_i^v\right) + \mathbf{B}_i^{v\mathrm{T}}\left(\mathbf{K}_i^{\mathbf{z}}\right)_{\hat{\boldsymbol{\rho}}}\mathbf{Q}_i^{\mathbf{z}} \\ +\mathbf{B}_i^{v\mathrm{T}}\left(\mathbf{K}_i\left(\left(\hat{\mathbf{Q}}_i^v\right)_{\hat{\boldsymbol{\rho}}} - \left(\hat{\mathbf{M}}_i^v\right)_{\hat{\boldsymbol{\rho}}}\mathbf{d}_i^v - \hat{\mathbf{M}}_i^v\left(\mathbf{d}_i^v\right)_{\hat{\boldsymbol{\rho}}}\right) + \mathbf{K}_i^{\mathbf{z}}\left(\mathbf{Q}_i^{\mathbf{z}}\right)_{\hat{\boldsymbol{\rho}}}\right)\end{aligned} \tag{4.66b}$$

wherein the derivatives of the accumulation matrices take the form:

$$\left(\mathbf{K}_i\right)_{\hat{\boldsymbol{\rho}}} = \left(\mathbf{K}_i^{\mathbf{z}}\right)_{\hat{\boldsymbol{\rho}}}\mathbf{b}_i^{v\mathrm{T}} + \mathbf{K}_i^{\mathbf{z}}\left(\mathbf{b}_i^{v\mathrm{T}}\right)_{\hat{\boldsymbol{\rho}}} \tag{4.67a}$$

$$\begin{aligned}\left(\mathbf{K}_i^{\mathbf{z}}\right)_{\hat{\boldsymbol{\rho}}} = -\left(\hat{\mathbf{M}}_i^v\right)_{\hat{\boldsymbol{\rho}}}\mathbf{b}_i^v\left[\mathbf{b}_i^{v\mathrm{T}}\hat{\mathbf{M}}_i^v\mathbf{b}_i^v\right]^{-1} - \hat{\mathbf{M}}_i^v\left(\mathbf{b}_i^v\right)_{\hat{\boldsymbol{\rho}}}\left[\mathbf{b}_i^{v\mathrm{T}}\hat{\mathbf{M}}_i^v\mathbf{b}_i^v\right]^{-1} \\ -\hat{\mathbf{M}}_i^v\mathbf{b}_i^v\left(-\left[\mathbf{b}_i^{v\mathrm{T}}\hat{\mathbf{M}}_i^v\mathbf{b}_i^v\right]^{-1}\left\{\left(\mathbf{b}_i^{v\mathrm{T}}\right)_{\hat{\boldsymbol{\rho}}}\hat{\mathbf{M}}_i^v\mathbf{b}_i^v + \mathbf{b}_i^{v\mathrm{T}}\left(\hat{\mathbf{M}}_i^v\right)_{\hat{\boldsymbol{\rho}}}\mathbf{b}_i^v\right.\right. \\ \left.\left. +\mathbf{b}_i^{v\mathrm{T}}\hat{\mathbf{M}}_i^v\left(\mathbf{b}_i^v\right)_{\hat{\boldsymbol{\rho}}}\right\}\left[\mathbf{b}_i^{v\mathrm{T}}\hat{\mathbf{M}}_i^v\mathbf{b}_i^v\right]^{-1}\right)\end{aligned} \tag{4.67b}$$

Analogously to (4.64), $\left(\mathbf{K}_i^{\mathbf{z}}\right)_{\hat{\boldsymbol{\rho}}}$ can be reformulated as:

$$\left(\mathbf{K}_i^{\mathbf{z}}\right)_{\hat{\mathbf{z}}} = -\left\{\mathbf{K}_i^{\mathbf{z}}\left(\mathbf{b}_i^{v\mathrm{T}}\right)_{\hat{\boldsymbol{\rho}}}\hat{\mathbf{M}}_i^v\mathbf{b}_i^v + \mathbf{K}_i\left(\hat{\mathbf{M}}_i^v\right)_{\hat{\boldsymbol{\rho}}}\mathbf{b}_i^v + \mathbf{K}_i\hat{\mathbf{M}}_i^v\left(\mathbf{b}_i^v\right)_{\hat{\boldsymbol{\rho}}}\right\}\left[\mathbf{b}_i^{v\mathrm{T}}\hat{\mathbf{M}}_i^v\mathbf{b}_i^v\right]^{-1} \tag{4.68}$$

Observe that several matrix and tensor products appear, but most of them can be easily simplified looking to the expressions of the individual derivative of each elemental term. Similarly, the same terms are recalled in different expressions, so the derivatives can be calculated once and then stored. Other simplification consists in the calculation of the derivative of any symmetric matrix by only evaluating and assembling the terms of the main diagonal and one of the superior or inferior triangular part of the resulting tensor.

2. **Solution of the sensitivities, from the root to the leaves.**

For the sake of clarity, let us transform (2.247e):

$$\ddot{\mathbf{z}}_i = \left[\mathbf{b}_i^{v\mathrm{T}}\hat{\mathbf{M}}_i^v\mathbf{b}_i^v\right]^{-1}\mathbf{H}_i \tag{4.69a}$$

$$\mathbf{H}_i = \mathbf{b}_i^{v\mathrm{T}}\left[\hat{\mathbf{Q}}_i^v - \hat{\mathbf{M}}_i^v\left(\mathbf{B}_i^v\dot{\mathbf{V}}_{i-1} + \mathbf{d}_i^v\right)\right] + \mathbf{Q}_i^{\mathbf{z}} \tag{4.69b}$$

Taking derivatives with respect to $\boldsymbol{\rho}$, and using equation (4.11):

$$\ddot{\mathbf{z}}_i' = \left(\ddot{\mathbf{z}}_i\right)_{\hat{\mathbf{z}}}\mathbf{z}' + \left(\ddot{\mathbf{z}}_i\right)_{\hat{\dot{\mathbf{z}}}}\dot{\mathbf{z}}' + \left(\ddot{\mathbf{z}}_i\right)_{\hat{\ddot{\mathbf{z}}}}\ddot{\mathbf{z}}' + \left(\ddot{\mathbf{z}}_i\right)_{\hat{\boldsymbol{\rho}}} \tag{4.70}$$

In (4.70), the terms $\left(\ddot{\mathbf{z}}_i\right)_{\hat{\mathbf{z}}}$, $\left(\ddot{\mathbf{z}}_i\right)_{\hat{\dot{\mathbf{z}}}}$ and $\left(\ddot{\mathbf{z}}_i\right)_{\hat{\ddot{\mathbf{z}}}}$ measure the variation of the acceleration of the joint $i$ with the changes in positions, velocities and accelerations in the rest of the coordinates of the model. These derivatives can be calculated from (4.69) as:

$$\left(\ddot{\mathbf{z}}_i\right)_{\hat{\mathbf{z}}} = \left(\left[\mathbf{b}_i^{v\mathrm{T}}\hat{\mathbf{M}}_i^v\mathbf{b}_i^v\right]^{-1}\right)_{\hat{\mathbf{z}}}\mathbf{H}_i + \left[\mathbf{b}_i^{v\mathrm{T}}\hat{\mathbf{M}}_i^v\mathbf{b}_i^v\right]^{-1}\left(\mathbf{H}_i\right)_{\hat{\mathbf{z}}} \tag{4.71a}$$

$$\left(\ddot{\mathbf{z}}_i\right)_{\hat{\dot{\mathbf{z}}}} = \left[\mathbf{b}_i^{v\mathrm{T}}\hat{\mathbf{M}}_i^v\mathbf{b}_i^v\right]^{-1}\left(\mathbf{H}_i\right)_{\hat{\dot{\mathbf{z}}}} \tag{4.71b}$$

$$\left(\ddot{\mathbf{z}}_i\right)_{\hat{\ddot{\mathbf{z}}}} = \left[\mathbf{b}_i^{v\mathrm{T}}\hat{\mathbf{M}}_i^v\mathbf{b}_i^v\right]^{-1}\left(\mathbf{H}_i\right)_{\hat{\ddot{\mathbf{z}}}} \tag{4.71c}$$

$$\left(\ddot{\mathbf{z}}_i\right)_{\hat{\boldsymbol{\rho}}} = \left(\left[\mathbf{b}_i^{v\mathrm{T}}\hat{\mathbf{M}}_i^v\mathbf{b}_i^v\right]^{-1}\right)_{\hat{\boldsymbol{\rho}}}\mathbf{H}_i + \left[\mathbf{b}_i^{v\mathrm{T}}\hat{\mathbf{M}}_i^v\mathbf{b}_i^v\right]^{-1}\left(\mathbf{H}_i\right)_{\hat{\boldsymbol{\rho}}} \tag{4.71d}$$

Observe that although this sensitivity is obtained from a fully-recursive formulation, the dependencies now are more mixed and linked, so in the calculation of each of the derivatives above, the derivatives with respect to all the relative coordinates must be considered, and not only the ones associated to each joint.

The new derivatives of $\mathbf{H}_i$ introduced in (4.71) have the following expressions:

$$\begin{aligned}\left(\mathbf{H}_i\right)_{\hat{\mathbf{z}}} = {} & \left(\mathbf{b}_i^{v\mathrm{T}}\right)_{\hat{\mathbf{z}}}\left[\hat{\mathbf{Q}}_i^v - \hat{\mathbf{M}}_i^v\left(\mathbf{B}_i^v\dot{\mathbf{V}}_{i-1} + \mathbf{d}_i^v\right)\right] + \left(\mathbf{Q}_i^{\mathbf{z}}\right)_{\hat{\mathbf{z}}} + \\ & \mathbf{b}_i^{v\mathrm{T}}\left[\left(\hat{\mathbf{Q}}_i^v\right)_{\hat{\mathbf{z}}} - \left(\hat{\mathbf{M}}_i^v\right)_{\hat{\mathbf{z}}}\left(\mathbf{B}_i^v\dot{\mathbf{V}}_{i-1} + \mathbf{d}_i^v\right)\right. \\ & \left. -\hat{\mathbf{M}}_i^v\left(\left(\mathbf{B}_i^v\right)_{\hat{\mathbf{z}}}\dot{\mathbf{V}}_{i-1} + \mathbf{B}_i^v\left(\dot{\mathbf{V}}_{i-1}\right)_{\hat{\mathbf{z}}} + \left(\mathbf{d}_i^v\right)_{\hat{\mathbf{z}}}\right)\right]\end{aligned} \tag{4.72a}$$

$$\left(\mathbf{H}_i\right)_{\hat{\dot{\mathbf{z}}}} = \mathbf{b}_i^{v\mathrm{T}}\left[\left(\hat{\mathbf{Q}}_i^v\right)_{\hat{\dot{\mathbf{z}}}} - \hat{\mathbf{M}}_i^v\left(\mathbf{d}_i^v\right)_{\hat{\dot{\mathbf{z}}}}\right] + \left(\mathbf{Q}_i^{\mathbf{z}}\right)_{\hat{\dot{\mathbf{z}}}} \tag{4.72b}$$

$$\left(\mathbf{H}_i\right)_{\hat{\ddot{\mathbf{z}}}} = -\mathbf{b}_i^{v\mathrm{T}}\hat{\mathbf{M}}_i^v\mathbf{B}_i^v\left(\dot{\mathbf{V}}_{i-1}\right)_{\hat{\ddot{\mathbf{z}}}} \tag{4.72c}$$

$$\begin{aligned}\left(\mathbf{H}_i\right)_{\hat{\boldsymbol{\rho}}} = \left(\mathbf{b}_i^{v\mathrm{T}}\right)_{\hat{\boldsymbol{\rho}}}\left[\hat{\mathbf{Q}}_i^v - \hat{\mathbf{M}}_i^v\left(\mathbf{B}_i^v\dot{\mathbf{V}}_{i-1} + \mathbf{d}_i^v\right)\right] + \left(\mathbf{Q}_i^{\mathbf{z}}\right)_{\hat{\boldsymbol{\rho}}} + \\ \mathbf{b}_i^{v\mathrm{T}}\left[\left(\hat{\mathbf{Q}}_i^v\right)_{\hat{\boldsymbol{\rho}}} - \left(\hat{\mathbf{M}}_i^v\right)_{\hat{\boldsymbol{\rho}}}\left(\mathbf{B}_i^v\dot{\mathbf{V}}_{i-1} + \mathbf{d}_i^v\right)\right. \\ \left. -\hat{\mathbf{M}}_i^v\left(\left(\mathbf{B}_i^v\right)_{\hat{\boldsymbol{\rho}}}\dot{\mathbf{V}}_{i-1} + \mathbf{B}_i^v\left(\dot{\mathbf{V}}_{i-1}\right)_{\hat{\boldsymbol{\rho}}} + \left(\mathbf{d}_i^v\right)_{\hat{\boldsymbol{\rho}}}\right)\right]\end{aligned} \tag{4.72d}$$

Observe that (4.70) can be easily solved applying a fixed point scheme in accelerations, and the sensitivities of positions and velocities can be obtained by means of a numerical integrator. Looking at the sensitivities of the accelerations, the term $\left(\mathbf{H}_i\right)_{\hat{\ddot{\mathbf{z}}}}$ is zero for the derivatives with respect to the coordinates from the joint $i-1$ to the joints of the leaves of the chain.

The forward sensitivity analysis of fully-recursive formulations applied to open-loop systems can be solved with the following algorithm, implemented in the MBSLIM multibody library:

1. Computation of the dynamics with a fully-recursive scheme.
2. Computation of corrector terms of the integrator by means of (4.16).
3. Accumulation of derivatives of forces and masses, from the leaves to the root, using (4.62), (4.63), (4.65) and (4.66).
4. Solution of $\ddot{\mathbf{z}}_i'$, of body $i$, using (4.70), starting from the base body.
5. Correction of the sensitivities of positions and velocities for joint $i$ by means of a Newmark's family integrator:

$$\mathbf{z}_{n+1}' = \mathbf{z}_n' + h\dot{\mathbf{z}}_n' + \frac{h^2}{2}\left\{(1-2\beta)\,\ddot{\mathbf{z}}_n' + 2\beta\ddot{\mathbf{z}}_{n+1}'\right\} \tag{4.73a}$$

$$\dot{\mathbf{z}}_{n+1}' = \dot{\mathbf{z}}_n' + h\left\{(1-\gamma)\,\ddot{\mathbf{z}}_n' + \gamma\ddot{\mathbf{z}}_{n+1}'\right\} \tag{4.73b}$$

6. Update of derivatives of $\dot{\mathbf{V}}_i$.
7. Repeat step 4 to 6 until the tips of the mechanism are reached.
8. Return to step 1 if $t < t_{fin}$.

The biggest drawback of the fully-recursive expressions in the dynamics relies on the concatenation of products of $6\times 6$ matrices, as $\mathbf{K}_i$ or $\hat{\mathbf{M}}_i^v$, and this problem is increased adding a new dimension in those matrices during the sensitivity analysis, generating concatenations of tensor products. Some of these products can be simplified decomposing them in smaller structures, like $\mathbf{B}_i^v$, but others cannot be simplified.

In this sense, the semi-recursive approach allows more simplifications during the assembly, but large single-chain systems are still better suited to be solved with the fully-recursive approach, for both dynamic and sensitivity problems.

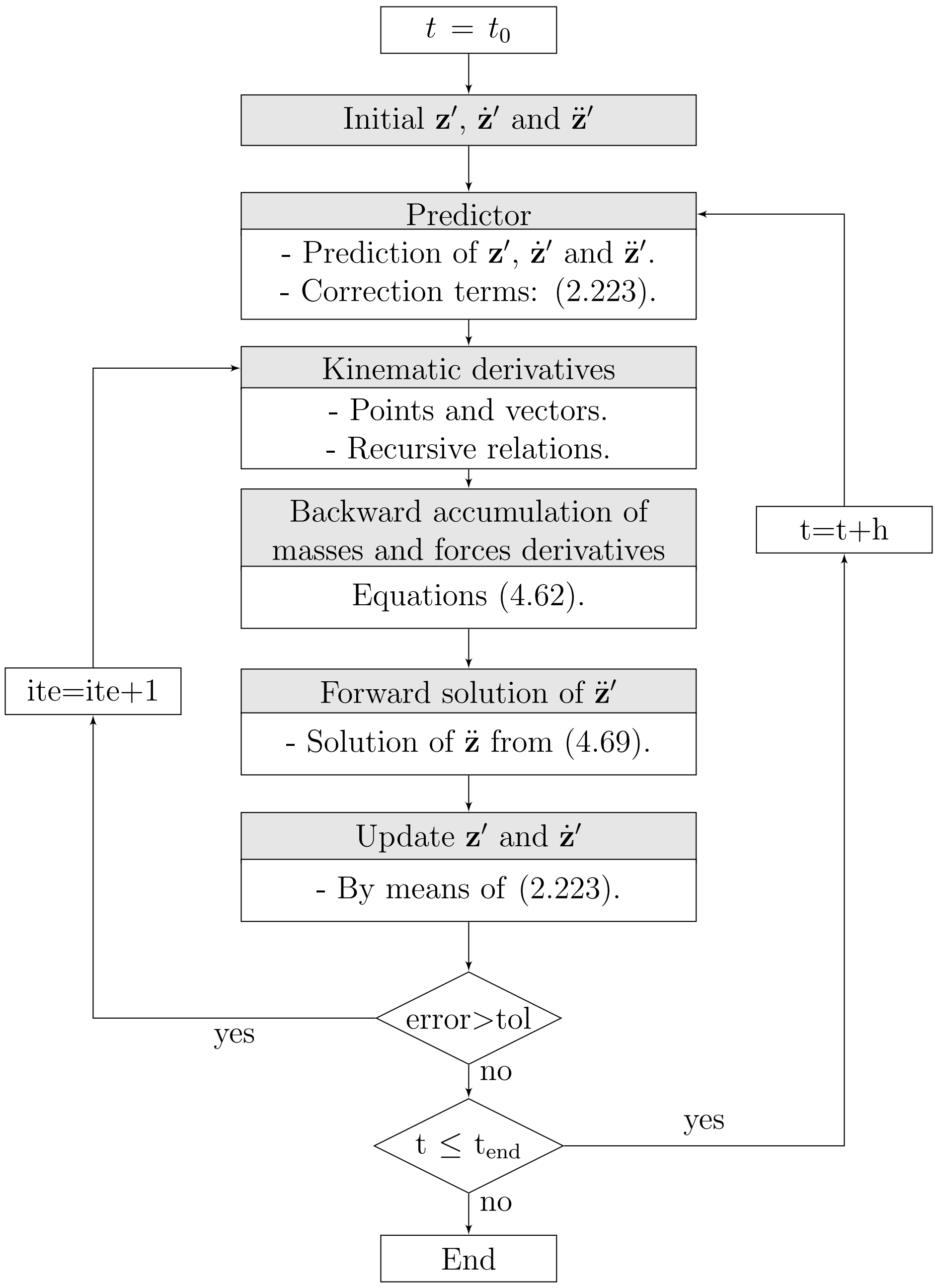

$t = t_0$
Initial $\mathbf{z}'$, $\dot{\mathbf{z}}'$ and $\ddot{\mathbf{z}}'$
Predictor
- Prediction of $\mathbf{z}'$, $\dot{\mathbf{z}}'$ and $\ddot{\mathbf{z}}'$.
- Correction terms: (2.223).
Kinematic derivatives
- Points and vectors.
- Recursive relations.
Backward accumulation of masses and forces derivatives
Equations (4.62).
Forward solution of $\ddot{\mathbf{z}}'$
- Solution of $\ddot{\mathbf{z}}$ from (4.69).
Update $\mathbf{z}'$ and $\dot{\mathbf{z}}'$
- By means of (2.223).
error>tol
yes
no
ite=ite+1
t=t+h
t ≤ t$_{\text{end}}$
yes
no
End

Figure 4.1: Flowchart for the sensitivity analysis of the fully-recursive dynamics of open-loop systems.

## 4.4 Derivatives of recursive kinematic relations

One of the most significant differences between global and topological formulations is the presence of kinematic joint relations needed for the assemblies and accumulations in topological models. These magnitudes are expressed in terms of positions and velocities of points and vectors as well as Euler parameters, which means that joint related terms vary with respect to the relative coordinates and other parameters, thus their derivatives have to be computed when a sensitivity analysis is addressed. In the case of partial derivatives with respect to parameters, they are null unless a local coordinate affecting the topology of the model is selected as sensitivity parameter.

Among all the recursive kinematic relations involved in the formulations described in this work, two categories could be identified: the terms dependent on the type of joint, and the ones whose calculation is invariant for any type of joint. $\mathbf{A}_i$, $\mathbf{b}_i^v$ and $\dot{\mathbf{b}}_i^v$ would be the terms found in the first class, whereas the second category would be composed of $\mathbf{B}_i^v$, $\dot{\mathbf{B}}_i^v$ and $\mathbf{d}_i^v$.

### 4.4.1 Evaluation of $(\mathbf{A}_i)_\mathbf{z}$

The rotation matrix is directly involved in the evaluation of different kinematic and dynamic terms, such as in the evaluation of the global position of a point (2.1) or a vector (2.2), in the kinematic relations of spherical joints (2.171a) (through term $\mathbf{E}$) or in the evaluation of the inertia tensor in global coordinates (2.196). Since the relative orientation of a body varies with joint coordinates, the derivative of a rotation matrix with respect to the relative states is explicitly required in any dynamic sensitivity analysis.

Regarding (2.19), it can be said that the rotation matrix of a body can be calculated as the product of the rotation matrix of the previous body in the kinematic chain by a relative rotation matrix dependent on the joint governing the motion between these two bodies. Starting from this expression, the derivative of the rotation matrix of a body can be calculated as:

$$(\mathbf{A}_i)_\mathbf{z} = \frac{\partial \mathbf{A}_{i-1}}{\partial \mathbf{z}} \mathbf{A}_i^{i-1} + \mathbf{A}_{i-1} \frac{\partial \mathbf{A}_i^{i-1}}{\partial \mathbf{z}} \tag{4.74}$$

In the previous expression, the concatenation and assembly of relative rotation matrices is needed to obtain the rotation matrices of each body, and the same procedure may be done to calculate its derivatives.

Equation (4.74) involves a concatenation and assembly of the derivatives of subsequent relative rotation matrices. The differentiation can be significantly simplified by reformulating the procedure. For the sake of clearness, let us consider an open-loop system composed of 3 bodies and the ground, with 3 joints of any kind connecting the bodies. The rotation matrix of body 3 can be expressed as:

$$\mathbf{A}_3 = \mathbf{A}_2 \mathbf{A}_3^2 = \mathbf{A}_1 \mathbf{A}_2^1 \mathbf{A}_3^2 = \mathbf{A}_1^0 \mathbf{A}_2^1 \mathbf{A}_3^2 \tag{4.75}$$

Considering only relative rotation matrices, the derivative of $\mathbf{A}_3$ with respect to $\mathbf{z}$ becomes:

$$\frac{\partial \mathbf{A}_i}{\partial \mathbf{z}} = \frac{\partial \mathbf{A}_1^0}{\partial \mathbf{z}} \mathbf{A}_2^1 \mathbf{A}_3^2 + \mathbf{A}_1^0 \frac{\partial \mathbf{A}_2^1}{\partial \mathbf{z}} \mathbf{A}_3^2 + \mathbf{A}_1^0 \mathbf{A}_2^1 \frac{\partial \mathbf{A}_3^2}{\partial \mathbf{z}} \tag{4.76}$$

Observe that (4.77) only involves derivatives of relative rotation matrices, which are exclusively dependent on the set of coordinates describing the joint. Considering the recursive composition of rotation matrices, (4.77) can be transformed into:

$$\frac{\partial \mathbf{A}_i}{\partial \mathbf{z}} = \frac{\partial \mathbf{A}_1^0}{\partial \mathbf{z}} \mathbf{A}_1^{\mathrm{T}} \mathbf{A}_3 + \mathbf{A}_1 \frac{\partial \mathbf{A}_2^1}{\partial \mathbf{z}} \mathbf{A}_2^{\mathrm{T}} \mathbf{A}_3 + \mathbf{A}_2 \frac{\partial \mathbf{A}_3^2}{\partial \mathbf{z}} \mathbf{A}_3^{\mathrm{T}} \mathbf{A}_3 \tag{4.77}$$

regarding that

$$\mathbf{A}_1^0 = \mathbf{A}_1 \tag{4.78a}$$
$$\mathbf{A}_1^0 \mathbf{A}_2^1 = \mathbf{A}_2 \tag{4.78b}$$
$$\mathbf{A}_2^1 \mathbf{A}_3^2 = \left(\mathbf{A}_1^0\right)^{\mathrm{T}} \mathbf{A}_1^0 \mathbf{A}_2^1 \mathbf{A}_3^2 = \mathbf{A}_1^{\mathrm{T}} \mathbf{A}_3 \tag{4.78c}$$
$$\mathbf{A}_3^2 = \mathbf{A}_2^{\mathrm{T}} \mathbf{A}_2 \mathbf{A}_3^2 = \mathbf{A}_2^{\mathrm{T}} \mathbf{A}_3 \tag{4.78d}$$
$$\mathbf{A}_3^{\mathrm{T}} \mathbf{A}_3 = \mathbf{I} \tag{4.78e}$$

Thus, from the previous example, the following expression for the derivative of any rotation matrix with respect to relative coordinates in compact form is achieved:

$$\left(\mathbf{A}_i\right)_{\mathbf{z}} = \left( \sum_{j=1}^{i} \mathbf{A}_{j-1} \frac{\partial \mathbf{A}_j^{j-1}}{\partial \mathbf{z}} \mathbf{A}_j{}^{\mathrm{T}} \right) \mathbf{A}_i \tag{4.79}$$

Observe in (4.79) that the derivatives are highly simplified since they are expressed directly in terms of relative rotation matrices. Moreover, the relative rotation matrix derivatives related to a particular joint with respect to any relative coordinate different from those of the joint are all null. In this sense, the differentiation process is more direct than (4.74).

Applying the matrix transpose properties, expression (4.79) can be reformulated to calculate the partial derivative of the transpose of the rotation matrix:

$$\left(\mathbf{A}_i^{\mathrm{T}}\right)_{\mathbf{z}} = \mathbf{A}_i^{\mathrm{T}} \left( \sum_{j=1}^{i} \mathbf{A}_j \frac{\partial \left(\mathbf{A}_j^{j-1}\right)^{\mathrm{T}}}{\partial \mathbf{z}} \mathbf{A}_{j-1}^{\mathrm{T}} \right) \tag{4.80}$$

Each type of joint involves a different expression for its relative rotation matrix, thus they have to be studied separately for each joint. In the following lines, the derivatives with respect to relative coordinates in positions are presented. In the cases in which differentiation does not involve a particular difficulty or a reformulation, final results will be directly introduced.

- **Revolute joint**

  The relative rotation matrix of a revolute joint (2.20) is described by the product of a constant matrix $\mathbf{A}^1$ by the rotation matrix of a single rotation of value

$z_i$ around the vector defining the joint. Differentiating expression (2.20) and reusing (2.22), the following derivative is obtained:

$$\frac{\partial \mathbf{A}_i^{i-1}}{\partial z_i} = \mathbf{A}^1 \left( -\sin(z_i)\ \mathbf{I} + \sin(z_i) \bar{\mathbf{w}}_j^i \left(\bar{\mathbf{w}}_j^i\right)^{\mathrm{T}} + \tilde{\bar{\mathbf{w}}}_j^i \cos(z_i) \right) \tag{4.81}$$

with $\mathbf{A}^1$ defined in (2.21).

- **Prismatic joint**

  Prismatic joints only allow relative translational motion and prevent relative rotation. Concerning to (2.30) and (2.31), the relative rotation matrix is constant, thus its derivative is null.

$$\frac{\partial \mathbf{A}_i^{i-1}}{\partial z_i} = \mathbf{0}_{3\times 3} \tag{4.82}$$

- **Spherical joint**

  The particular spherical joint modeling introduced in chapter 2 by means of Euler parameters presents a series of issues into the differentiation process that have to be carefully handled in order to keep consistency.

  The kinematic relations for a spherical joint were reached under the assumption of normalized Euler parameters, which means that expressions of angular velocities and accelerations are subjected to this condition too. In order to keep consistency, the rotation matrix parameterized by means of Euler parameters $\bar{\mathbf{p}} = \begin{bmatrix} e_0 & e_1 & e_2 & e_3 \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} e_0 & \bar{\mathbf{e}} \end{bmatrix}^{\mathrm{T}}$, with $\bar{\mathbf{p}} \in \mathbb{R}^4$ and $\bar{\mathbf{e}} \in \mathbb{R}^3$, must be computed subjected to the normalization constraint:

$$\mathbf{A}_i^{i-1} = \bar{\mathbf{E}}\bar{\mathbf{G}}^{\mathrm{T}} + \left(1 - \bar{\mathbf{p}}^{\mathrm{T}}\bar{\mathbf{p}}\right) \bar{\mathbf{E}}\bar{\mathbf{G}}^{\mathrm{T}} = \bar{\mathbf{E}}\bar{\mathbf{G}}^{\mathrm{T}} \left(2 - \bar{\mathbf{p}}^{\mathrm{T}}\bar{\mathbf{p}}\right) \tag{4.83}$$

  in which $\bar{\mathbf{E}}$ and $\bar{\mathbf{G}}$ have been defined in section 2.1.5. Behold that, as long as the normalization constraint is fulfilled at position level, equations (4.83) and (2.63) are identical. Considering (4.83), the derivative of the rotation matrix takes the form:

$$\frac{\partial \mathbf{A}_i^{i-1}}{\partial \bar{\mathbf{p}}} = \left( \frac{\partial \bar{\mathbf{E}}}{\partial \bar{\mathbf{p}}} \bar{\mathbf{G}}^{\mathrm{T}} + \bar{\mathbf{E}} \frac{\partial \bar{\mathbf{G}}^{\mathrm{T}}}{\partial \bar{\mathbf{p}}} \right) \left(2 - \bar{\mathbf{p}}^{\mathrm{T}}\bar{\mathbf{p}}\right) + 2\bar{\mathbf{E}}\bar{\mathbf{G}}^{\mathrm{T}} \left(\bar{\mathbf{p}}^{\mathrm{T}}\right) \tag{4.84}$$

  In (4.84), partial derivatives of $\bar{\mathbf{E}}$ and $\bar{\mathbf{G}}$ are determined by:

$$\frac{\partial \bar{\mathbf{E}}}{\partial e_0} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{4.85a}$$

$$\frac{\partial \bar{\mathbf{E}}}{\partial e_1} = \begin{bmatrix} \mathbf{h} & \tilde{\mathbf{h}} \end{bmatrix};\ \textit{with}\ \mathbf{h} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \tag{4.85b}$$

$$\frac{\partial \bar{\mathbf{E}}}{\partial e_2} = \begin{bmatrix} \mathbf{h} & \tilde{\mathbf{h}} \end{bmatrix};\ \textit{with}\ \mathbf{h} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \tag{4.85c}$$

$$\frac{\partial \bar{\mathbf{E}}}{\partial e_3} = \begin{bmatrix} \mathbf{h} & \tilde{\mathbf{h}} \end{bmatrix};\ \textit{with}\ \mathbf{h} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \tag{4.85d}$$

$$\frac{\partial \bar{\mathbf{G}}}{\partial e_0} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{4.86a}$$

$$\frac{\partial \bar{\mathbf{G}}}{\partial e_1} = \begin{bmatrix} \mathbf{h} & -\tilde{\mathbf{h}} \end{bmatrix} \text{ ; } \textit{with}\, \mathbf{h} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \tag{4.86b}$$

$$\frac{\partial \bar{\mathbf{G}}}{\partial e_2} = \begin{bmatrix} \mathbf{h} & -\tilde{\mathbf{h}} \end{bmatrix} \text{ ; } \textit{with}\, \mathbf{h} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \tag{4.86c}$$

$$\frac{\partial \bar{\mathbf{G}}}{\partial e_3} = \begin{bmatrix} \mathbf{h} & -\tilde{\mathbf{h}} \end{bmatrix} \text{ ; } \textit{with}\, \mathbf{h} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \tag{4.86d}$$

It should be remarked that the rotation matrix can be expressed by different arithmetic operations of Euler parameters regarding the fulfillment of the normalization condition, as demonstrated by equations (2.61) and (2.63), and each one of these expressions would lead to different derivatives. In fact, there are also multiple configurations of the Euler parameters that lead to exactly the same rotation matrix since the four parameters are not independent. Consequently, depending on how the Euler parameters are handled in velocity and acceleration expressions, a different formula for the rotation matrix could be convenient.

The particular derivative presented stemmed from several numerical experiments in which several derivative identities between positions and velocities of different dynamic and kinematic terms were verified.

- **Cylindrical joint**

  In the cylindrical joint, only the coordinate related to the rotation around the vector defining the joint contributes to the change the rotation matrix:

$$\frac{\partial \mathbf{A}_i^{i-1}}{\partial z_{i1}} = \mathbf{0}_{3\times 3} \tag{4.87}$$

$$\frac{\partial \mathbf{A}_i^{i-1}}{\partial z_{i2}} = \mathbf{A}^1 \left( -\sin z_{i2}\, \mathbf{I} + \sin z_{i2} \bar{\mathbf{w}}_j^i \left( \bar{\mathbf{w}}_j^i \right)^{\mathrm{T}} + \tilde{\bar{\mathbf{w}}}_j^i \cos z_{i2} \right) \tag{4.88}$$

- **Cardan joint**

  The relative rotation matrix of a Cardan joint is the result of the product of three rotation matrices, as shown in (2.40). Taking derivatives on (2.40) and (2.41), this term takes the form:

$$\frac{\partial \mathbf{A}_i^{i-1}}{\partial \mathbf{z}} = \frac{\partial \mathbf{A}_{k1}^1}{\partial \mathbf{z}} \mathbf{A}^2 \mathbf{A}_{k2}^3 + \mathbf{A}_{k1}^1 \mathbf{A}^2 \frac{\partial \mathbf{A}_{k2}^3}{\partial \mathbf{z}} \tag{4.89}$$

  wherein

$$\frac{\partial \mathbf{A}_{k1}^1}{\partial z_{i1}} = -\sin z_{i1}\, \mathbf{I} + \sin z_{i1} \bar{\mathbf{w}}_j^{i-1} \left( \bar{\mathbf{w}}_j^{i-1} \right)^{\mathrm{T}} + \tilde{\bar{\mathbf{w}}}_j^{i-1} \cos z_{i1} \tag{4.90}$$

$$\frac{\partial \mathbf{A}_{k2}^3}{\partial z_{i2}} = -\sin z_{i2}\, \mathbf{I} + \sin z_{i2} \bar{\mathbf{w}}_{j+1}^i \left( \bar{\mathbf{w}}_{j+1}^i \right)^{\mathrm{T}} + \tilde{\bar{\mathbf{w}}}_{j+1}^i \cos z_{i2} \tag{4.91}$$

- **Floating joint**

  The three first coordinates of the floating joint coincide with the position of the CoM of the first body of the kinematic chain. The derivative of the rotation matrix with respect to these is null due to the fact that the motion described by these coordinates is only translational.

$$\frac{\partial \mathbf{A}_i^{i-1}}{\partial \mathbf{r}_G^i} = \mathbf{0}_{3\times3\times3} \tag{4.92}$$

  Besides, the derivatives with respect to the Euler parameters of the floating joint are identical to the ones presented for the spherical joint (4.84).

- **Planar joint**

  The planar joint is internally composed of 2 prismatic and one revolute joint. Regarding that prismatic joints prevent orientation changes, the derivatives of the rotation matrix with respect to the two first coordinates of the joint are null. On the other hand, the derivative with respect to the third angular coordinate is:

$$\frac{\partial \mathbf{A}_i^{i-1}}{\partial z_{k3}} = \mathbf{A}^1 \left( -\sin z_{k3}\ \mathbf{I} + \sin z_{k3} \bar{\mathbf{w}}_j^i \left( \bar{\mathbf{w}}_j^i \right)^{\mathrm{T}} + \tilde{\bar{\mathbf{w}}}_j^i \cos z_{k3} \right) \tag{4.93}$$

### 4.4.2 Evaluation of $(\mathbf{A}_i)_{\boldsymbol{\rho}}$

The number of parameters that can explicitly affect a rotation matrix can be reduced to the local coordinates of a point or a vector that defines the orientation of a body or a kinematic joint. More precisely, if only vector coordinates are considered in the definition of any relative rotation axis (as implemented in MBSLIM), then only local coordinates of vectors will affect rotation matrices. In that case, the casuistic of partial derivatives of a rotation matrix with respect to parameters is reduced to the case in which a local coordinate of a vector defining a joint is defined as parameter.

Although the derivatives of each relative rotation matrix with respect to each parameter must be calculated individually, the following assembly (equivalent to (4.79)) is needed:

$$(\mathbf{A}_i)_{\boldsymbol{\rho}} = \left( \sum_{j=1}^{i} \mathbf{A}_{j-1} \frac{\partial \mathbf{A}_j^{j-1}}{\partial \boldsymbol{\rho}} \mathbf{A}_j{}^T \right) \mathbf{A}_i \tag{4.94}$$

During the assessment of each $\dfrac{\partial \mathbf{A}_j^{j-1}}{\partial \boldsymbol{\rho}}$, the constant rotation matrices involved in the calculation of the relative ones, described in chapter 2 and which remain constant in the calculation of $\dfrac{\partial \mathbf{A}_j^{j-1}}{\partial \mathbf{z}}$, must be considered and differentiated too.

The particular expressions for the partial derivative for each type of joint are omitted here for the sake of brevity. They can be directly obtained by taking derivatives on each relative rotation matrix expression in combination with equation (4.94).

### 4.4.3 Evaluation of $(\mathbf{b}_i^v)_{\hat{\mathbf{z}}}$

Each type of joint has different recursive kinematic expressions, but regarding the entities used to express $\mathbf{b}_i^v$, some patterns can be detected and harnessed in order to gather terms and reuse structures of computation. Basically, the derivatives of $\mathbf{b}_i^v$ with respect to the relative coordinates can be divided in two types: the ones exclusively dependent on points and vectors, and the ones involving also Euler parameters.

As a means to simplify the notation and make the final expressions more clear, a decomposition of the term $\mathbf{b}_i^v$ is considered. Firstly, $\mathbf{b}_i^v$ is assumed to be composed of $2n_j^i$ arrays $\mathbf{b}_i^{vj,k} \in \mathbb{R}^3$ (being $n_j^i$ the number of coordinates of joint $i$). With this consideration, the term $\mathbf{b}_i^v$ of a spherical joint, for instance, can be expressed as:

$$\mathbf{b}_i^v = \begin{bmatrix} \mathbf{b}_i^{v1,1} & \mathbf{b}_i^{v1,2} & \mathbf{b}_i^{v1,3} & \mathbf{b}_i^{v1,4} \\ \mathbf{b}_i^{v2,1} & \mathbf{b}_i^{v2,2} & \mathbf{b}_i^{v2,3} & \mathbf{b}_i^{v2,4} \end{bmatrix} \tag{4.95}$$

A second decomposition into arrays $\mathbf{b}_i^{vj} \in \mathbb{R}^6$ will be also considered, with each array $\mathbf{b}_i^{vj}$ being the $j$-th column of matrix $\mathbf{b}_i^v$. Observe that both definitions are related by means of:

$$\mathbf{b}_i^{vj} = \begin{bmatrix} \mathbf{b}_i^{v1,j} \\ \mathbf{b}_i^{v2,j} \end{bmatrix} \tag{4.96}$$

Accordingly, the term $\mathbf{b}_i^v$ of the spherical joint can be expressed with this notation as:

$$\mathbf{b}_i^v = \begin{bmatrix} \mathbf{b}_i^{v1} & \mathbf{b}_i^{v2} & \mathbf{b}_i^{v3} & \mathbf{b}_i^{v4} \end{bmatrix} \tag{4.97}$$

These two divisions of $\mathbf{b}_i^v$ make possible a differentiation of arrays instead of matrices and they allow to take derivatives separately on the linear part of the term (three first rows) and on the angular part (fourth to sixth rows). The present notation intends to make the differentiation more readable and to reflect the close relation of the term $\mathbf{b}_i^v$ with its own derivatives.

Furthermore, it is convenient to extend the properties of the skew symmetric matrix of a vector to the skew symmetric tensor of a matrix. The description and properties of this operation are presented in appendix B.

For the brevity and the sake of clearness, only a few of the expressions are fully explained, and in other cases only the final expressions are presented. First, the expression for any reference point is described, and then the particularizations for RTdyn0 and RTdyn1 are introduced.

- **Revolute joint**

  Taking derivatives on the revolute joint expression of $\mathbf{b}_i^v$ for any reference point (2.133a), the following expression is reached:

$$(\mathbf{b}_i^v)_{\hat{\mathbf{z}}} = \begin{bmatrix} \tilde{\mathbf{w}}_j \left( \dfrac{\partial \mathbf{r}_i}{\partial \mathbf{z}} - \dfrac{\partial \mathbf{r}_j}{\partial \mathbf{z}} \right) + \left( \tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i \right) \dfrac{\partial \mathbf{w}_j}{\partial \mathbf{z}} \\ \dfrac{\partial \mathbf{w}_j}{\partial \mathbf{z}} \end{bmatrix} \tag{4.98}$$

The derivatives of global positions of points and vectors with respect to relative coordinates introduced in section 3.6 can be now applied:

$$(\mathbf{b}_i^v)_{\hat{\mathbf{z}}} = \begin{bmatrix} \tilde{\mathbf{w}}_j \left( \frac{\partial \mathbf{r}_i}{\partial \mathbf{z}} - \begin{bmatrix} \mathbf{I} & \tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_j \end{bmatrix} \mathbf{R}_i^v \right) + \left( \tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i \right) \begin{bmatrix} \mathbf{0} & -\tilde{\mathbf{w}}_j \end{bmatrix} \mathbf{R}_i^v \\ \begin{bmatrix} \mathbf{0} & -\tilde{\mathbf{w}}_j \end{bmatrix} \mathbf{R}_i^v \end{bmatrix} \tag{4.99}$$

Note that the position of the reference point is not differentiated due to the diverse types of points that could be selected as reference points, each one of them with a different variation with respect to the relative coordinates. This is evidenced in RTdyn0 and RTdyn1 semi-recursive accumulations. On the first formulation, the reference point is fixed in the local reference frame of each body and moves with it, hence the derivative expressions presented in 3.6 apply to this point. The RTdyn1 version, on the contrary, uses a point which is fixed in the global reference frame, so its global position is always constant and its derivative is zero. The sake of the general formulation for any set of reference points is to obtain a series of expressions that could be applied to any reference point selection and which are universally valid for any of them. In the sensitivities, the same philosophy is kept.

Equation (4.99) can be regrouped as:

$$(\mathbf{b}_i^v)_{\hat{\mathbf{z}}} = \begin{bmatrix} \left( \tilde{\mathbf{w}}_j \left( -\tilde{\mathbf{r}}_i + \tilde{\mathbf{r}}_j \right) + \left( \tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i \right) \left( -\tilde{\mathbf{w}}_j \right) \right) \begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix} \mathbf{R}_i^v \\ -\tilde{\mathbf{w}}_j \begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix} \mathbf{R}_i^v \end{bmatrix} + \begin{bmatrix} \tilde{\mathbf{w}}_j \left( \frac{\partial \mathbf{r}_i}{\partial \mathbf{z}} - \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^v \right) \\ \mathbf{0} \end{bmatrix} \tag{4.100}$$

Now, let us consider the following property of the skew symmetric matrix, with $\mathbf{u} \in \mathbb{R}^3$ and $\mathbf{v} \in \mathbb{R}^3$:

$$\tilde{\mathbf{u}}\tilde{\mathbf{v}} = \tilde{\mathbf{v}}\tilde{\mathbf{u}} + \left( \widetilde{\tilde{\mathbf{u}}\mathbf{v}} \right) \tag{4.101}$$

Applying (4.101) to (4.100):

$$\begin{aligned}(\mathbf{b}_i^v)_{\hat{\mathbf{z}}} = &\begin{bmatrix} \left( \tilde{\mathbf{w}}_j \left( -\tilde{\mathbf{r}}_i + \tilde{\mathbf{r}}_j \right) - \tilde{\mathbf{w}}_j \left( \tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i \right) + \operatorname{skew}\left( \left( \tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i \right) \left( -\mathbf{w}_j \right) \right) \right) \begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix} \mathbf{R}_i^v \\ -\tilde{\mathbf{w}}_j \begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix} \mathbf{R}_i^v \end{bmatrix} + \\ &\begin{bmatrix} \tilde{\mathbf{w}}_j \left( \frac{\partial \mathbf{r}_i}{\partial \mathbf{z}} - \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^v \right) \\ \mathbf{0} \end{bmatrix} = \\ &\begin{bmatrix} \operatorname{skew}\left( \left( \tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i \right) \left( -\mathbf{w}_j \right) \right) \begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix} \mathbf{R}_i^v \\ -\tilde{\mathbf{w}}_j \begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix} \mathbf{R}_i^v \end{bmatrix} + \begin{bmatrix} \tilde{\mathbf{w}}_j \left( \frac{\partial \mathbf{r}_i}{\partial \mathbf{z}} - \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^v \right) \\ \mathbf{0} \end{bmatrix}\end{aligned} \tag{4.102}$$

being skew() the skew symmetric matrix of the vector between parenthesis (see appendix B).

At this point, considering that $\tilde{\mathbf{u}}\mathbf{v} = -\tilde{\mathbf{v}}\mathbf{u}$ (see appendix B), equation (4.102) takes the form:

$$(\mathbf{b}_i^v)_{\hat{\mathbf{z}}} = \begin{bmatrix} \text{skew}\left(\tilde{\mathbf{w}}_j\left(\mathbf{r}_j - \mathbf{r}_i\right)\right) \\ -\tilde{\mathbf{w}}_j \end{bmatrix} \begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix} \mathbf{R}_i^v + \begin{bmatrix} \tilde{\mathbf{w}}_j \left( \dfrac{\partial \mathbf{r}_i}{\partial \mathbf{z}} - \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix} \mathbf{R}_i^v \right) \\ \mathbf{0} \end{bmatrix} \tag{4.103}$$

Comparing expressions (2.133a) and (4.103), the first addend of (4.103) can be computed as the skew symmetric matrix of the linear and angular parts of $\mathbf{b}^v$ multiplied by a term related to the topology of the mechanism. Therefore, the derivative of the recursive term $\mathbf{b}^v$ with respect to the relative coordinates can be finally expressed as:

$$(\mathbf{b}_i^v)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{v1,1}} \\ \widetilde{\mathbf{b}_i^{v2,1}} \end{bmatrix} \begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix} \mathbf{R}_i^v + \begin{bmatrix} \tilde{\mathbf{w}}_j \\ \mathbf{0} \end{bmatrix} \left( \frac{\partial \mathbf{r}_i}{\partial \mathbf{z}} - \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix} \mathbf{R}_i^v \right) \tag{4.104}$$

The collection of terms developed in this derivative has two purposes: first, the achievement of simple expressions dependent on the same term being differentiated; and second, the separation of the derivative of the reference point from the rest of the expression, so that it can be easily substituted. The scheme of differentiation and grouping of terms presented here will be used in the following types of joints.

**RTdyn0:** The derivative of the reference point, as it is rigidly attached to a body, is, in this case:

$$\frac{\partial \mathbf{r}_G^i}{\partial \mathbf{z}} = \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix} \mathbf{R}_i^y \tag{4.105}$$

Therefore, the derivative of $\mathbf{b}^y$ has the simplest expression:

$$(\mathbf{b}_i^y)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{y1,1}} \\ \widetilde{\mathbf{b}_i^{y2,1}} \end{bmatrix} \begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix} \mathbf{R}_i^y \tag{4.106}$$

**RTdyn1:** In this case, the derivative of the reference point global position is null, and the final expression is slightly more complex than the RTdyn0 version:

$$(\mathbf{b}_i^z)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{z1,1}} \\ \widetilde{\mathbf{b}_i^{z2,1}} \end{bmatrix} \begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix} \mathbf{R}_i^z + \begin{bmatrix} \tilde{\mathbf{w}}_j \\ \mathbf{0} \end{bmatrix} \begin{bmatrix}-\mathbf{I} & \mathbf{0}\end{bmatrix} \mathbf{R}_i^z \tag{4.107}$$

- **Prismatic joint**

This type of joint has the simplest expression for $\mathbf{b}_i^v$, with only a vector defining this term, without products or sums of terms, and with the same expression for any set of reference points. The derivative of expression (2.133a) yields:

$$(\mathbf{b}_i^v)_{\hat{\mathbf{z}}} = \begin{bmatrix} \tilde{\mathbf{u}}_j \\ \mathbf{0} \end{bmatrix} \begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix} \mathbf{R}_i^v = \begin{bmatrix} \widetilde{\mathbf{b}_i^{v1,1}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix} \mathbf{R}_i^v \tag{4.108}$$

Note that the expression of $\mathbf{b}^v$, and thus $\mathbf{b}_i^{v1,1}$ too, does not include a reference point, and consequently, equation (4.108) apply for both RTdyn0 and RTdyn1 versions, but with the corresponding topological matrix $\mathbf{R}^y$ or $\mathbf{R}^z$.

- **Cardan joint**

The derivative of the term $\mathbf{b}_i^v$ of a Cardan joint with respect to the relative coordinates can be readily calculated from the expressions of the revolute joint, but 3 considerations must be taken into account: first, the two vectors defining the joint belong to different bodies; second, the point defining the joint is shared between the two bodies of the joint; and third, the relative motions of the two elemental revolute joints are coupled, this is, the rotation around the first revolute joint determines the global position of the axis of the second rotation. Let us recall (2.149):

$$\mathbf{b}_i^v = \begin{bmatrix} \mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) & \mathbf{w}_{j+1} \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) \\ \mathbf{w}_j & \mathbf{w}_{j+1} \end{bmatrix} = \begin{bmatrix}\mathbf{b}_i^{v1} & \mathbf{b}_i^{v2}\end{bmatrix} \tag{4.109}$$

The point defining the joint, $\mathbf{r}_j$, is shared by the two bodies connected by the joint, and therefore, two different expressions for its derivative are possible, using expression (3.91):

$$\frac{\partial \mathbf{r}_j}{\partial \mathbf{z}} = \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_{i-1} - \tilde{\mathbf{r}}_j\end{bmatrix} \mathbf{R}_{i-1}^v \tag{4.110a}$$

$$\frac{\partial \mathbf{r}_j}{\partial \mathbf{z}} = \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_{i} - \tilde{\mathbf{r}}_j\end{bmatrix} \mathbf{R}_{i}^v \tag{4.110b}$$

Expanding terms, it could be proved that the two expressions are exactly the same, so the selection of one or other is made only regarding a notation criterion and with the aim of simplifying the analytical expressions. In this particular case, the point $\mathbf{r}_j$ is differentiated as belonging to body $i$ due to the fact that it appears in the expression of $\mathbf{b}_i$ in a subtraction with the reference point $\mathbf{r}_i$, contained in body $i$. With this consideration, the derivatives of both points are expressed in terms $\mathbf{R}_i^v$, simplifying the final expression.

Similarly to the cylindrical joint differentiation procedure, the term $\mathbf{b}_i$ will be divided in two parts, having each one an expression equivalent to the revolute joint. As it was commented before, the derivative of the term related to second revolute joint $\mathbf{b}_i^{v2}$ is identical to (4.104), but the first term will be different since the point $\mathbf{r}_i$ and vector $\mathbf{w}_j$ belong to different bodies.

First of all, the following relation between $\mathbf{R}_i^v$ and $\mathbf{R}_{i-1}^v$, derived form the assembly expressions of $\mathbf{R}^v$ (2.211) and (2.212), has to be introduced:

$$\mathbf{R}_i^v = \mathbf{B}_i^v \mathbf{R}_{i-1}^v + \mathbf{b}_i^{v0} \tag{4.111}$$

where the new term $\mathbf{b}_i^{v0}$ in (4.111) is a matrix of dimensions $6 \times n$ composed of zeros and with the term $\mathbf{b}_i^v$ assembled in the position of joint $i$ coordinates, being $n$ the number of joint-coordinates.

$$\mathbf{b}_i^{v0} = \begin{bmatrix} 0 & ... & 0 & \mathbf{b}_i^v & 0 & ... & 0 \end{bmatrix} \tag{4.112}$$

Now, taking derivatives of $\mathbf{b}_i^{v1}$ considering equation (4.112) and bearing in mind that $\mathbf{w}_j$ belongs to the first body and the points are contained in the second body, the following derivative is achieved:

$$\begin{aligned} \left(\mathbf{b}_i^{v1}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{v1,1}} \\ \widetilde{\mathbf{b}_i^{v2,1}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_{i-1}^v + \begin{bmatrix} \tilde{\mathbf{w}}_j \\ \mathbf{0} \end{bmatrix} \left( \frac{\partial \mathbf{r}_i}{\partial \mathbf{z}} - \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^v \right) \\ + \begin{bmatrix} \tilde{\mathbf{w}}_j \left( \tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_j \right) \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{b}_i^{v0} \end{aligned} \tag{4.113}$$

With the exposed conditions, different expressions could have been obtained, but for the sake of clearness and uniformity, the derivative is presented with an analog structure to the revolute joint plus a correction term.

The derivative of the second revolute joint takes the form:

$$\left(\mathbf{b}_i^{v2}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{v1,2}} \\ \widetilde{\mathbf{b}_i^{v2,2}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^v + \begin{bmatrix} \tilde{\mathbf{w}}_{j+1} \\ \mathbf{0} \end{bmatrix} \left( \frac{\partial \mathbf{r}_i}{\partial \mathbf{z}} - \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^v \right) \tag{4.114}$$

Depending on the selection of the reference points, these expressions can be reformulated.

**RTdyn0**: For the case of the center of mass of each body as reference point, the following derivatives are obtained:

$$\left(\mathbf{b}_i^{y1}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{y1,1}} \\ \widetilde{\mathbf{b}_i^{y2,1}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_{i-1}^y + \begin{bmatrix} \tilde{\mathbf{w}}_j \left( \tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_j \right) \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{b}_i^{y0} \tag{4.115}$$

$$\left(\mathbf{b}_i^{y2}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{y1,2}} \\ \widetilde{\mathbf{b}_i^{y2,2}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^y \tag{4.116}$$

**RTdyn1**: The selection of the set of reference points matching the global origin of coordinates at each instant of time yields:

$$\left(\mathbf{b}_i^{z1}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{z1,1}} \\ \widetilde{\mathbf{b}_i^{z2,1}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_{i-1}^z + \begin{bmatrix} \tilde{\mathbf{w}}_j \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} -\mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^z + \begin{bmatrix} -\tilde{\mathbf{w}}_j \tilde{\mathbf{r}}_j \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{b}_i^{z0} \tag{4.117}$$

$$\left(\mathbf{b}_i^{z2}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{z1,2}} \\ \widetilde{\mathbf{b}_i^{z2,2}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^z + \begin{bmatrix} \tilde{\mathbf{w}}_{j+1} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} -\mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^z \tag{4.118}$$

- **Cylindrical joint**

  The cylindrical joint can be described as the combination of a prismatic and a revolute joint. In fact, the expression of $\mathbf{b}_i^v$ (2.160a) can be divided into two separated parts: one related to a relative rotation, and other to a translation.

$$\mathbf{b}_i^v = \begin{bmatrix} \mathbf{w}_j & \mathbf{w}_j \wedge \left(\mathbf{r}_i - \mathbf{r}_j\right) \\ \mathbf{0} & \mathbf{w}_j \end{bmatrix} = \begin{bmatrix} \mathbf{b}_i^{v1} & \mathbf{b}_i^{v2} \end{bmatrix} \tag{4.119}$$

The rotation part has the same formula as the term $\mathbf{b}_i^v$ of the revolute joint, while the translation component is equivalent to the term $\mathbf{b}_i^v$ of the prismatic joint. Therefore, since there is no coupling between rotation and translational motions, their derivatives are identical to the ones of the mentioned joints described by equations (4.104) and (4.108):

$$\left(\mathbf{b}_i^{v1}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{v1,1}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^v \tag{4.120}$$

$$\left(\mathbf{b}_i^{v2}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{v1,2}} \\ \widetilde{\mathbf{b}_i^{v2,2}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^v + \begin{bmatrix} \tilde{\mathbf{u}}_j \\ \mathbf{0} \end{bmatrix} \left( \frac{\partial \mathbf{r}_i}{\partial \mathbf{z}} - \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^v \right) \tag{4.121}$$

Observe that the uncoupled relative motion and the internal description of the cylindrical joint through elemental relative and prismatis joints allow a reuse of code in a general implementation, preventing also the inclusion of errors. Similarly to the previous joints, the general expression can be particularized for the two specific accumulations.

**RTdyn0**:

$$\left(\mathbf{b}_i^{y1}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{y1,1}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^y \tag{4.122}$$

$$\left(\mathbf{b}_i^{y2}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{y1,2}} \\ \widetilde{\mathbf{b}_i^{y2,2}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^y \tag{4.123}$$

**RTdyn1**:

$$\left(\mathbf{b}_i^{z1}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{z1,1}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^z \tag{4.124}$$

$$\left(\mathbf{b}_i^{z2}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{z1,2}} \\ \widetilde{\mathbf{b}_i^{z2,2}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^z + \begin{bmatrix} \tilde{\mathbf{u}}_j \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} -\mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^z \tag{4.125}$$

- **Spherical joint**

  The spherical joint is a special case among all the joints included in this work. The term $\mathbf{b}_i^v$ for spherical joints is defined not only as a function of points and vectors, but also of a set of parameters measuring the angular orientation. Furthermore, the Euler parameters used appear in $\mathbf{b}_i^v$ through a term $\mathbf{E}$, which involves the rotation matrix of the previous body. This causes that two different schemes of computation of the derivatives will be present in this term: on the one hand, the differentiation of the reference point and the point defining the joint, given by (3.91) and expressed in terms of $\mathbf{R}^v$; and on the other hand, the derivatives of the rotation matrix $\mathbf{A}_{i-1}$, calculated by (4.79) and which are expressed by means of partial derivatives of relative rotation matrices. This can be seen in the general expression of the derivative of $\mathbf{b}_i^v$ with respect to the relative coordinates:

$$(\mathbf{b}_i^v)_{\hat{\mathbf{z}}} = 2 \begin{bmatrix} \tilde{\mathbf{E}} \left( \dfrac{\partial \mathbf{r}_i}{\partial \mathbf{z}} + \begin{bmatrix} -\mathbf{I} & \tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i \end{bmatrix} \mathbf{R}_i^v \right) + \left( \tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i \right) \left( \mathbf{A}_{i-1} \dfrac{\partial \bar{\mathbf{E}}}{\partial \mathbf{z}} + \dfrac{\partial \mathbf{A}_{i-1}}{\partial \mathbf{z}} \bar{\mathbf{E}} \right) \\ \left( \mathbf{A}_{i-1} \dfrac{\partial \bar{\mathbf{E}}}{\partial \mathbf{z}} + \dfrac{\partial \mathbf{A}_{i-1}}{\partial \mathbf{z}} \bar{\mathbf{E}} \right) \end{bmatrix} \tag{4.126}$$

  where the term $\dfrac{\partial \bar{\mathbf{E}}}{\partial \mathbf{z}}$ solely depends on the Euler parameters of the joint and $\tilde{\mathbf{E}}$ is the skew symmetric tensor[3] of matrix $\mathbf{E}$. The partial derivatives of $\bar{\mathbf{E}}$ with respect to each of the Euler parameters have been described in equations (4.85).

  The derivatives of revolute and prismatic joints described above have been reformulated so as to gather terms multiplied by the topological matrix $\mathbf{R}^v$, since any of the partial derivatives of points and vectors defining the joint can be formulated as expressions involving this matrix (see (3.91)). However, in the case of a spherical joint, the derivative of the rotation matrix (explicitly required in the derivative of $\mathbf{b}_i^v$) is not described in terms of this topological matrix, thus the gathering of terms presented for previous joints cannot be applied here.

  A different possibility for the evaluation of this derivative could be the use of the derivative of the position of a point with respect to the relative coordinates based on the recursive calculation of the positions of points and vectors, instead of the expressions in velocities. To clarify this point, let us consider an open-loop multibody system composed of 2 bodies and the ground, whose relative motion is described by 2 spherical joints. The position of any point at the tip of the kinematic chain can be computed as:

$$\mathbf{r}_2 = \mathbf{r}_0 + \mathbf{A}_1 \left( \bar{\mathbf{r}}_1 - \bar{\mathbf{r}}_0 \right) + \mathbf{A}_2 \left( \bar{\mathbf{r}}_2 - \bar{\mathbf{r}}_1 \right) \tag{4.127}$$

[3]The skew symmetric tensor of a matrix is an operation equivalent to the skew symmetric matrix of a vector. The resulting tensor is composed of the skew symmetric matrices of the columns of the original matrix. The operation is described in appendix B.

being 0 and 1 the points defining the joints and 2 a point on body 2. The derivative of point 2 with respect to the relative coordinates can be calculated in terms of derivatives of rotation matrices as:

$$\frac{\partial \mathbf{r}_2}{\partial \mathbf{z}} = \frac{\partial \mathbf{A}_1}{\partial \mathbf{z}} \left( \bar{\mathbf{r}}_1 - \bar{\mathbf{r}}_0 \right) + \frac{\partial \mathbf{A}_2}{\partial \mathbf{z}} \left( \bar{\mathbf{r}}_2 - \bar{\mathbf{r}}_1 \right) \tag{4.128}$$

Observe that this notation does not allow significant simplifications in (4.126). Moreover, it include terms of local coordinates in the derivatives, which would lead to more complexity in the general expressions. Those were the main reasons for avoiding this method of differentiation and for selecting the one presented in section 3.6 .

The derivative of the reference point can be substituted for the two specific reference point selections, producing similar expressions without meaningful simplifications in any of them.

**RTdyn0**: Applying the derivative of the global position of the reference point (CoM) of the RTdyn0 accumulation (3.93) to equation (4.126), the following expression is reached:

$$(\mathbf{b}_i^y)_{\hat{\mathbf{z}}} = 2 \begin{bmatrix} \tilde{\bar{\mathbf{E}}} \left[ \mathbf{0} \quad \tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i \right] \mathbf{R}_i^y + \left( \tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i \right) \left( \mathbf{A}_{i-1} \dfrac{\partial \bar{\mathbf{E}}}{\partial \mathbf{z}} + \dfrac{\partial \mathbf{A}_{i-1}}{\partial \mathbf{z}} \bar{\mathbf{E}} \right) \\ \left( \mathbf{A}_{i-1} \dfrac{\partial \bar{\mathbf{E}}}{\partial \mathbf{z}} + \dfrac{\partial \mathbf{A}_{i-1}}{\partial \mathbf{z}} \bar{\mathbf{E}} \right) \end{bmatrix} \tag{4.129}$$

**RTdyn1**: In this accumulation, each reference point is selected as fixed in the global reference frame (coincident with the origin of coordinates), thus the derivative of its global position is null. This yields the following derivative of $\mathbf{b}_i^z$:

$$(\mathbf{b}_i^z)_{\hat{\mathbf{z}}} = 2 \begin{bmatrix} \tilde{\bar{\mathbf{E}}} \left[ -\mathbf{I} \quad \tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i \right] \mathbf{R}_i^z + \left( \tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i \right) \left( \mathbf{A}_{i-1} \dfrac{\partial \bar{\mathbf{E}}}{\partial \mathbf{z}} + \dfrac{\partial \mathbf{A}_{i-1}}{\partial \mathbf{z}} \bar{\mathbf{E}} \right) \\ \left( \mathbf{A}_{i-1} \dfrac{\partial \bar{\mathbf{E}}}{\partial \mathbf{z}} + \dfrac{\partial \mathbf{A}_{i-1}}{\partial \mathbf{z}} \bar{\mathbf{E}} \right) \end{bmatrix} \tag{4.130}$$

- **Floating joint**

The floating joint is a very particular type of joint due to the fact that one of the bodies is always the ground. Other peculiarities are the use of fixed vectors to describe the three possible translations and the use of Euler parameters to measure its angular motion. Accordingly, the part of the derivative of $\mathbf{b}_i^v$ related to translations is always null and the part related to the angular motion has the same expression already obtained for the spherical joint:

$$(\mathbf{b}_i^v)_{\hat{\mathbf{z}}} = 2 \begin{bmatrix} \mathbf{0}_{3\times 3\times n} & \tilde{\bar{\bar{\mathbf{E}}}} \left( \dfrac{\partial \mathbf{r}_i}{\partial \mathbf{z}} + \left[ \mathbf{I} \quad \tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_i \right] \mathbf{R}_i^v \right) + \left( \tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_i \right) \dfrac{\partial \bar{\mathbf{E}}}{\partial \mathbf{z}} \\ \mathbf{0}_{3\times 3\times n} & \dfrac{\partial \bar{\mathbf{E}}}{\partial \mathbf{z}} \end{bmatrix} \tag{4.131}$$

It must be remarked that the rotation matrix of the first body of the joint $\mathbf{A}_{i-1}$ along with its derivative disappear form the expressions of the floating joint because the first body is the ground, and its rotation matrix is considered fixed and equal to the identity, thus its derivative with respect to relative coordinates is null.

**RTdyn0**: The particular selection of the origin of the internal spherical joint in the definition of the floating joint, coincident with the CoM of the second body of the joint, makes the expression of $\mathbf{b}_i^v$ particularly simple for RTdyn0, as well as its derivative:

$$(\mathbf{b}_i^y)_{\hat{\mathbf{z}}} = 2 \begin{bmatrix} \mathbf{0}_{3\times3\times n} & \mathbf{0}_{3\times4\times n} \\ \mathbf{0}_{3\times3\times n} & \dfrac{\partial \bar{\mathbf{E}}}{\partial \mathbf{z}} \end{bmatrix} \tag{4.132}$$

**RTdyn1**: In this case, the expression is slightly more complex than in the case of the RTdyn0 formulation:

$$(\mathbf{b}_i^z)_{\hat{\mathbf{z}}} = 2 \begin{bmatrix} \mathbf{0}_{3\times3\times n} & \tilde{\bar{\mathbf{E}}} \begin{bmatrix} \mathbf{I} & \tilde{\mathbf{r}}_G^i \end{bmatrix} \mathbf{R}_i^v + \tilde{\mathbf{r}}_G^i \dfrac{\partial \bar{\mathbf{E}}}{\partial \mathbf{z}} \\ \mathbf{0}_{3\times3\times n} & \dfrac{\partial \bar{\mathbf{E}}}{\partial \mathbf{z}} \end{bmatrix} \tag{4.133}$$

- **Planar joint**

The planar joint is defined by an uncoupled sequence of prismatic and revolute joints, and therefore, $(\mathbf{b}_i^v)_{\hat{\mathbf{z}}}$ can be built with the derivatives of these types of joints. Let us recall equation (2.160a), and divide it in 3 parts:

$$\mathbf{b}_i^v = \begin{bmatrix} \mathbf{u}_j & \mathbf{v}_j & \mathbf{w}_j \wedge (\mathbf{r}_i - \mathbf{r}_G^i) \\ \mathbf{0} & \mathbf{0} & \mathbf{w}_j \end{bmatrix} = \begin{bmatrix} \mathbf{b}_i^{v1} & \mathbf{b}_i^{v2} & \mathbf{b}_i^{v3} \end{bmatrix} \tag{4.134}$$

The derivatives are, therefore:

$$\left(\mathbf{b}_i^{v1}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{v1,1}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_{i-1}^v \tag{4.135}$$

$$\left(\mathbf{b}_i^{v2}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{v1,2}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_{i-1}^v \tag{4.136}$$

$$\left(\mathbf{b}_i^{v3}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{v1,3}} \\ \widetilde{\mathbf{b}_i^{v2,3}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^v + \begin{bmatrix} \tilde{\mathbf{w}}_j \\ \mathbf{0} \end{bmatrix} \left( \frac{\partial \mathbf{r}_i}{\partial \mathbf{z}} - \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^v \right) \tag{4.137}$$

**RTdyn0**: The selection of the rotation point of the revolute joint coincident with the CoM of the second body in the definition of the planar joint simplifies the derivative of $\mathbf{b}_i^v$ in this formulation. Since the derivatives with respect to the prismatic coordinates do not depend explicitly on any reference point, equations (4.135) and (4.136) are equally valid for RTdyn0, while (4.137) becomes:

$$\left(\mathbf{b}_i^{y3}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \mathbf{0} \\ \widetilde{\mathbf{b}_i^{y2,3}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^y \tag{4.138}$$

Behold that the vector or zeros appearing in (4.138) comes from the particular definition of the joint, in which the point considered as the center of the elemental revolute joint is the center of mass of the second body.

**RTdyn1**: Like in the case of the floating joint, the expression of this derivative for RTdyn1 involves more terms. Expressions (4.135) and (4.136) apply also to this formulation, while (4.137) takes the form:

$$\left(\mathbf{b}_i^{z3}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{z1,3}} \\ \widetilde{\mathbf{b}_i^{z2,3}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^z + \begin{bmatrix} \tilde{\mathbf{w}}_j \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} -\mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^z \tag{4.139}$$

### 4.4.4 Evaluation of $\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}}$

In general, the partial derivative of the term $\dot{\mathbf{b}}_i^v$ with respect to the relative coordinates in positions $\mathbf{z}$ can be calculated as the time derivative of the expressions of section 4.4.3:

$$\dot{\mathbf{b}}_i^v = \frac{\mathrm{d}\mathbf{b}_i^v}{\mathrm{d}t} \Rightarrow \left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}} = \frac{\mathrm{d}(\mathbf{b}_i^v)_{\hat{\mathbf{z}}}}{\mathrm{d}t} \tag{4.140}$$

In order to prove (4.140), let us consider separately the temporal derivative of $(\mathbf{b}_i^v)_{\hat{\mathbf{z}}}$ and the derivative of $\dot{\mathbf{b}}_i^v$ with respect to $\mathbf{z}$ using (4.9):

$$\frac{\mathrm{d}(\mathbf{b}_i^v)_{\hat{\mathbf{z}}}}{\mathrm{d}t} = \left((\mathbf{b}_i^v)_{\hat{\mathbf{z}}}\right)_{\hat{\mathbf{z}}} \dot{\mathbf{z}} \tag{4.141a}$$

$$\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}} = ((\mathbf{b}_i^v)_{\hat{\mathbf{z}}}\dot{\mathbf{z}})_{\hat{\mathbf{z}}} = \left((\mathbf{b}_i^v)_{\hat{\mathbf{z}}}\right)_{\hat{\mathbf{z}}} \dot{\mathbf{z}} \tag{4.141b}$$

Behold that (4.141b) and (4.141a) are exactly the same expression, thus (4.140) is validated.

The process of taking temporal derivatives on the expressions of section 4.4.3 is almost straightforward, and accordingly, intermediate developments are omitted hereinafter. The commented expressions of $(\mathbf{b}_i^v)_{\hat{\mathbf{z}}}$ are formulated in terms of global positions of points and vectors, rotation matrices, terms related to Euler parameters and the topological matrix $\mathbf{R}^v$. The temporal derivatives of all these entities have been introduced in chapter 2 (velocities of points and vectors, angular velocities and time derivatives of Euler parameters terms) and in section 3.8 (time derivative of $\mathbf{R}^v$), hence they will be directly applied on the differentiation process.

The particular expressions for RTdyn0 and RTdyn1 are listed in appendices C.1 and C.2 respectively.

- **Revolute joint**

  The derivative $\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}}$ for a revolute joint, calculated as the time derivative of equation (4.104), takes the form:

$$\begin{aligned}\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}} &= \begin{bmatrix}\widetilde{\mathbf{b}_i^{v1,1}}\\ \widetilde{\mathbf{b}_i^{v2,1}}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\dot{\mathbf{R}}_i^v + \begin{bmatrix}\widetilde{\dot{\mathbf{b}}_i^{v1,1}}\\ \widetilde{\dot{\mathbf{b}}_i^{v2,1}}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\mathbf{R}_i^v \\ &+ \begin{bmatrix}\dot{\tilde{\mathbf{w}}}_j\\ \mathbf{0}\end{bmatrix}\left(\frac{\partial \mathbf{r}_i}{\partial \mathbf{z}} - \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\mathbf{R}_i^v\right) + \begin{bmatrix}\tilde{\mathbf{w}}_j\\ \mathbf{0}\end{bmatrix}\left(\frac{\partial \dot{\mathbf{r}}_i}{\partial \mathbf{z}} - \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\dot{\mathbf{R}}_i^v\right)\end{aligned} \tag{4.142}$$

- **Prismatic joint**

  The term $\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}}$ for a prismatic joint can be obtained as the time derivative of (4.108):

$$\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}} = \begin{bmatrix}\widetilde{\dot{\mathbf{b}}_i^{v1,1}}\\ \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\mathbf{R}_i^v + \begin{bmatrix}\widetilde{\mathbf{b}_i^{v1,1}}\\ \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\dot{\mathbf{R}}_i^v \tag{4.143}$$

- **Cardan joint**

  The derivative of $\dot{\mathbf{b}}_i^v$ with respect to joint coordinates for the case of a Cardan joint, evaluated through the temporal derivatives of (4.113) and (4.114), takes the form:

$$\begin{aligned}\left(\dot{\mathbf{b}}_i^{v1}\right)_{\hat{\mathbf{z}}} &= \begin{bmatrix}\widetilde{\dot{\mathbf{b}}_i^{v1,1}}\\ \widetilde{\dot{\mathbf{b}}_i^{v2,1}}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\mathbf{R}_{i-1}^v + \begin{bmatrix}\widetilde{\mathbf{b}_i^{v1,1}}\\ \widetilde{\mathbf{b}_i^{v2,1}}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\dot{\mathbf{R}}_{i-1}^v \\ &+ \begin{bmatrix}\dot{\tilde{\mathbf{w}}}_j\\ \mathbf{0}\end{bmatrix}\left(\frac{\partial \mathbf{r}_i}{\partial \mathbf{z}} - \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\mathbf{R}_i^v\right) + \begin{bmatrix}\tilde{\mathbf{w}}_j\\ \mathbf{0}\end{bmatrix}\left(\frac{\partial \dot{\mathbf{r}}_i}{\partial \mathbf{z}} - \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\dot{\mathbf{R}}_i^v\right) \\ &+ \begin{bmatrix}\dot{\tilde{\mathbf{w}}}_j\left(\tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_j\right) + \tilde{\mathbf{w}}_j\left(\dot{\tilde{\mathbf{r}}}_i - \dot{\tilde{\mathbf{r}}}_j\right)\\ \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\mathbf{b}_i^{v0} + \begin{bmatrix}\tilde{\mathbf{w}}_j\left(\tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_j\right)\\ \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\dot{\mathbf{b}}_i^{v0}\end{aligned} \tag{4.144}$$

$$\begin{aligned}\left(\dot{\mathbf{b}}_i^{v2}\right)_{\hat{\mathbf{z}}} &= \begin{bmatrix}\widetilde{\dot{\mathbf{b}}_i^{v1,2}}\\ \widetilde{\dot{\mathbf{b}}_i^{v2,2}}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\mathbf{R}_i^v + \begin{bmatrix}\widetilde{\mathbf{b}_i^{v1,2}}\\ \widetilde{\mathbf{b}_i^{v2,2}}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\dot{\mathbf{R}}_i^v \\ &+ \begin{bmatrix}\dot{\tilde{\mathbf{w}}}_{j+1}\\ \mathbf{0}\end{bmatrix}\left(\frac{\partial \mathbf{r}_i}{\partial \mathbf{z}} - \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\mathbf{R}_i^v\right) + \begin{bmatrix}\tilde{\mathbf{w}}_{j+1}\\ \mathbf{0}\end{bmatrix}\left(\frac{\partial \dot{\mathbf{r}}_i}{\partial \mathbf{z}} - \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\dot{\mathbf{R}}_i^v\right)\end{aligned} \tag{4.145}$$

- **Cylindrical joint**

  The term of $\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}}$ for a cylindrical can be decomposed into derivatives of the an elemental prismatic joint (4.143) and an elemental revolute joint (4.142), yielding:

$$\left(\dot{\mathbf{b}}_i^{v1}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\dot{\mathbf{b}}_i^{v1,1}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix} \mathbf{R}_i^v + \begin{bmatrix} \widetilde{\mathbf{b}_i^{v1,1}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix} \dot{\mathbf{R}}_i^v \tag{4.146}$$

$$\begin{gathered}\left(\dot{\mathbf{b}}_i^{v2}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\dot{\mathbf{b}}_i^{v1,2}} \\ \widetilde{\dot{\mathbf{b}}_i^{v2,2}} \end{bmatrix} \begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix} \mathbf{R}_i^v + \begin{bmatrix} \widetilde{\mathbf{b}_i^{v1,2}} \\ \widetilde{\mathbf{b}_i^{v2,2}} \end{bmatrix} \begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix} \dot{\mathbf{R}}_i^v \\ + \begin{bmatrix} \dot{\tilde{\mathbf{u}}}_j \\ \mathbf{0} \end{bmatrix} \left( \frac{\partial \mathbf{r}_i}{\partial \mathbf{z}} - \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix} \mathbf{R}_i^v \right) + \begin{bmatrix} \tilde{\mathbf{u}}_j \\ \mathbf{0} \end{bmatrix} \left( \frac{\partial \dot{\mathbf{r}}_i}{\partial \mathbf{z}} - \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix} \dot{\mathbf{R}}_i^v \right)\end{gathered} \tag{4.147}$$

- **Spherical joint**

  The derivatives of the spherical joint are in general the most complex and the ones that involve more terms among all the types of joints studied in the present work. In the following lines, the expression of $\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}}$ is firstly calculated as the time derivative of $\left(\mathbf{b}_i^v\right)_{\hat{\mathbf{z}}}$, and then as the partial derivative of $\dot{\mathbf{b}}_i^v$:

$$\begin{gathered}\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}} = 2 \begin{bmatrix} \dot{\tilde{\mathbf{E}}} \left( \frac{\partial \mathbf{r}_i}{\partial \mathbf{z}} + \begin{bmatrix}-\mathbf{I} & \tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i\end{bmatrix} \mathbf{R}_i^v \right) \\ \mathbf{0} \end{bmatrix} \\ +2 \begin{bmatrix} \tilde{\mathbf{E}} \left( \frac{\partial \dot{\mathbf{r}}_i}{\partial \mathbf{z}} + \begin{bmatrix}\mathbf{0} & \dot{\tilde{\mathbf{r}}}_j - \dot{\tilde{\mathbf{r}}}_i\end{bmatrix} \mathbf{R}_i^v + \begin{bmatrix}-\mathbf{I} & \tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i\end{bmatrix} \dot{\mathbf{R}}_i^v \right) \\ \mathbf{0} \end{bmatrix} \\ +2 \begin{bmatrix} \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i\right) \\ \mathbf{I} \end{bmatrix} \left( \dot{\mathbf{A}}_{i-1} \frac{\partial \bar{\mathbf{E}}}{\partial \mathbf{z}} + \frac{\partial \dot{\mathbf{A}}_{i-1}}{\partial \mathbf{z}} \bar{\mathbf{E}} + \frac{\partial \mathbf{A}_{i-1}}{\partial \mathbf{z}} \dot{\bar{\mathbf{E}}} \right) \\ +2 \begin{bmatrix} \left(\dot{\tilde{\mathbf{r}}}_j - \dot{\tilde{\mathbf{r}}}_i\right) \\ \mathbf{0} \end{bmatrix} \left( \mathbf{A}_{i-1} \frac{\partial \bar{\mathbf{E}}}{\partial \mathbf{z}} + \frac{\partial \mathbf{A}_{i-1}}{\partial \mathbf{z}} \bar{\mathbf{E}} \right)\end{gathered} \tag{4.148}$$

  Observe that the previous scheme based on the direct differentiation of (4.126) with respect to time produces a large expression with several concatenations of sums of matrix and tensor products. This expression can be highly simplified if the temporal derivative $\dot{\mathbf{b}}_i^v$ is calculated first and then its partial derivative is obtained. For the clarity, let us recall expression (2.172), which computes the term $\dot{\mathbf{b}}_i^v$ by means of the angular velocity of the body of the joint and the term $\mathbf{b}_i^v$:

$$\dot{\mathbf{b}}_i^v = \begin{bmatrix} 2\tilde{\boldsymbol{\omega}}_i \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i\right) \mathbf{E} \\ 2\tilde{\boldsymbol{\omega}}_i \mathbf{E} \end{bmatrix} = \begin{bmatrix} \tilde{\boldsymbol{\omega}}_i & \mathbf{0} \\ \mathbf{0} & \tilde{\boldsymbol{\omega}}_i \end{bmatrix} \mathbf{b}_i^v \tag{4.149}$$

Using (4.149), the partial derivative is highly simplified:

$$\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} (\tilde{\boldsymbol{\omega}}_i)_{\hat{\mathbf{z}}} & \mathbf{0} \\ \mathbf{0} & (\tilde{\boldsymbol{\omega}}_i)_{\hat{\mathbf{z}}} \end{bmatrix} \mathbf{b}_i^v + \begin{bmatrix} \tilde{\boldsymbol{\omega}}_i & \mathbf{0} \\ \mathbf{0} & \tilde{\boldsymbol{\omega}}_i \end{bmatrix} (\mathbf{b}_i^v)_{\hat{\mathbf{z}}} \tag{4.150}$$

where $(\mathbf{b}_i^v)_{\hat{\mathbf{z}}}$ has a known expression (equation (4.126)) and $(\tilde{\boldsymbol{\omega}}_i)_{\hat{\mathbf{z}}}$ is the skew symmetric tensor[4] of the matrix $(\boldsymbol{\omega}_i)_{\hat{\mathbf{z}}}$ (with the known expression (4.38)), which can be easily calculated, stored and reused.

- **Floating joint**

  The derivative in the case of the floating joint can be computed analogously to the spherical joint:

$$\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} (\tilde{\boldsymbol{\omega}}_i)_{\hat{\mathbf{z}}} & \mathbf{0} \\ \mathbf{0} & (\tilde{\boldsymbol{\omega}}_i)_{\hat{\mathbf{z}}} \end{bmatrix} \mathbf{b}_i^v + \begin{bmatrix} \tilde{\boldsymbol{\omega}}_i & \mathbf{0} \\ \mathbf{0} & \tilde{\boldsymbol{\omega}}_i \end{bmatrix} \frac{\partial \mathbf{b}_i^v}{\partial \mathbf{z}} \tag{4.151}$$

  Observe that in this case, the matrices and tensors have bigger dimensions, but note also that most of the terms are null. In fact, this derivative in RTdyn0 is completely null (see (C.8)), since the unique dependency of $\dot{\mathbf{b}}_i^v$ is on the velocities of the Euler parameters defining the motion of the joint.

- **Planar joint**

  The derivative $\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}}$ for the planar joint can be directly reached taking temporal derivatives on (4.135), (4.136) and (4.137):

$$\left(\dot{\mathbf{b}}_i^{v1}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\dot{\mathbf{b}}_i^{v1,1}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_{i-1}^v + \begin{bmatrix} \widetilde{\mathbf{b}_i^{v1,1}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \dot{\mathbf{R}}_{i-1}^v \tag{4.152}$$

$$\left(\dot{\mathbf{b}}_i^{v2}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\dot{\mathbf{b}}_i^{v1,2}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_{i-1}^v + \begin{bmatrix} \widetilde{\mathbf{b}_i^{v1,2}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \dot{\mathbf{R}}_{i-1}^v \tag{4.153}$$

$$\begin{aligned} \left(\dot{\mathbf{b}}_i^{v3}\right)_{\hat{\mathbf{z}}} &= \begin{bmatrix} \widetilde{\dot{\mathbf{b}}_i^{v1,3}} \\ \widetilde{\dot{\mathbf{b}}_i^{v2,3}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^v + \begin{bmatrix} \widetilde{\mathbf{b}_i^{v1,3}} \\ \widetilde{\mathbf{b}_i^{v2,3}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \dot{\mathbf{R}}_i^v \\ &+ \begin{bmatrix} \dot{\tilde{\mathbf{w}}}_j \\ \mathbf{0} \end{bmatrix} \left( \frac{\partial \mathbf{r}_i}{\partial \mathbf{z}} - \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^v \right) + \begin{bmatrix} \tilde{\mathbf{w}}_j \\ \mathbf{0} \end{bmatrix} \left( \frac{\partial \dot{\mathbf{r}}_i}{\partial \mathbf{z}} - \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \dot{\mathbf{R}}_i^v \right) \end{aligned} \tag{4.154}$$

[4]The skew symmetric tensor of a matrix is an operation equivalent to the skew symmetric matrix of a vector. The resulting tensor is composed of the skew symmetric matrices of the columns of the original matrix. The operation is described in appendix B.

Although the particularization to RTdyn0 and RTdyn1 accumulations is almost straightforward, an additional difficulty emerges in RTdyn1 expressions. As commented in section 3.6, derivatives of a reference point fixed in the global reference frame are different to the ones of a point rigidly attached to a body. It has been stated that the derivative of the global position of a RTdyn1 reference point is always null, but let us study the derivative of its velocity.

Since the point is fixed in the global reference frame at each instant of time, but the body to which it is associated can move, the point have a non-zero global velocity. Regarding that the reference point moves with the body, and considering that its global position is constant and placed at the global origin of coordinates, its velocity can be computed as:

$$\dot{\mathbf{r}}_0^i = \dot{\mathbf{r}}_j + \tilde{\boldsymbol{\omega}}_i \left(\mathbf{r}_0^i - \mathbf{r}_j\right) = \dot{\mathbf{r}}_j - \tilde{\boldsymbol{\omega}}_i \mathbf{r}_j \tag{4.155}$$

being $j$ a point defined in the rigid body $i$.

Now, taking derivatives on (4.155) with respect to joint coordinates, the following expression for the derivative of the velocity of RTdyn1 reference points is obtained:

$$\frac{\partial \dot{\mathbf{r}}_0^i}{\partial \mathbf{z}} = \frac{\partial \dot{\mathbf{r}}_j}{\partial \mathbf{z}} - \tilde{\boldsymbol{\omega}}_i \frac{\partial \mathbf{r}_j}{\partial \mathbf{z}} + \tilde{\mathbf{r}}_j \Big(\boldsymbol{\omega}_i\Big)_{\hat{\mathbf{z}}} \tag{4.156}$$

Since $j$ is a point fixed in the local reference frame of body $i$, the expression (3.95) and its time derivative (3.96) are valid for this point and can be substituted in (4.156). Applying also the analytical expression of the partial derivative of the angular velocity with respect to relative coordinates (4.38), the following expression in terms of the topological matrix $\mathbf{R}^z$ is obtained:

$$\frac{\partial \dot{\mathbf{r}}_0^i}{\partial \mathbf{z}} = \Big[\mathbf{0} \quad \dot{\tilde{\mathbf{r}}}_0^i - \dot{\tilde{\mathbf{r}}}_j\Big] \mathbf{R}_i^z + \left[\mathbf{I} \quad -\tilde{\mathbf{r}}_j\right] \dot{\mathbf{R}}_i^z - \tilde{\boldsymbol{\omega}}_i \left[\mathbf{I} \quad -\tilde{\mathbf{r}}_j\right] \mathbf{R}_i^z + \tilde{\mathbf{r}}_j \Big(\left[\mathbf{0} \quad \mathbf{I}\right] \dot{\mathbf{R}}_i^z - \left[\mathbf{0} \quad \tilde{\boldsymbol{\omega}}_i\right] \mathbf{R}_i^z\Big) \tag{4.157}$$

Using (4.155) and simplifying, the derivative of $\dot{\mathbf{r}}_0^i$ can be finally described by:

$$\frac{\partial \dot{\mathbf{r}}_0^i}{\partial \mathbf{z}} = \left[\mathbf{I} \quad \mathbf{0}\right] \dot{\mathbf{R}}_i^z - \tilde{\boldsymbol{\omega}}_i \left[\mathbf{I} \quad \mathbf{0}\right] \mathbf{R}_i^z \tag{4.158}$$

### 4.4.5 Evaluation of $\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}}$

The derivative of the term $\dot{\mathbf{b}}_i^v$ is required in both kinematic and dynamic problems. Prior to any particular developments, let us declare that, in general:

$$\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}} \neq \left(\mathbf{b}_i^v\right)_{\hat{\mathbf{z}}} \tag{4.159}$$

Let us consider for instance the case of the reference point equal to the global origin of coordinates used in RTdyn1. In that case, as stated in section 3.6, and more particularly, in equation (3.97), the derivative of its global velocity with respect to joint-coordinate velocities is different to the derivative of its global position with respect to the joint-coordinate positions.

In this regard, the unique problematic term involved in $\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}}$ in which the relation $\dot{(\cdot)}_{\hat{\mathbf{z}}} = (\cdot)_{\hat{\mathbf{z}}}$ is not hold is related to positions and velocities of reference points.

Accordingly, in RTdyn0:

$$\left(\dot{\mathbf{b}}_i^y\right)_{\hat{\mathbf{z}}} = \left(\mathbf{b}_i^y\right)_{\hat{\mathbf{z}}} \tag{4.160}$$

Behold that this is a relevant implementation issue that makes the RTdyn0 approach more attractive than the RTdyn1 approach.

In brief, since the analytical divergence between $\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}}$ and $\left(\mathbf{b}_i^v\right)_{\hat{\mathbf{z}}}$ can be attributed to the selection of the reference point, then, thanks to the separation of derivatives associated to these reference points during the derivation of $\left(\mathbf{b}_i^v\right)_{\hat{\mathbf{z}}}$, expressions of section 4.4.3 can be reused for $\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}}$ with an unique modification, consisting in substituting $\frac{\partial \mathbf{r}_i}{\partial \mathbf{z}}$ by $\frac{\partial \dot{\mathbf{r}}_i}{\partial \dot{\mathbf{z}}}$. With a few operations, the expressions of $\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}}$ for each joint treated in this work can be readily obtained.

For the sake of brevity, and since the application of this change is systematic, the final expressions with the commented substitution are skipped.

The case of RTdyn1 accumulations is still tricky for this calculation. Let us evaluate the derivative of the global velocity of a point coincident with the global origin of coordinates with respect to the joint-coordinate velocities. Taking derivatives on the velocity expression of this particular point (4.155) with respect to joint coordinates at velocity level:

$$\frac{\partial \dot{\mathbf{r}}_0^i}{\partial \dot{\mathbf{z}}} = \frac{\partial \dot{\mathbf{r}}_j}{\partial \dot{\mathbf{z}}} + \tilde{\mathbf{r}}_j \left(\boldsymbol{\omega}_i\right)_{\hat{\mathbf{z}}} \tag{4.161}$$

Now, applying the derivative expressions of the velocity of a point rigidly attached to a body (3.95) and of the angular velocity (4.48) with respect to joint-coordinate velocities, the following analytical formula is reached:

$$\frac{\partial \dot{\mathbf{r}}_0^i}{\partial \dot{\mathbf{z}}} = \begin{bmatrix} \mathbf{I} & -\tilde{\mathbf{r}}_j \end{bmatrix} \mathbf{R}_i^z + \tilde{\mathbf{r}}_j \begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix} \mathbf{R}_i^z = \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^z \tag{4.162}$$

Behold that (4.162) is exactly the same expression of the derivative of the RTdyn1 reference point position as if it was rigidly attached to the body, this is, equal to (3.95) and particularized for the case in which $\mathbf{r}_j = \mathbf{r}_0^i$.

### 4.4.6 Evaluation of $(\mathbf{b}_i^v)_{\hat{\boldsymbol{\rho}}}$

In general, the parameters of any dynamic simulation can be any of the constants or coefficients affecting the dynamic response of the system. Depending on the type of parameter selected, the implications in the analytical derivatives are different, ones affecting only the elemental computation of masses and forces with respect to each body, others related to a group of constraints, and others affecting the whole topology of the mechanism, and therefore, the assembly and accumulation processes.

The kinematic recursive expressions relating the motion between bodies are kinematic expressions created by means of geometrical considerations. Therefore, the term $\mathbf{b}_i^v$ as well as the rest of the recursive terms $\dot{\mathbf{b}}_i^v$, $\mathbf{B}_i^v$, $\dot{\mathbf{B}}_i^v$ and $\mathbf{d}_i^v$ are exclusively affected by geometrical considerations, which can be reduced to:

- A local coordinate of a point in the local reference frame of a body.

- A local coordinate of a vector in the local reference frame of a body.

The term $(\mathbf{b}_i^v)_{\hat{\boldsymbol{\rho}}}$ can be obtained for any of the kinematic joint types described in the present document by means of a three stage process: first, derivatives can be taken with respect to the points and vectors that define the joint (Euler parameters are not considered as sensitivity analysis parameters because they are variables of the problem); the second stage consists in computing the derivative of the global coordinates of the point or vector with respect to the parameters. Third, the derivative of $\mathbf{b}_i^v$ can be evaluated applying the chain rule of differentiation.

The first part of the process, consisting in taking derivatives with respect to global coordinates of points and vectors is skipped here since the differentiation is immediate and no significant simplifications can be achieved. The second stage is valid for any point and vector of the model belonging to a body, and will be explained in detail in section 4.5.7.

The decomposition of the derivation process is primarily used because of the ubiquitous presence of the global coordinates of points and vectors along the assembly of forces, masses and constraints, as well as to avoid the generation of new different expressions for every new term involving $\mathbf{q}_{\boldsymbol{\rho}}$.

The derivative of the recursive terms of the revolute joint with respect to the local coordinates of a point $k$ of a body $l$ is given as an example of this computation:

$$(\mathbf{b}_i^v)_{\hat{\boldsymbol{\rho}}} = \begin{bmatrix} \tilde{\mathbf{w}}_j \left( \dfrac{\partial \mathbf{r}_i}{\partial \bar{\mathbf{r}}_k^l} - \dfrac{\partial \mathbf{r}_j}{\partial \bar{\mathbf{r}}_k^l} \right) + \left( \tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i \right) \dfrac{\partial \mathbf{w}_j}{\partial \bar{\mathbf{r}}_k^l} \\ \dfrac{\partial \mathbf{w}_j}{\partial \bar{\mathbf{r}}_k^l} \end{bmatrix} \tag{4.163}$$

in which derivatives $\dfrac{\partial \mathbf{r}_i}{\partial \bar{\mathbf{r}}_k^l}$ and $\dfrac{\partial \mathbf{w}_j}{\partial \bar{\mathbf{r}}_k^l}$ will be addressed in section 4.5.7.

### 4.4.7 Evaluation of $\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\boldsymbol{\rho}}}$

Analogously to $(\mathbf{b}_i^v)_{\hat{\boldsymbol{\rho}}}$, the derivative $\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\boldsymbol{\rho}}}$ can be executed in a two stage process: first, the derivatives with respect to positions and velocities of the points and vectors defining the joint can be computed; secondly, the partial derivatives of these positions and velocities with respect to the parameters of the system can be evaluated; and in third place, $\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\boldsymbol{\rho}}}$ can be obtained applying the chain rule of differentiation.

For the example case of the revolute joint presented in section 4.4.6, $\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\boldsymbol{\rho}}}$ becomes:

$$\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\boldsymbol{\rho}}} = \begin{bmatrix} \tilde{\mathbf{w}}_j \left(\frac{\partial \dot{\mathbf{r}}_i}{\partial \bar{\mathbf{r}}_k^l} - \frac{\partial \dot{\mathbf{r}}_j}{\partial \bar{\mathbf{r}}_k^l}\right) + \dot{\tilde{\mathbf{w}}}_j \left(\frac{\partial \mathbf{r}_i}{\partial \bar{\mathbf{r}}_k^l} - \frac{\partial \mathbf{r}_j}{\partial \bar{\mathbf{r}}_k^l}\right) + \left(\dot{\tilde{\mathbf{r}}}_j - \dot{\tilde{\mathbf{r}}}_i\right) \frac{\partial \mathbf{w}_j}{\partial \bar{\mathbf{r}}_k^l} + \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i\right) \frac{\partial \dot{\mathbf{w}}_j}{\partial \bar{\mathbf{r}}_k^l} \\ \frac{\partial \dot{\mathbf{w}}_j}{\partial \bar{\mathbf{r}}_k^l} \end{bmatrix} \tag{4.164}$$

wherein the partial derivatives of positions and velocities of points and vectors with respect to local coordinates of points will be addressed in section 4.5.7.

The process for reaching the derivative of $\dot{\mathbf{b}}_i^v$ with respect to the parameters for every kinematic joint type is analog to the revolute joint case presented in (4.164), and therefore, the particular expressions for each joint type will be omitted here.

### 4.4.8 Evaluation of $(\mathbf{B}_i^v)_{\hat{\mathbf{z}}}$

$\mathbf{B}_i^v$ is a term required in a recursive accumulation for the translation of kinematic relations between subsequent bodies. Its mission is to express kinematic magnitudes in terms of the reference point of the subsequent body, thus it is essential for a kinematic forward open-loop evaluation and for the backwards composition (or accumulation) of masses and generalized forces vector in semi-recursive (or fully-recursive) formulations.

For simplicity, let us consider the calculation of the product of this derivative by a given array $\mathbf{x} = \begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2\end{bmatrix}^{\mathrm{T}}$ with $\mathbf{x} \in \mathbb{R}^6$ and $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^3$. According to the general analytical expression of $\mathbf{B}_i^v$ given by (2.121c) for any selection of reference points, its derivative takes the form:

$$(\mathbf{B}_i^v)_{\hat{\mathbf{z}}}\mathbf{x} = (\mathbf{B}_i^v)_{\hat{\mathbf{z}}} \begin{bmatrix}\mathbf{x}_1 \\ \mathbf{x}_2\end{bmatrix} = \begin{bmatrix} \left(\frac{\partial \tilde{\mathbf{r}}_{i-1}}{\partial \mathbf{z}} - \frac{\partial \tilde{\mathbf{r}}_i}{\partial \mathbf{z}}\right)\mathbf{x}_2 \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{x}}_2 \left(\frac{\partial \mathbf{r}_i}{\partial \mathbf{z}} - \frac{\partial \mathbf{r}_{i-1}}{\partial \mathbf{z}}\right) \\ \mathbf{0} \end{bmatrix} \tag{4.165}$$

Behold that no further simplifications can be applied into (4.165) since the derivative of each reference point is undetermined in a general case.

For RTdyn0 accumulations, since each reference point belongs to each body, this is, is fixed in the local reference frame of each body, expression (3.91) can be substituted into (4.165), producing:

$$(\mathbf{B}_i^y)_{\hat{\mathbf{z}}}\mathbf{x} = (\mathbf{B}_i^y)_{\hat{\mathbf{z}}} \begin{bmatrix}\mathbf{x}_1 \\ \mathbf{x}_2\end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{x}}_2 \left(\begin{bmatrix}\mathbf{0} & \tilde{\mathbf{r}}_G^{i-1} - \tilde{\mathbf{r}}_G^{i}\end{bmatrix} \mathbf{R}_{i-1}^y + \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix} \mathbf{b}_i^{y0}\right) \\ \mathbf{0} \end{bmatrix} \tag{4.166}$$

Observe that the expression is already simplified using the relations of the assembly of $\mathbf{R}^y$, exposed in equations (4.111) and (4.112).

The case of RTdyn1 is completely different due to the fact that the global position of each body in the global reference frame is fixed, thus $\mathbf{B}_i^z$ is constant and equal to the identity matrix (see (2.123c)), and the derivative with respect to joint coordinates is null.

### 4.4.9 Evaluation of $\left(\dot{\mathbf{B}}_i^v\right)_{\hat{\mathbf{z}}}$

Similarly to section 4.4.8, it is clearer to express this derivative multiplied by an array $\mathbf{x} = \begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2\end{bmatrix}^{\mathrm{T}}$ with $\mathbf{x} \in \mathbb{R}^6$ and $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^3$. The general case for an arbitrary selection of reference points takes the form:

$$\left(\dot{\mathbf{B}}_i^v\right)_{\hat{\mathbf{z}}} \mathbf{x} = \left(\dot{\mathbf{B}}_i^v\right)_{\hat{\mathbf{z}}} \begin{bmatrix}\mathbf{x}_1 \\ \mathbf{x}_2\end{bmatrix} = \begin{bmatrix}\left(\dfrac{\partial \dot{\tilde{\mathbf{r}}}_{i-1}}{\partial \mathbf{z}} - \dfrac{\partial \dot{\tilde{\mathbf{r}}}_i}{\partial \mathbf{z}}\right) \mathbf{x}_2 \\ \mathbf{0}\end{bmatrix} = \begin{bmatrix}\tilde{\mathbf{x}}_2 \left(\dfrac{\partial \dot{\mathbf{r}}_i}{\partial \mathbf{z}} - \dfrac{\partial \dot{\mathbf{r}}_{i-1}}{\partial \mathbf{z}}\right) \\ \mathbf{0}\end{bmatrix} \tag{4.167}$$

As commented in section 4.4.8, expression (4.167) cannot be further developed since the derivative of the velocity of each reference point depends on the type of point selected.

Let us consider the case of RTdyn0. If each reference point is fixed in the local reference frame of each body, like in RTdyn0, the expression of the present derivative will be:

$$\left(\dot{\mathbf{B}}_i^y\right)_{\hat{\mathbf{z}}} \mathbf{x} = \left(\dot{\mathbf{B}}_i^y\right)_{\hat{\mathbf{z}}} \begin{bmatrix}\mathbf{x}_1 \\ \mathbf{x}_2\end{bmatrix} = \begin{bmatrix}\tilde{\mathbf{x}}_2 \left(\begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix} \dot{\mathbf{R}}_i^y - \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix} \dot{\mathbf{R}}_{i-1}^y\right) \\ \mathbf{0}\end{bmatrix} \tag{4.168}$$

Besides, if the reference point is fixed in the global frame, like in RTdyn1, the derivative will change significantly. The substitution of the derivative of the velocity of the RTdyn1 reference point given by (4.158) into (4.167) yields:

$$\left(\dot{\mathbf{B}}_i^z\right)_{\hat{\mathbf{z}}} \mathbf{x} = \left(\dot{\mathbf{B}}_i^z\right)_{\hat{\mathbf{z}}} \begin{bmatrix}\mathbf{x}_1 \\ \mathbf{x}_2\end{bmatrix} = \begin{bmatrix}\tilde{\mathbf{x}}_2 \left(\begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix} \dot{\mathbf{R}}_i^z - \begin{bmatrix}\tilde{\boldsymbol{\omega}}_i & \mathbf{0}\end{bmatrix} \mathbf{R}_i^z - \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix} \dot{\mathbf{R}}_{i-1}^z + \begin{bmatrix}\tilde{\boldsymbol{\omega}}_{i-1} & \mathbf{0}\end{bmatrix} \mathbf{R}_{i-1}^z\right) \\ \mathbf{0}\end{bmatrix} \tag{4.169}$$

Note that neither RTdyn0 nor RTdyn1 is the best option to evaluate every dynamic derivative. In this case, for example, the RTdyn1 expression is more involved than the RTdyn0 one, but in 4.4.8, the opposite occurs.

### 4.4.10 Evaluation of $\left(\dot{\mathbf{B}}_i^v\right)_{\hat{\dot{\mathbf{z}}}}$

Regarding that the equality $\left(\dot{\mathbf{B}}_i^v\right)_{\hat{\dot{\mathbf{z}}}} = \left(\mathbf{B}_i^v\right)_{\hat{\mathbf{z}}}$ cannot be assumed for an arbitrary selection of reference points, special attention should be paid to the evaluation of

$\left(\dot{\mathbf{B}}_i^v\right)_{\hat{\dot{\mathbf{z}}}}$. In brief, considering this derivative multiplied by an array $\mathbf{x} = \begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2\end{bmatrix}^{\mathrm{T}}$ with $\mathbf{x} \in \mathbb{R}^6$ and $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^3$, it takes the form:

$$\left(\dot{\mathbf{B}}_i^v\right)_{\hat{\dot{\mathbf{z}}}}\mathbf{x} = \left(\dot{\mathbf{B}}_i^v\right)_{\hat{\dot{\mathbf{z}}}}\begin{bmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{bmatrix} = \begin{bmatrix}\left(\dfrac{\partial \tilde{\dot{\mathbf{r}}}_{i-1}}{\partial \dot{\mathbf{z}}} - \dfrac{\partial \tilde{\dot{\mathbf{r}}}_i}{\partial \dot{\mathbf{z}}}\right)\mathbf{x}_2\\ \mathbf{0}\end{bmatrix} = \begin{bmatrix}\tilde{\mathbf{x}}_2\left(\dfrac{\partial \dot{\mathbf{r}}_i}{\partial \dot{\mathbf{z}}} - \dfrac{\partial \dot{\mathbf{r}}_{i-1}}{\partial \dot{\mathbf{z}}}\right)\\ \mathbf{0}\end{bmatrix} \tag{4.170}$$

Let us briefly mention that the differentiation procedure is analog to the one developed in sections 4.4.8 and 4.4.9, and that the substitution of the derivative of the velocity of each particular reference point has to be addressed according to its nature. For the specific versions of the accumulation process, equation (3.93) should be employed in RTdyn0 and equation (4.162) in RTdyn1, yielding:

$$\left(\dot{\mathbf{B}}_i^y\right)_{\hat{\dot{\mathbf{z}}}}\mathbf{x} = \left(\mathbf{B}_i^y\right)_{\hat{\mathbf{z}}}\mathbf{x} = \begin{bmatrix}\tilde{\mathbf{x}}_2\left(\begin{bmatrix}\mathbf{0} & \tilde{\mathbf{r}}_G^{i-1} - \tilde{\mathbf{r}}_G^{i}\end{bmatrix}\mathbf{R}_{i-1}^y + \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\mathbf{b}_i^{y0}\right)\\ \mathbf{0}\end{bmatrix} \tag{4.171}$$

and

$$\left(\dot{\mathbf{B}}_i^z\right)_{\hat{\dot{\mathbf{z}}}}\mathbf{x} = \begin{bmatrix}\tilde{\mathbf{x}}_2\left(\begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\mathbf{b}_i^{z0}\right)\\ \mathbf{0}\end{bmatrix} \tag{4.172}$$

wherein the recursive relations (4.111) and (4.112) have been applied

## 4.4.11 Evaluation of $(\mathbf{B}_i^v)_{\hat{\boldsymbol{\rho}}}$

The expression of $\mathbf{B}_i^v$ is formulated exclusively in terms of global positions of reference points. Depending on whether this global position is affected or not by a parameter, the partial derivative $(\mathbf{B}_i^v)_{\hat{\boldsymbol{\rho}}}$ takes different expressions.

The general expression of this term multiplied by an array $\mathbf{x} = \begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2\end{bmatrix}^{\mathrm{T}}$ with $\mathbf{x} \in \mathbb{R}^6$ and $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^3$ takes the form:

$$(\mathbf{B}_i^v)_{\hat{\boldsymbol{\rho}}}\mathbf{x} = (\mathbf{B}_i^v)_{\hat{\boldsymbol{\rho}}}\begin{bmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{bmatrix} = \begin{bmatrix}\tilde{\mathbf{x}}_2\left(\dfrac{\partial \mathbf{r}_i}{\partial \boldsymbol{\rho}} - \dfrac{\partial \mathbf{r}_{i-1}}{\partial \boldsymbol{\rho}}\right)\\ \mathbf{0}\end{bmatrix} \tag{4.173}$$

In RTdyn0 formulations, $\dfrac{\partial \mathbf{r}_i}{\partial \boldsymbol{\rho}}$ can be evaluated through expressions further discussed in section 4.5.7. On the contrary, $\mathbf{B}_i^v$ is the identity in the RTdyn1 version, thus its derivative is null.

## 4.4.12 Evaluation of $\left(\dot{\mathbf{B}}_i^v\right)_{\hat{\boldsymbol{\rho}}}$

Considering this derivative multiplied by an array $\mathbf{x} = \begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2\end{bmatrix}^{\mathrm{T}}$ with $\mathbf{x} \in \mathbb{R}^6$ and $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^3$, it can be expressed as:

$$\left(\dot{\mathbf{B}}_i^v\right)_{\hat{\boldsymbol{\rho}}}\mathbf{x} = \left(\dot{\mathbf{B}}_i^v\right)_{\hat{\boldsymbol{\rho}}}\begin{bmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{bmatrix} = \begin{bmatrix}\tilde{\mathbf{x}}_2\left(\dfrac{\partial \dot{\mathbf{r}}_i}{\partial \boldsymbol{\rho}} - \dfrac{\partial \dot{\mathbf{r}}_{i-1}}{\partial \boldsymbol{\rho}}\right)\\ \mathbf{0}\end{bmatrix} \tag{4.174}$$

As commented in section 4.4.11, RTdyn0 expressions for $\dfrac{\partial \mathbf{r}_i}{\partial \boldsymbol{\rho}}$ will be introduced in section 4.5.8.

The RTdyn1 case, once again, requires a particularization of this derivative. Let us take partial derivatives with respect to a set of parameters of the expression of the velocity of a point placed at the global origin of coordinates (4.155):

$$\frac{\partial \dot{\mathbf{r}}_0^i}{\partial \boldsymbol{\rho}} = \frac{\partial \dot{\mathbf{r}}_j}{\partial \boldsymbol{\rho}} - \tilde{\boldsymbol{\omega}}_i \frac{\partial \mathbf{r}_j}{\partial \boldsymbol{\rho}} + \tilde{\mathbf{r}}_j \Big(\boldsymbol{\omega}_i\Big)_{\hat{\boldsymbol{\rho}}} \tag{4.175}$$

Since, equation (4.175) entails derivatives of other terms not considered yet, it is easier to compute this derivative as a result of the recursive kinematics evaluation required in both semi-recursive and fully-recursive methods. Recalling (2.123a) and taking partial derivatives:

$$(\mathbf{Z}_i)_{\hat{\boldsymbol{\rho}}} = (\mathbf{B}_i^z)_{\hat{\boldsymbol{\rho}}} \mathbf{Z}_{i-1} + \mathbf{B}_i^z (\mathbf{Z}_{i-1})_{\hat{\boldsymbol{\rho}}} + (\mathbf{b}_i^z)_{\hat{\boldsymbol{\rho}}} \dot{\mathbf{z}}_i \tag{4.176}$$

in which:

$$\frac{\partial \dot{\mathbf{r}}_0^i}{\partial \boldsymbol{\rho}} = \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix} (\mathbf{Z}_i)_{\hat{\boldsymbol{\rho}}} \tag{4.177}$$

Behold that this process can be used likewise for any selection of reference points, but in the case of RTdyn0 formulations, expressions developed on section 4.5.8 will be more efficient.

### 4.4.13 Evaluation of $(\mathbf{d}_i^v)_{\hat{\mathbf{z}}}$

The term $\mathbf{d}_i^v$ is a magnitude related to the transmission of linear and angular accelerations in kinematic joints. As presented in (2.121e), its expression is independent of the type of joint involved, even though particular expressions can be also reached for each joint type, as presented in section 2.2. In this section and in the MBSLIM implementation, only the generic expression has been considered, with the consequent reduction in the implementation effort and a similar computational expense.

First of all, let us recall (2.121e) and substitute $\dot{\mathbf{B}}_i^v$ by its expanded expression (2.116) and $\mathbf{V}_{i-1}$ by $\begin{bmatrix}\dot{\mathbf{r}}_i & \boldsymbol{\omega}_i\end{bmatrix}^{\mathrm{T}}$:

$$\mathbf{d}_i^v = \dot{\mathbf{B}}_i^v \mathbf{V}_{i-1} + \dot{\mathbf{b}}_i^v \dot{\mathbf{z}}_i = \begin{bmatrix}\mathbf{0} & \dot{\tilde{\mathbf{r}}}_{i-1} - \dot{\tilde{\mathbf{r}}}_i \\ \mathbf{0} & \mathbf{0}\end{bmatrix} \begin{bmatrix}\dot{\mathbf{r}}_{i-1} \\ \boldsymbol{\omega}_{i-1}\end{bmatrix} + \dot{\mathbf{b}}_i^v \dot{\mathbf{z}}_i = \begin{bmatrix}\left(\dot{\tilde{\mathbf{r}}}_{i-1} - \dot{\tilde{\mathbf{r}}}_i\right) \boldsymbol{\omega}_{i-1} \\ \mathbf{0}\end{bmatrix} + \dot{\mathbf{b}}_i^v \dot{\mathbf{z}}_i \tag{4.178}$$

Thus, term $(\mathbf{d}_i^v)_{\hat{\mathbf{z}}}$ can be expressed as:

$$(\mathbf{d}_i^v)_{\hat{\mathbf{z}}} = \begin{bmatrix}\left(\dot{\tilde{\mathbf{r}}}_{i-1} - \dot{\tilde{\mathbf{r}}}_i\right) (\boldsymbol{\omega}_{i-1})_{\hat{\mathbf{z}}} - \tilde{\boldsymbol{\omega}}_{i-1} \left(\dfrac{\partial \dot{\tilde{\mathbf{r}}}_{i-1}}{\partial \mathbf{z}} - \dfrac{\partial \dot{\tilde{\mathbf{r}}}_i}{\partial \mathbf{z}}\right) \\ \mathbf{0}\end{bmatrix} + \left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}} \dot{\mathbf{z}}_i \tag{4.179}$$

Since the derivative of each reference point velocity depends on the set of reference points selected, only the derivative of the angular velocity (4.38) can be substituted in (4.179):

$$(\mathbf{d}_i^v)_{\hat{\mathbf{z}}} = \begin{bmatrix} \left(\dot{\tilde{\mathbf{r}}}_{i-1} - \dot{\tilde{\mathbf{r}}}_i\right)\left(\begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix}\dot{\mathbf{R}}_{i-1}^v - \begin{bmatrix}\mathbf{0} & \tilde{\boldsymbol{\omega}}_{i-1}\end{bmatrix}\mathbf{R}_{i-1}^v\right) - \tilde{\boldsymbol{\omega}}_{i-1}\left(\dfrac{\partial \dot{\tilde{\mathbf{r}}}_{i-1}}{\partial \mathbf{z}} - \dfrac{\partial \dot{\tilde{\mathbf{r}}}_i}{\partial \mathbf{z}}\right) \\ \mathbf{0} \end{bmatrix} + \left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}} \dot{\mathbf{z}}_i \tag{4.180}$$

Note that (4.180) cannot be further simplified without resorting to the particular expression of $\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\mathbf{z}}}$ for each kinematic joint type.

In RTdyn0 accumulations, each reference point selected coincides with the center of mass of each body. These points are fixed in the local reference frame of each body, thus the derivative of their velocity with respect to joint coordinates in positions can be obtained from equation (3.98). The substitution of (3.98) in (4.180) yields:

$$(\mathbf{d}_i^y)_{\hat{\mathbf{z}}} = \begin{bmatrix} \begin{bmatrix}\tilde{\boldsymbol{\omega}}_{i-1} & \mathbf{0}\end{bmatrix}\dot{\mathbf{R}}_i^y - \begin{bmatrix}\tilde{\boldsymbol{\omega}}_{i-1} & \dot{\tilde{\mathbf{r}}}_i - \dot{\tilde{\mathbf{r}}}_{i-1}\end{bmatrix}\dot{\mathbf{R}}_{i-1}^y + \begin{bmatrix}\mathbf{0} & \left(\dot{\tilde{\mathbf{r}}}_i - \dot{\tilde{\mathbf{r}}}_{i-1}\right)\tilde{\boldsymbol{\omega}}_{i-1}\end{bmatrix}\mathbf{R}_{i-1}^y \\ \mathbf{0} \end{bmatrix} + \left(\dot{\mathbf{b}}_i^y\right)_{\hat{\mathbf{z}}} \dot{\mathbf{z}}_i \tag{4.181}$$

The expression of this derivative in the RTdyn1 formulations is different due to the different type of point used as reference point. Applying (4.158) to (4.179), this derivative becomes:

$$\begin{aligned}(\mathbf{d}_i^z)_{\hat{\mathbf{z}}} = &\begin{bmatrix} \begin{bmatrix}\tilde{\boldsymbol{\omega}}_{i-1} & \mathbf{0}\end{bmatrix}\dot{\mathbf{R}}_i^z - \begin{bmatrix}\tilde{\boldsymbol{\omega}}_{i-1} & \dot{\tilde{\mathbf{r}}}_i - \dot{\tilde{\mathbf{r}}}_{i-1}\end{bmatrix}\dot{\mathbf{R}}_{i-1}^z + \begin{bmatrix}\mathbf{0} & \left(\dot{\tilde{\mathbf{r}}}_i - \dot{\tilde{\mathbf{r}}}_{i-1}\right)\tilde{\boldsymbol{\omega}}_{i-1}\end{bmatrix}\mathbf{R}_{i-1}^z \\ \mathbf{0} \end{bmatrix} \\ &+ \tilde{\boldsymbol{\omega}}_{i-1}\begin{bmatrix} \left(\tilde{\boldsymbol{\omega}}_{i-1} - \tilde{\boldsymbol{\omega}}_i\right)\begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\mathbf{R}_{i-1}^z - \tilde{\boldsymbol{\omega}}_i\begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\mathbf{b}_i^{z0} \\ \mathbf{0} \end{bmatrix} + \left(\dot{\mathbf{b}}_i^z\right)_{\hat{\mathbf{z}}} \dot{\mathbf{z}}_i\end{aligned} \tag{4.182}$$

Comparing (4.182) with the equivalent RTdyn0 derivative (4.181), it can be seen that an additional term is required in RTdyn1, related to the relative motion of the reference point in the local reference frame of each body. In fact, both expressions can be reformulated, but the expressions presented allow a direct comparison of terms and of the effects of the reference point selection.

In semi-recursive formulations, a term $\mathbf{d}_i^{v\Sigma}$ emerges in the composition of the vector of generalized forces $\mathbf{Q}^d$ and its derivative appears during the evaluation of the stiffness matrix of the model (4.34b). Taking derivatives on (2.215), the derivative of the accumulated term $\mathbf{d}_i^{v\Sigma}$ can be expressed as:

$$\left(\mathbf{d}_i^{v\Sigma}\right)_{\hat{\mathbf{z}}} = \left(\mathbf{d}_i^{v}\right)_{\hat{\mathbf{z}}} + \left(\mathbf{B}_i^{v}\right)_{\hat{\mathbf{z}}} \mathbf{d}_h^{v\Sigma} + \mathbf{B}_i^{v} \left(\mathbf{d}_h^{v\Sigma}\right)_{\hat{\mathbf{z}}} \tag{4.183}$$

being $h$ the preceding body in the kinematic chain.

The particularization of (4.183) for RTdyn0 generates a similar expression:

$$\left(\mathbf{d}_i^{y\Sigma}\right)_{\hat{\mathbf{z}}} = \left(\mathbf{d}_i^{y}\right)_{\hat{\mathbf{z}}} + \left(\mathbf{B}_i^{y}\right)_{\hat{\mathbf{z}}} \mathbf{d}_h^{y\Sigma} + \mathbf{B}_i^{y} \left(\mathbf{d}_h^{y\Sigma}\right)_{\hat{\mathbf{z}}} \tag{4.184}$$

For RTdyn1, the general assembly of $\left(\mathbf{d}_i^{v\Sigma}\right)_{\hat{\mathbf{z}}}$ is more direct:

$$\left(\mathbf{d}_i^{z\Sigma}\right)_{\hat{\mathbf{z}}} = \left(\mathbf{d}_i^{z}\right)_{\hat{\mathbf{z}}} + \left(\mathbf{d}_h^{z\Sigma}\right)_{\hat{\mathbf{z}}} \tag{4.185}$$

### 4.4.14 Evaluation of $\left(\mathbf{d}_i^{v}\right)_{\hat{\mathbf{z}}}$

Recalling equation (2.121e), it can be regarded that $\mathbf{d}_i^{v}$ depends on positions but also on velocities of the joint coordinates vector. The derivative with respect to $\dot{\mathbf{z}}$ can be expressed as:

$$\begin{aligned}
\left(\mathbf{d}_i^{v}\right)_{\hat{\mathbf{z}}} &= \left(\dot{\mathbf{B}}_i^{v}\mathbf{V}_{i-1} + \dot{\mathbf{b}}_i^{v}\dot{\mathbf{z}}_i\right)_{\hat{\mathbf{z}}} = \left(\begin{bmatrix} \left(\dot{\tilde{\mathbf{r}}}_{i-1} - \dot{\tilde{\mathbf{r}}}_i\right) \boldsymbol{\omega}_{i-1} \\ \mathbf{0} \end{bmatrix} + \dot{\mathbf{b}}_i^{v}\dot{\mathbf{z}}_i\right)_{\hat{\mathbf{z}}} = \\
&\begin{bmatrix} \left(\dot{\tilde{\mathbf{r}}}_{i-1} - \dot{\tilde{\mathbf{r}}}_i\right) \left(\boldsymbol{\omega}_{i-1}\right)_{\hat{\mathbf{z}}} - \tilde{\boldsymbol{\omega}}_{i-1} \left(\dfrac{\partial \dot{\mathbf{r}}_{i-1}}{\partial \dot{\mathbf{z}}} - \dfrac{\partial \dot{\mathbf{r}}_{i}}{\partial \dot{\mathbf{z}}}\right) \\ \mathbf{0} \end{bmatrix} + \left(\dot{\mathbf{b}}_i^{v}\right)_{\hat{\mathbf{z}}} \dot{\mathbf{z}}_i + \dot{\mathbf{b}}_i^{v0} = \\
&\begin{bmatrix} \left(\dot{\tilde{\mathbf{r}}}_{i-1} - \dot{\tilde{\mathbf{r}}}_i\right) \begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix} \mathbf{R}_{i-1}^{v} - \tilde{\boldsymbol{\omega}}_{i-1} \left(\dfrac{\partial \dot{\mathbf{r}}_{i-1}}{\partial \dot{\mathbf{z}}} - \dfrac{\partial \dot{\mathbf{r}}_{i}}{\partial \dot{\mathbf{z}}}\right) \\ \mathbf{0} \end{bmatrix} + \left(\dot{\mathbf{b}}_i^{v}\right)_{\hat{\mathbf{z}}} \dot{\mathbf{z}}_i + \dot{\mathbf{b}}_i^{v0}
\end{aligned} \tag{4.186}$$

in which $\dot{\mathbf{b}}_i^{v0} \in \mathbb{R}^{6\times n}$ (with $n$ the number of joint coordinates) is defined as:

$$\dot{\mathbf{b}}_i^{v0} = \dot{\mathbf{b}}_i^{v} \frac{\partial \dot{\mathbf{z}}_i}{\partial \dot{\mathbf{z}}} = \begin{bmatrix} 0 & ... & 0 & \dot{\mathbf{b}}_i^{v} & 0 & ... & 0 \end{bmatrix} \tag{4.187}$$

For the particular reference point selection of RTdyn0, (4.186) becomes:

$$\begin{aligned}
\left(\mathbf{d}_i^{y}\right)_{\hat{\mathbf{z}}} = &\begin{bmatrix} \left(\dot{\tilde{\mathbf{r}}}_G^{i-1} - \dot{\tilde{\mathbf{r}}}_G^{i} + \tilde{\boldsymbol{\omega}}_{i-1}\left(\tilde{\mathbf{r}}_G^{i-1} - \tilde{\mathbf{r}}_G^{i}\right)\right) \begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix} \mathbf{R}_{i-1}^{y} + \tilde{\boldsymbol{\omega}}_{i-1} \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix} \mathbf{b}_i^{y0} \\ \mathbf{0} \end{bmatrix} \\
&+ \dot{\mathbf{b}}_i^{y0} + \left(\dot{\mathbf{b}}_i^{y}\right)_{\hat{\mathbf{z}}} \dot{\mathbf{z}}_i
\end{aligned} \tag{4.188}$$

with $\mathbf{b}_i^{y0}$ given by (4.112).

In RTdyn1, applying (4.162) to (4.186), the following expression is obtained:

$$(\mathbf{d}_i^z)_{\hat{\mathbf{z}}} = \begin{bmatrix} \left(\dot{\tilde{\mathbf{r}}}_0^{i-1} - \dot{\tilde{\mathbf{r}}}_0^{i}\right)\begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix}\mathbf{R}_{i-1}^z + \tilde{\boldsymbol{\omega}}_{i-1}\begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\mathbf{b}_i^{z0} \\ \mathbf{0} \end{bmatrix} + \dot{\mathbf{b}}_i^{z0} + \left(\dot{\mathbf{b}}_i^z\right)_{\hat{\mathbf{z}}}\dot{\mathbf{z}}_i \tag{4.189}$$

Behold that the evaluation of $\left(\dot{\mathbf{b}}_i^z\right)_{\hat{\mathbf{z}}}$ in RTdyn1 as it has been described in section 4.4.5 is of special relevance, whereas its substitution by $(\mathbf{b}_i^z)_{\hat{\mathbf{z}}}$ in this formulation would lead to erroneous expressions and results.

The accumulation term $\mathbf{d}_i^{v\Sigma}$ originated by the semi-recursive method described in section 2.4.1 can be differentiated with respect to joint coordinate velocities, which yields:

$$\left(\mathbf{d}_i^{v\Sigma}\right)_{\hat{\mathbf{z}}} = (\mathbf{d}_i^v)_{\hat{\mathbf{z}}} + \mathbf{B}_i^v\left(\mathbf{d}_h^{v\Sigma}\right)_{\hat{\mathbf{z}}} \tag{4.190}$$

where $h$ is the parent body of $i$.

For RTdyn0, (4.190) becomes:

$$\left(\mathbf{d}_i^{y\Sigma}\right)_{\hat{\mathbf{z}}} = (\mathbf{d}_i^y)_{\hat{\mathbf{z}}} + \mathbf{B}_i^y\left(\mathbf{d}_h^{y\Sigma}\right)_{\hat{\mathbf{z}}} \tag{4.191}$$

In the RTdyn1 version, (4.190) is reduced to:

$$\left(\mathbf{d}_i^{z\Sigma}\right)_{\hat{\mathbf{z}}} = (\mathbf{d}_i^z)_{\hat{\mathbf{z}}} + \left(\mathbf{d}_h^{z\Sigma}\right)_{\hat{\mathbf{z}}} \tag{4.192}$$

### 4.4.15 Evaluation of $(\mathbf{d}_i^v)_{\hat{\boldsymbol{\rho}}}$

The recursive kinematic relation $\mathbf{d}_i^v$ can be affected by geometrical considerations such as local coordinates of points and vectors defining any of the preceding joints in the kinematic chain. In this regard, its partial derivative with respect to a set of parameters can exist, and the simplest method to compute it, is through a recursive accumulation. Taking derivatives on (2.121e) with respect to any parameter, $(\mathbf{d}_i^v)_{\hat{\boldsymbol{\rho}}}$ takes the form:

$$(\mathbf{d}_i^v)_{\hat{\boldsymbol{\rho}}} = \left(\dot{\mathbf{B}}_i^v\right)_{\hat{\boldsymbol{\rho}}}\mathbf{V}_{i-1} + \dot{\mathbf{B}}_i^v(\mathbf{V}_{i-1})_{\hat{\boldsymbol{\rho}}} + \left(\dot{\mathbf{b}}_i^v\right)_{\hat{\boldsymbol{\rho}}}\dot{\mathbf{z}}_i \tag{4.193}$$

in which $\left(\dot{\mathbf{B}}_i^v\right)_{\hat{\boldsymbol{\rho}}}$ has been studied in section 4.4.12 and $\left(\dot{\mathbf{b}}_i^v\right)_{\hat{\boldsymbol{\rho}}}$ in section 4.4.7.

## 4.5 Point and vector derivatives

Although the main variables of the topological models studied in this work are joint coordinates, Cartesian coordinates of points and vectors are ubiquitous in the expressions of kinematic relations, in the dynamic accumulations and even in the definition of constraints. The evaluation of natural coordinates positions, velocities and accelerations does not constitute a significant computational effort in general problems, but their derivatives are much more time demanding. Due to the omnipresence of natural coordinate derivatives in any joint-coordinate sensitivity analysis, an efficient assessment is studied and developed in this section.

Regarding the differentiation with respect to relative coordinates, the expressions included in chapter 2 are valid and correct, and also offer accurate results. However, they are based on a partial computation of the matrix $\mathbf{R}^v$, involving products of $3 \times 6$ by $6 \times n$ matrices, and although the sparsity of the matrix $\mathbf{R}^v$ is considered, it implies a computational effort that can be reduced.

Using as starting point the same expressions of the derivatives of $\mathbf{q}$ introduced in section 3.6, a new set of equations involving lower order products is achieved. In addition, the expressions obtained are independent of the set of reference points selected, allowing the use of the same expressions for RTdyn0 and RTdyn1 accumulations. Behold that expressions in terms of $\mathbf{R}^v$ and the new expressions introduced are identical, being the difference the computation method employed to obtain them.

On the other hand, the method for the evaluation of the derivatives with respect to a set of parameters is totally different. The types of parameters that affect the position, velocity or acceleration of a point or a vector are exclusively those related to the local coordinates of other or the same point or vector in the local reference frame of a body. Accordingly, a group of equations where the position, velocity and acceleration of a point or a vector explicitly involves the local coordinates of points and vectors of the mechanism have to be used to compute the derivatives. For this reason, the expressions of chapter 2 involving rotation matrices have to be used during these derivations.

In the following sections, the derivatives of the positions, velocities and accelerations of points and vectors with respect to the relative coordinates are firstly described, and then the derivatives with respect to any local coordinate are tackled.

### 4.5.1 Elemental evaluation of $\mathbf{q_z}$

Looking at expressions (3.93), (3.94), (3.95) and (3.96), the effect of the reference point is evident in the derivatives of any point. Theoretically, natural coordinates are not affected by the reference points selected since they only depend on the relative coordinates. Therefore, it is possible to reach an expression of the derivatives of any point or vector that do not involve the reference points. Expanding the derivatives of natural coordinates for any reference point (3.91) and (3.92), it can be observed that they can be computed only with the entities that define the joint (points, vectors or Euler parameters).

Prior to the description of the general expression of the new method, it is convenient to introduce its fundamentals through a particular example. Consider the calculation of the partial derivative of the coordinates of a point located in the body number 6 of the six body mechanism of Figure 2.10, with the assumption that all the joints are revolute joints. Applying (3.91):

$$\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}} = \frac{\partial \dot{\mathbf{r}}_k}{\partial \dot{\mathbf{z}}} = \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_6 - \tilde{\mathbf{r}}_k\end{bmatrix} \mathbf{R}_i^v = \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_6 - \tilde{\mathbf{r}}_k\end{bmatrix} \begin{bmatrix}(\mathbf{B}_6^v \mathbf{B}_4^v \mathbf{b}_1^v) & \mathbf{0} & \mathbf{0} & (\mathbf{B}_6^v \mathbf{b}_4^v) & \mathbf{0} & \mathbf{b}_6^v\end{bmatrix} \tag{4.194}$$

where

$$\begin{aligned}\begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_6 - \tilde{\mathbf{r}}_k\end{bmatrix} (\mathbf{B}_6^v \mathbf{B}_4^v \mathbf{b}_1^v) &= \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_6 - \tilde{\mathbf{r}}_k\end{bmatrix} \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_4 - \tilde{\mathbf{r}}_6 \\ \mathbf{0} & \mathbf{I}\end{bmatrix} \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_1 - \tilde{\mathbf{r}}_4 \\ \mathbf{0} & \mathbf{I}\end{bmatrix} \begin{bmatrix}\mathbf{w}_{j1} \wedge (\mathbf{r}_1 - \mathbf{r}_{j1}) \\ \mathbf{w}_{j1}\end{bmatrix} = \\ &= \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_1 - \tilde{\mathbf{r}}_k\end{bmatrix} \begin{bmatrix}\mathbf{w}_{j1} \wedge (\mathbf{r}_1 - \mathbf{r}_{j1}) \\ \mathbf{w}_{j1}\end{bmatrix} = \left(\tilde{\mathbf{r}}_{j1} - \tilde{\mathbf{r}}_k\right) \mathbf{w}_{j1}\end{aligned} \tag{4.195}$$

$$\begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_6 - \tilde{\mathbf{r}}_k\end{bmatrix} (\mathbf{B}_6^v \mathbf{b}_2^v) = \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_6 - \tilde{\mathbf{r}}_k\end{bmatrix} \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_4 - \tilde{\mathbf{r}}_6 \\ \mathbf{0} & \mathbf{I}\end{bmatrix} \begin{bmatrix}\mathbf{w}_{j4} \wedge (\mathbf{r}_4 - \mathbf{r}_{j4}) \\ \mathbf{w}_{j4}\end{bmatrix} = \left(\tilde{\mathbf{r}}_{j4} - \tilde{\mathbf{r}}_k\right) \mathbf{w}_{j4} \tag{4.196}$$

$$\begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_6 - \tilde{\mathbf{r}}_k\end{bmatrix} \mathbf{b}_6^v = \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_6 - \tilde{\mathbf{r}}_k\end{bmatrix} \begin{bmatrix}\mathbf{w}_{j6} \wedge (\mathbf{r}_6 - \mathbf{r}_{j6}) \\ \mathbf{w}_{j6}\end{bmatrix} = \left(\tilde{\mathbf{r}}_{j6} - \tilde{\mathbf{r}}_k\right) \mathbf{w}_{j6} \tag{4.197}$$

Recalling equation 4.194, it can be seen that the final expression has no dependence on the reference point selected, but only on the points and vectors defining the joint.

$$\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}} = \begin{bmatrix}\mathbf{I} & \tilde{\mathbf{r}}_6 - \tilde{\mathbf{r}}_k\end{bmatrix} \mathbf{R}_i^v = \begin{bmatrix}\left(\tilde{\mathbf{r}}_{j1} - \tilde{\mathbf{r}}_k\right) \mathbf{w}_{j1} & \mathbf{0} & \mathbf{0} & \left(\tilde{\mathbf{r}}_{j4} - \tilde{\mathbf{r}}_k\right) \mathbf{w}_{j4} & \mathbf{0} & \left(\tilde{\mathbf{r}}_{j6} - \tilde{\mathbf{r}}_k\right) \mathbf{w}_{j6}\end{bmatrix} \tag{4.198}$$

Continuing with the calculation of $\mathbf{q}_\mathbf{z}$ independently of the reference points selected in a general fashion, the elemental terms $\mathbf{q}_{\mathbf{z}_i}$ are introduced. Let us consider:

$$\mathbf{q}_\mathbf{z} = \begin{bmatrix}\mathbf{q}_{\mathbf{z}_1} & \mathbf{q}_{\mathbf{z}_2} & \cdots & \mathbf{q}_{\mathbf{z}_{n_j}}\end{bmatrix} \tag{4.199}$$

where $n_j$ is the number of joints of the mechanism, and the terms $\mathbf{q}_{\mathbf{z}_i}$ represent the partial derivative of the position of a point or a vector with respect to the relative coordinates of joint $i$.

In the following expressions, the terms with the subscript $j$ are part of the definition of the joint, while the subscript $k$ represents the point or vector being differentiated.

- **Revolute joint**

  The partial derivative of the position of a point with respect to the relative coordinates of a revolute joint is:

$$\frac{\partial \mathbf{r}_k}{\partial z_i} = \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right) \mathbf{w}_j \tag{4.200}$$

  while the partial derivative of a vector is:

$$\frac{\partial \mathbf{u}_k}{\partial z_i} = \left(-\tilde{\mathbf{u}}_k\right) \mathbf{w}_j \tag{4.201}$$

- **Prismatic joint**
  For a point:
$$\frac{\partial \mathbf{r}_k}{\partial z_i} = \mathbf{w}_j \tag{4.202}$$
  For a vector:
$$\frac{\partial \mathbf{u}_k}{\partial z_i} = \mathbf{0} \tag{4.203}$$

- **Cardan joint**
  For a point:
$$\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}_i} = \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\begin{bmatrix}\mathbf{w}_{j1} & \mathbf{w}_{j2}\end{bmatrix} \tag{4.204}$$
  For a vector:
$$\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}_i} = \left(-\tilde{\mathbf{u}}_k\right)\begin{bmatrix}\mathbf{w}_{j1} & \mathbf{w}_{j2}\end{bmatrix} \tag{4.205}$$

- **Cylindrical joint**
  For a point:
$$\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}_i} = \begin{bmatrix}\mathbf{w}_j & \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\mathbf{w}_j\end{bmatrix} \tag{4.206}$$
  For a vector:
$$\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}_i} = \begin{bmatrix}\mathbf{0} & \left(-\tilde{\mathbf{u}}_k\right)\mathbf{w}_j\end{bmatrix} \tag{4.207}$$

- **Spherical joint**
  For a point:
$$\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}_i} = 2\left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\mathbf{E} \tag{4.208}$$
  For a vector:
$$\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}_i} = 2\left(-\tilde{\mathbf{u}}_k\right)\mathbf{E} \tag{4.209}$$

- **Floating joint**
  For a point:
$$\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}_i} = \begin{bmatrix}\mathbf{I}_3 & 2\left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\mathbf{E}\end{bmatrix} \tag{4.210}$$
  For a vector:
$$\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}_i} = \begin{bmatrix}\mathbf{0}_3 & 2\left(-\tilde{\mathbf{u}}_k\right)\mathbf{E}\end{bmatrix} \tag{4.211}$$

- **Planar joint**
  For a point:
$$\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}_i} = \begin{bmatrix}\mathbf{u}_j & \mathbf{v}_j & \left(\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_k\right)\mathbf{w}_k\end{bmatrix} \tag{4.212}$$
  For a vector:
$$\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}_i} = \begin{bmatrix}\mathbf{0} & \mathbf{0} & \left(-\tilde{\mathbf{u}}_k\right)\mathbf{w}_k\end{bmatrix} \tag{4.213}$$

Observe the simplicity of the resulting terms and that the matrix products involved in any of the previous expressions are $3 \times 3$ by $3 \times j$, with $j = 1,..,4$, while the equivalent calculations presented in section 3.6 include products of $3 \times 6$ by $6 \times j$, with $j = 1,..,7$. It is important to remark also that no different expressions are needed for any of the semi-recursive accumulations presented in this work, due to the independence of the final expressions with respect to the set of reference points selected.

Despite being a significant improvement into the computation of $\mathbf{q_z}$, the relevance of these simplifications relies on the second derivatives $\mathbf{q_{zz}}$ and $\dot{\mathbf{q}}_{\mathbf{zz}}$, presented in the subsequent sections.

### 4.5.2 Elemental evaluation of $\dot{\mathbf{q}}_{\mathbf{z}}$

The partial derivative of the velocity of any point or vector can be easily assembled with the scheme presented in the previous section. In this case, the elemental derivatives can be obtained as the time derivatives of the joint expressions of the previous section, as long as $\frac{\mathrm{d}\mathbf{q_z}}{\mathrm{d}t} = \dot{\mathbf{q}}_{\mathbf{z}}$ (see (3.102)).

$$\dot{\mathbf{q}}_{\mathbf{z}} = \begin{bmatrix} \dot{\mathbf{q}}_{\mathbf{z}_1} & \dot{\mathbf{q}}_{\mathbf{z}_2} & \cdots & \dot{\mathbf{q}}_{\mathbf{z}_{n_j}} \end{bmatrix} \tag{4.214}$$

being $n_j$ the number of joints.

- **Revolute joint**
  The partial derivative of the velocity of a point with respect to the relative coordinates of a revolute joint takes the form:

$$\frac{\partial \dot{\mathbf{r}}_k}{\partial z_i} = \left(\dot{\tilde{\mathbf{r}}}_j - \dot{\tilde{\mathbf{r}}}_k\right) \mathbf{w}_j + \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right) \dot{\mathbf{w}}_j \tag{4.215}$$

  while the partial derivative of a vector becomes:

$$\frac{\partial \dot{\mathbf{u}}_k}{\partial z_i} = \left(-\dot{\tilde{\mathbf{u}}}_k\right) \mathbf{w}_j + \left(-\tilde{\mathbf{u}}_k\right) \dot{\mathbf{w}}_j \tag{4.216}$$

- **Prismatic joint**
  For a point:

$$\frac{\partial \dot{\mathbf{r}}_k}{\partial z_i} = \dot{\mathbf{w}}_j \tag{4.217}$$

  For a vector:

$$\frac{\partial \dot{\mathbf{u}}_k}{\partial z_i} = \mathbf{0} \tag{4.218}$$

- **Cardan joint**
  For a point:

$$\frac{\partial \dot{\mathbf{r}}_k}{\partial \mathbf{z}_i} = \left(\dot{\tilde{\mathbf{r}}}_j - \dot{\tilde{\mathbf{r}}}_k\right) \begin{bmatrix} \mathbf{w}_{j1} & \mathbf{w}_{j2} \end{bmatrix} + \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right) \begin{bmatrix} \dot{\mathbf{w}}_{j1} & \dot{\mathbf{w}}_{j2} \end{bmatrix} \tag{4.219}$$

For a vector:

$$\frac{\partial \dot{\mathbf{u}}_k}{\partial \mathbf{z}_i} = \left(-\dot{\tilde{\mathbf{u}}}_k\right) \begin{bmatrix}\mathbf{w}_{j1} & \mathbf{w}_{j2}\end{bmatrix} + \left(-\tilde{\mathbf{u}}_k\right) \begin{bmatrix}\dot{\mathbf{w}}_{j1} & \dot{\mathbf{w}}_{j2}\end{bmatrix} \tag{4.220}$$

- **Cylindrical joint**
  For a point:

$$\frac{\partial \dot{\mathbf{r}}_k}{\partial \mathbf{z}_i} = \begin{bmatrix}\dot{\mathbf{w}}_j & \left(\dot{\tilde{\mathbf{r}}}_j - \dot{\tilde{\mathbf{r}}}_k\right)\mathbf{w}_j\end{bmatrix} + \begin{bmatrix}\mathbf{0} & \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\dot{\mathbf{w}}_j\end{bmatrix} \tag{4.221}$$

  For a vector:

$$\frac{\partial \dot{\mathbf{u}}_k}{\partial \mathbf{z}_i} = \begin{bmatrix}\mathbf{0} & \left(-\dot{\tilde{\mathbf{u}}}_k\right)\mathbf{w}_j\end{bmatrix} + \begin{bmatrix}\mathbf{0} & \left(-\tilde{\mathbf{u}}_k\right)\dot{\mathbf{w}}_j\end{bmatrix} \tag{4.222}$$

- **Spherical joint**
  For a point:

$$\frac{\partial \dot{\mathbf{r}}_k}{\partial \mathbf{z}_i} = 2\left(\dot{\tilde{\mathbf{r}}}_j - \dot{\tilde{\mathbf{r}}}_k\right)\mathbf{E} + 2\left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\dot{\mathbf{E}} \tag{4.223}$$

  For a vector:

$$\frac{\partial \dot{\mathbf{u}}_k}{\partial \mathbf{z}_i} = 2\left(-\dot{\tilde{\mathbf{u}}}_k\right)\mathbf{E} + 2\left(-\tilde{\mathbf{u}}_k\right)\dot{\mathbf{E}} \tag{4.224}$$

- **Floating joint**
  For a point:

$$\frac{\partial \dot{\mathbf{r}}_k}{\partial \mathbf{z}_i} = \begin{bmatrix}\mathbf{0}_3 & 2\left(\dot{\tilde{\mathbf{r}}}_j - \dot{\tilde{\mathbf{r}}}_k\right)\mathbf{E}\end{bmatrix} + \begin{bmatrix}\mathbf{0}_3 & 2\left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\dot{\mathbf{E}}\end{bmatrix} \tag{4.225}$$

  For a vector:

$$\frac{\partial \dot{\mathbf{u}}_k}{\partial \mathbf{z}_i} = \begin{bmatrix}\mathbf{0}_3 & 2\left(-\dot{\tilde{\mathbf{u}}}_k\right)\mathbf{E}\end{bmatrix} + \begin{bmatrix}\mathbf{0}_3 & 2\left(-\tilde{\mathbf{u}}_k\right)\dot{\mathbf{E}}\end{bmatrix} \tag{4.226}$$

- **Planar joint**
  For a point:

$$\frac{\partial \dot{\mathbf{r}}_k}{\partial \mathbf{z}_i} = \begin{bmatrix}\dot{\mathbf{u}}_j & \dot{\mathbf{v}}_j & \left(\dot{\tilde{\mathbf{r}}}_G^i - \dot{\tilde{\mathbf{r}}}_k\right)\mathbf{w}_k + \left(\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_k\right)\dot{\mathbf{w}}_k\end{bmatrix} \tag{4.227}$$

  For a vector:

$$\frac{\partial \dot{\mathbf{u}}_k}{\partial \mathbf{z}_i} = \begin{bmatrix}\mathbf{0} & \mathbf{0} & \left(-\dot{\tilde{\mathbf{u}}}_k\right)\mathbf{w}_k + \left(-\tilde{\mathbf{u}}_k\right)\dot{\mathbf{w}}_k\end{bmatrix} \tag{4.228}$$

The derivatives $\frac{\partial \dot{\mathbf{r}}_k}{\partial \mathbf{z}_i}$ or $\frac{\partial \dot{\mathbf{u}}_k}{\partial \mathbf{z}_i}$ can be then computed by assembling the corresponding elemental derivatives in a matrix of $3 \times n$, being $n$ the number of relative coordinates. The elemental terms are exclusively dependent on the position and velocity of the point whose velocity is being differentiated, and the position and velocity of the points, vectors and Euler parameters defining each one of the joints.

### 4.5.3 Elemental evaluation of $\ddot{\mathbf{q}}_{\mathbf{z}}$

It can be proved that the partial derivative of the accelerations of any point or vector can be also calculated differentiating once again the expressions of the previous section with respect to time. The term $\ddot{\mathbf{q}}_{\mathbf{z}}$ is not always necessary in the dynamics or the sensitivity analysis, but it is required if the objective function of the sensitivity analysis is expressed in terms of accelerations of natural coordinates.

- **Revolute joint**
  The partial derivative of the acceleration of a point with respect to the relative coordinates of a revolute joint is:

$$\frac{\partial \ddot{\mathbf{r}}_k}{\partial z_i} = \left(\ddot{\tilde{\mathbf{r}}}_j - \ddot{\tilde{\mathbf{r}}}_k\right)\mathbf{w}_j + 2\left(\dot{\tilde{\mathbf{r}}}_j - \dot{\tilde{\mathbf{r}}}_k\right)\dot{\mathbf{w}}_j + \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\ddot{\mathbf{w}}_j \qquad (4.229)$$

  while the partial derivative of a vector is:

$$\frac{\partial \ddot{\mathbf{u}}_k}{\partial z_i} = \left(-\ddot{\tilde{\mathbf{u}}}_k\right)\mathbf{w}_j + 2\left(-\dot{\tilde{\mathbf{u}}}_k\right)\dot{\mathbf{w}}_j + \left(-\tilde{\mathbf{u}}_k\right)\ddot{\mathbf{w}}_j \qquad (4.230)$$

- **Prismatic joint**
  For a point:

$$\frac{\partial \ddot{\mathbf{r}}_k}{\partial z_i} = \ddot{\mathbf{w}}_j \qquad (4.231)$$

  For a vector:

$$\frac{\partial \ddot{\mathbf{u}}_k}{\partial z_i} = \mathbf{0} \qquad (4.232)$$

- **Cardan joint**
  For a point:

$$\frac{\partial \ddot{\mathbf{r}}_k}{\partial \mathbf{z}_i} = \left(\ddot{\tilde{\mathbf{r}}}_j - \ddot{\tilde{\mathbf{r}}}_k\right)\begin{bmatrix}\mathbf{w}_{j1} & \mathbf{w}_{j2}\end{bmatrix} + 2\left(\dot{\tilde{\mathbf{r}}}_j - \dot{\tilde{\mathbf{r}}}_k\right)\begin{bmatrix}\dot{\mathbf{w}}_{j1} & \dot{\mathbf{w}}_{j2}\end{bmatrix} + \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\begin{bmatrix}\ddot{\mathbf{w}}_{j1} & \ddot{\mathbf{w}}_{j2}\end{bmatrix} \qquad (4.233)$$

  For a vector:

$$\frac{\partial \ddot{\mathbf{u}}_k}{\partial \mathbf{z}_i} = \left(-\ddot{\tilde{\mathbf{u}}}_k\right)\begin{bmatrix}\mathbf{w}_{j1} & \mathbf{w}_{j2}\end{bmatrix} + 2\left(-\dot{\tilde{\mathbf{u}}}_k\right)\begin{bmatrix}\dot{\mathbf{w}}_{j1} & \dot{\mathbf{w}}_{j2}\end{bmatrix} + \left(-\tilde{\mathbf{u}}_k\right)\begin{bmatrix}\ddot{\mathbf{w}}_{j1} & \ddot{\mathbf{w}}_{j2}\end{bmatrix} \qquad (4.234)$$

- **Cylindrical joint**
  For a point:

$$\frac{\partial \ddot{\mathbf{r}}_k}{\partial \mathbf{z}_i} = \begin{bmatrix}\ddot{\mathbf{w}}_j & \left(\ddot{\tilde{\mathbf{r}}}_j - \ddot{\tilde{\mathbf{r}}}_k\right)\mathbf{w}_j\end{bmatrix} + 2\begin{bmatrix}\mathbf{0} & \left(\dot{\tilde{\mathbf{r}}}_j - \dot{\tilde{\mathbf{r}}}_k\right)\dot{\mathbf{w}}_j\end{bmatrix} + \begin{bmatrix}\mathbf{0} & \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\ddot{\mathbf{w}}_j\end{bmatrix} \qquad (4.235)$$

  For a vector:

$$\frac{\partial \ddot{\mathbf{u}}_k}{\partial \mathbf{z}_i} = \begin{bmatrix}\mathbf{0} & \left(-\ddot{\tilde{\mathbf{u}}}_k\right)\mathbf{w}_j\end{bmatrix} + 2\begin{bmatrix}\mathbf{0} & \left(-\dot{\tilde{\mathbf{u}}}_k\right)\dot{\mathbf{w}}_j\end{bmatrix} + \begin{bmatrix}\mathbf{0} & \left(-\tilde{\mathbf{u}}_k\right)\ddot{\mathbf{w}}_j\end{bmatrix} \qquad (4.236)$$

- **Spherical joint**
  For a point:

$$\frac{\partial \ddot{\mathbf{r}}_k}{\partial \mathbf{z}_i} = 2\left(\ddot{\tilde{\mathbf{r}}}_j - \ddot{\tilde{\mathbf{r}}}_k\right)\mathbf{E} + 4\left(\dot{\tilde{\mathbf{r}}}_j - \dot{\tilde{\mathbf{r}}}_k\right)\dot{\mathbf{E}} + 2\left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\ddot{\mathbf{E}} \qquad (4.237)$$

  For a vector:

$$\frac{\partial \ddot{\mathbf{u}}_k}{\partial \mathbf{z}_i} = 2\left(-\ddot{\tilde{\mathbf{u}}}_k\right)\mathbf{E} + 4\left(-\dot{\tilde{\mathbf{u}}}_k\right)\dot{\mathbf{E}} + 2\left(-\tilde{\mathbf{u}}_k\right)\ddot{\mathbf{E}} \qquad (4.238)$$

  Observe that the second time derivative of the term **E** has not been defined before, and it can be calculated with the rotation matrix, angular velocity, angular acceleration and the Euler parameters and its first and second time derivatives.

$$\ddot{\mathbf{E}} = \left(\dot{\tilde{\boldsymbol{\omega}}}_{i-1} + \tilde{\boldsymbol{\omega}}_{i-1}\tilde{\boldsymbol{\omega}}_{i-1}\right)\mathbf{E} + 2\tilde{\boldsymbol{\omega}}_{i-1}\mathbf{A}_{i-1}\dot{\bar{\mathbf{E}}} + \mathbf{A}_{i-1}\ddot{\bar{\mathbf{E}}} \qquad (4.239)$$

- **Floating joint**
  For a point:

$$\frac{\partial \ddot{\mathbf{r}}_k}{\partial \mathbf{z}_i} = \left[\mathbf{0}_3 \quad 2\left(\ddot{\tilde{\mathbf{r}}}_j - \ddot{\tilde{\mathbf{r}}}_k\right)\mathbf{E}\right] + \left[\mathbf{0}_3 \quad 4\left(\dot{\tilde{\mathbf{r}}}_j - \dot{\tilde{\mathbf{r}}}_k\right)\dot{\mathbf{E}}\right] + \left[\mathbf{0}_3 \quad 2\left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\ddot{\mathbf{E}}\right] \qquad (4.240)$$

  For a vector:

$$\frac{\partial \ddot{\mathbf{u}}_k}{\partial \mathbf{z}_i} = \left[\mathbf{0}_3 \quad 2\left(-\ddot{\tilde{\mathbf{u}}}_k\right)\mathbf{E}\right] + \left[\mathbf{0}_3 \quad 4\left(-\dot{\tilde{\mathbf{u}}}_k\right)\dot{\mathbf{E}}\right] + \left[\mathbf{0}_3 \quad 2\left(-\tilde{\mathbf{u}}_k\right)\ddot{\mathbf{E}}\right] \qquad (4.241)$$

- **Planar joint**
  For a point:

$$\frac{\partial \ddot{\mathbf{r}}_k}{\partial \mathbf{z}_i} = \left[\ddot{\mathbf{u}}_j \quad \ddot{\mathbf{v}}_j \quad \left(\ddot{\tilde{\mathbf{r}}}^i_G - \ddot{\tilde{\mathbf{r}}}_k\right)\mathbf{w}_k + 2\left(\dot{\tilde{\mathbf{r}}}^i_G - \dot{\tilde{\mathbf{r}}}_k\right)\dot{\mathbf{w}}_k + \left(\tilde{\mathbf{r}}^i_G - \tilde{\mathbf{r}}_k\right)\ddot{\mathbf{w}}_k\right] \qquad (4.242)$$

  For a vector:

$$\frac{\partial \ddot{\mathbf{u}}_k}{\partial \mathbf{z}_i} = \left[\mathbf{0} \quad \mathbf{0} \quad \left(-\ddot{\tilde{\mathbf{u}}}_k\right)\mathbf{w}_k + 2\left(-\dot{\tilde{\mathbf{u}}}_k\right)\dot{\mathbf{w}}_k + \left(-\tilde{\mathbf{u}}_k\right)\ddot{\mathbf{w}}_k\right] \qquad (4.243)$$

The computation of the present term is slightly more time demanding than the previous ones because it requires the evaluation of the accelerations of the points, vectors and the Euler parameters that define the joints of the mechanism. These accelerations are not needed in general since all the assemblies and accumulations are carried out by means of positions and velocities of these entities. However, if needed, the accelerations can be computed on demand using the equations of the rigid body.

### 4.5.4 Elemental evaluation of $\mathbf{q}_{\mathbf{zz}}$

Using the expressions presented in the previous section, the derivation of the tensor $\mathbf{q}_{\mathbf{zz}}$ is straightforward. Let us consider the matrix $\mathbf{q}_{\mathbf{zz}}\mathbf{x}$, defined as the product of the tensor $\mathbf{q}_{\mathbf{zz}} \in \mathbb{R}^{n_q \times n \times n}$ by a vector $\mathbf{x} \in \mathbb{R}^n$.

$$\mathbf{q}_{\mathbf{zz}}\mathbf{x} = \frac{\partial}{\partial \mathbf{z}} \begin{bmatrix} \frac{\partial \mathbf{q}_1}{\partial \mathbf{z}} \\ \cdots \\ \frac{\partial \mathbf{q}_i}{\partial \mathbf{z}} \\ \cdots \\ \frac{\partial \mathbf{q}_{n_q}}{\partial \mathbf{z}} \end{bmatrix} \mathbf{x} = \frac{\partial}{\partial \mathbf{z}} \begin{bmatrix} \frac{\partial \mathbf{q}_1}{\partial \mathbf{z}_1} & \cdots & \frac{\partial \mathbf{q}_1}{\partial \mathbf{z}_j} & \cdots & \frac{\partial \mathbf{q}_1}{\partial \mathbf{z}_{n_j}} \\ \cdots \\ \frac{\partial \mathbf{q}_i}{\partial \mathbf{z}_1} & \cdots & \frac{\partial \mathbf{q}_i}{\partial \mathbf{z}_j} & \cdots & \frac{\partial \mathbf{q}_i}{\partial \mathbf{z}_{n_j}} \\ \cdots \\ \frac{\partial \mathbf{q}_{n_q}}{\partial \mathbf{z}_1} & \cdots & \frac{\partial \mathbf{q}_{n_q}}{\partial \mathbf{z}_j} & \cdots & \frac{\partial \mathbf{q}_{n_q}}{\partial \mathbf{z}_{n_j}} \end{bmatrix} \mathbf{x}$$
$$= \begin{bmatrix} \frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{q}_1}{\partial \mathbf{z}_1}\right)\mathbf{x}_1 & \cdots & \frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{q}_1}{\partial \mathbf{z}_j}\right)\mathbf{x}_j & \cdots & \frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{q}_1}{\partial \mathbf{z}_{nj}}\right)\mathbf{x}_{n_j} \\ \cdots \\ \frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{q}_i}{\partial \mathbf{z}_1}\right)\mathbf{x}_1 & \cdots & \frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{q}_i}{\partial \mathbf{z}_j}\right)\mathbf{x}_j & \cdots & \frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{q}_i}{\partial \mathbf{z}_{n_j}}\right)\mathbf{x}_{n_j} \\ \cdots \\ \frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{q}_{n_q}}{\partial \mathbf{z}_1}\right)\mathbf{x}_1 & \cdots & \frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{q}_{n_q}}{\partial \mathbf{z}_j}\right)\mathbf{x}_j & \cdots & \frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{q}_{n_q}}{\partial \mathbf{z}_{n_j}}\right)\mathbf{x}_{n_j} \end{bmatrix} \tag{4.244}$$

again, $n_j$ is the number of kinematic joints.

Observe that the terms $\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{q}_i}{\partial \mathbf{z}_j}\right)\mathbf{x}_j$ can be easily calculated with the expressions of the previous section.

- **Revolute joint**
  For a point:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\frac{\partial \mathbf{w}_j}{\partial \mathbf{z}}\mathbf{x}_j + \tilde{\mathbf{w}}_j\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}} - \frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}\right)\mathbf{x}_j \tag{4.245}$$

  For a vector:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = -\tilde{\mathbf{u}}_k\frac{\partial \mathbf{w}_j}{\partial \mathbf{z}}\mathbf{x}_j + \tilde{\mathbf{w}}_j\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}}\mathbf{x}_j \tag{4.246}$$

- **Prismatic joint**
  For a point:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = \frac{\partial \mathbf{w}_j}{\partial \mathbf{z}}\mathbf{x}_j \tag{4.247}$$

  For a vector:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = \mathbf{0} \tag{4.248}$$

- **Cardan joint**
  For a point:

$$\begin{aligned}\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j &= \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\begin{bmatrix}\frac{\partial \mathbf{w}_{j1}}{\partial \mathbf{z}} & \frac{\partial \mathbf{w}_{j2}}{\partial \mathbf{z}}\end{bmatrix}\mathbf{x}_j \\ &+ \begin{bmatrix}\tilde{\mathbf{w}}_{j1}\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}} - \frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}\right) & \tilde{\mathbf{w}}_{j2}\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}} - \frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}\right)\end{bmatrix}\mathbf{x}_j\end{aligned} \tag{4.249}$$

  For a vector:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = \left(-\tilde{\mathbf{u}}_k\right)\begin{bmatrix}\frac{\partial \mathbf{w}_{j1}}{\partial \mathbf{z}} & \frac{\partial \mathbf{w}_{j2}}{\partial \mathbf{z}}\end{bmatrix}\mathbf{x}_j + \begin{bmatrix}\tilde{\mathbf{w}}_{j1}\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}} & \tilde{\mathbf{w}}_{j2}\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}}\end{bmatrix}\mathbf{x}_j \tag{4.250}$$

- **Cylindrical joint**
  For a point:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = \begin{bmatrix}\frac{\partial \mathbf{w}_j}{\partial \mathbf{z}} & \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\frac{\partial \mathbf{w}_j}{\partial \mathbf{z}}\end{bmatrix}\mathbf{x}_j + \begin{bmatrix}\mathbf{0} & \tilde{\mathbf{w}}_j\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}} - \frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}\right)\end{bmatrix}\mathbf{x}_j \tag{4.251}$$

  For a vector:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = \left(-\tilde{\mathbf{u}}_k\right)\begin{bmatrix}\mathbf{0} & \frac{\partial \mathbf{w}_j}{\partial \mathbf{z}}\end{bmatrix}\mathbf{x}_j + \begin{bmatrix}\mathbf{0} & \tilde{\mathbf{w}}_j\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}}\end{bmatrix}\mathbf{x}_j \tag{4.252}$$

- **Spherical joint**
  For a point:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = 2\left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\frac{\partial \mathbf{E}}{\partial \mathbf{z}}\mathbf{x}_j + 2\tilde{\mathbf{E}}\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}} - \frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}\right)\mathbf{x}_j \tag{4.253}$$

  For a vector:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = -2\tilde{\mathbf{u}}_k\frac{\partial \mathbf{E}}{\partial \mathbf{z}}\mathbf{x}_j + 2\tilde{\mathbf{E}}\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}}\mathbf{x}_j \tag{4.254}$$

- **Floating joint**
  For a point:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = \begin{bmatrix}\mathbf{0} & 2\left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\frac{\partial \mathbf{E}}{\partial \mathbf{z}}\mathbf{x}_j + \tilde{\mathbf{E}}\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}} - \frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}\right)\mathbf{x}_j\end{bmatrix} \tag{4.255}$$

  For a vector:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = \begin{bmatrix}\mathbf{0} & -2\tilde{\mathbf{u}}_k\frac{\partial \mathbf{E}}{\partial \mathbf{z}}\mathbf{x}_j + \tilde{\mathbf{E}}\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}}\mathbf{x}_j\end{bmatrix} \tag{4.256}$$

  For the sake of simplicity in the definition of the derivative for the floating joint case, $\mathbf{x}_j$ is assumed to be the part of the vector $\mathbf{x}$ corresponding to the position of the Euler parameters of the floating joint, thus neglecting the three first coordinates of the floating joint corresponding to the translations, which are zero.

- **Planar joint**
  For a point:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}_i}\right)\mathbf{x}_j = \begin{bmatrix}\frac{\partial \mathbf{u}_j}{\partial \mathbf{z}_i}\mathbf{x}_{j1} & \frac{\partial \mathbf{v}_j}{\partial \mathbf{z}_i}\mathbf{x}_{j2} & (\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_k)\frac{\partial \mathbf{w}_k}{\partial \mathbf{z}_i}\mathbf{x}_{j3} + \tilde{\mathbf{w}}_k\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}_i} - \frac{\partial \mathbf{r}_G^i}{\partial \mathbf{z}_i}\right)\mathbf{x}_{j3}\end{bmatrix} \tag{4.257}$$

  For a vector:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}_i}\right)\mathbf{x}_j = \begin{bmatrix}\mathbf{0} & \mathbf{0} & \tilde{\mathbf{w}}_k\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}}\mathbf{x}_{j3} - \tilde{\mathbf{u}}_k\frac{\partial \mathbf{w}_k}{\partial \mathbf{z}}\mathbf{x}_{j3}\end{bmatrix} \tag{4.258}$$

  where

$$\mathbf{x}_j = \begin{bmatrix}\mathbf{x}_{j1} & \mathbf{x}_{j2} & \mathbf{x}_{j3}\end{bmatrix}^{\mathrm{T}} \tag{4.259}$$

Observe that the resulting elemental second derivatives are formulated in terms of single derivatives of points, vectors and Euler parameters, whose expressions have been presented in previous sections. The scheme of differentiation presented enhances the efficiency of the computation of $\mathbf{q_{zz}}$ compared to the expressions depending on $\mathbf{R}^v$.

### 4.5.5 Elemental evaluation of $\mathbf{q}_{\mathbf{zz}}^{\mathrm{T}}$

This term uses the same expressions obtained in section 4.5.4, but under a different assembly.

$$\mathbf{q}_{\mathbf{zz}}^{\mathrm{T}}\mathbf{x} = \frac{\partial}{\partial \mathbf{z}}\begin{bmatrix}\frac{\partial \mathbf{q}_1}{\partial \mathbf{z}_1}^{\mathrm{T}} & \dots & \frac{\partial \mathbf{q}_i}{\partial \mathbf{z}_1}^{\mathrm{T}} & \dots & \frac{\partial \mathbf{q}_{n_q}}{\partial \mathbf{z}_1}^{\mathrm{T}} \\ \dots & & & & \\ \frac{\partial \mathbf{q}_1}{\partial \mathbf{z}_j}^{\mathrm{T}} & \dots & \frac{\partial \mathbf{q}_i}{\partial \mathbf{z}_j}^{\mathrm{T}} & \dots & \frac{\partial \mathbf{q}_{n_q}}{\partial \mathbf{z}_j}^{\mathrm{T}} \\ \dots & & & & \\ \frac{\partial \mathbf{q}_1}{\partial \mathbf{z}_{n_j}}^{\mathrm{T}} & \dots & \frac{\partial \mathbf{q}_i}{\partial \mathbf{z}_{n_j}}^{\mathrm{T}} & \dots & \frac{\partial \mathbf{q}_{n_q}}{\partial \mathbf{z}_{n_j}}^{\mathrm{T}}\end{bmatrix}\mathbf{x} = \begin{bmatrix}\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{q}}{\partial \mathbf{z}_1}^{\mathrm{T}}\right)\mathbf{x} \\ \dots \\ \frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{q}}{\partial \mathbf{z}_j}^{\mathrm{T}}\right)\mathbf{x} \\ \dots \\ \frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{q}}{\partial \mathbf{z}_{n_j}}^{\mathrm{T}}\right)\mathbf{x}\end{bmatrix} \tag{4.260}$$

where

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{q}}{\partial \mathbf{z}_j}^{\mathrm{T}}\right)\mathbf{x} = \sum_{i=1}^{n}\left(\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{q}_i}{\partial \mathbf{z}_j}^{\mathrm{T}}\right)\mathbf{x}_i\right) \tag{4.261}$$

and the terms $\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{q}_i}{\partial \mathbf{z}_j}^{\mathrm{T}}\right)$ can be easily obtained by transposing the expressions obtained in section 4.5.4. The main difference with the evaluation of $\mathbf{q_{zz}}$ is the assembly of the resulting matrix and the partial products by $\mathbf{x}$.

### 4.5.6 Elemental evaluation of $\dot{\mathbf{q}}_{\mathbf{zz}}$

The partial derivative $\dot{\mathbf{q}}_{\mathbf{zz}}$ can be easily obtained by assembling the time derivatives of the terms introduced in section 4.5.4.

- **Revolute joint**
  For a point:

$$\begin{aligned}\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \dot{\mathbf{r}}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = \left(\dot{\tilde{\mathbf{r}}}_j - \dot{\tilde{\mathbf{r}}}_k\right)\frac{\partial \mathbf{w}_j}{\partial \mathbf{z}}\mathbf{x}_j + \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\frac{\partial \dot{\mathbf{w}}_j}{\partial \mathbf{z}}\mathbf{x}_j + \\ \dot{\tilde{\mathbf{w}}}_j\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}} - \frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}\right)\mathbf{x}_j + \tilde{\mathbf{w}}_j\left(\frac{\partial \dot{\mathbf{r}}_k}{\partial \mathbf{z}} - \frac{\partial \dot{\mathbf{r}}_j}{\partial \mathbf{z}}\right)\mathbf{x}_j\end{aligned} \tag{4.262}$$

  For a vector:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \dot{\mathbf{u}}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = -\dot{\tilde{\mathbf{u}}}_k\frac{\partial \mathbf{w}_j}{\partial \mathbf{z}}\mathbf{x}_j - \tilde{\mathbf{u}}_k\frac{\partial \dot{\mathbf{w}}_j}{\partial \mathbf{z}}\mathbf{x}_j + \dot{\tilde{\mathbf{w}}}_j\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}}\mathbf{x}_j + \tilde{\mathbf{w}}_j\frac{\partial \dot{\mathbf{u}}_k}{\partial \mathbf{z}}\mathbf{x}_j \tag{4.263}$$

- **Prismatic joint**
  For a point:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \dot{\mathbf{r}}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = \frac{\partial \dot{\mathbf{w}}_j}{\partial \mathbf{z}}\mathbf{x}_j \tag{4.264}$$

  For a vector:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \dot{\mathbf{u}}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = \mathbf{0} \tag{4.265}$$

- **Cardan joint**
  For a point:

$$\begin{aligned}\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \dot{\mathbf{r}}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = \left(\dot{\tilde{\mathbf{r}}}_j - \dot{\tilde{\mathbf{r}}}_k\right)\begin{bmatrix}\frac{\partial \mathbf{w}_{j1}}{\partial \mathbf{z}} & \frac{\partial \mathbf{w}_{j2}}{\partial \mathbf{z}}\end{bmatrix}\mathbf{x}_j + \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\begin{bmatrix}\frac{\partial \dot{\mathbf{w}}_{j1}}{\partial \mathbf{z}} & \frac{\partial \dot{\mathbf{w}}_{j2}}{\partial \mathbf{z}}\end{bmatrix}\mathbf{x}_j \\ + \begin{bmatrix}\dot{\tilde{\mathbf{w}}}_{j1}\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}} - \frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}\right) & \dot{\tilde{\mathbf{w}}}_{j2}\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}} - \frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}\right)\end{bmatrix}\mathbf{x}_j \\ + \begin{bmatrix}\tilde{\mathbf{w}}_{j1}\left(\frac{\partial \dot{\mathbf{r}}_k}{\partial \mathbf{z}} - \frac{\partial \dot{\mathbf{r}}_j}{\partial \mathbf{z}}\right) & \tilde{\mathbf{w}}_{j2}\left(\frac{\partial \dot{\mathbf{r}}_k}{\partial \mathbf{z}} - \frac{\partial \dot{\mathbf{r}}_j}{\partial \mathbf{z}}\right)\end{bmatrix}\mathbf{x}_j\end{aligned} \tag{4.266}$$

  For a vector:

$$\begin{aligned}\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \dot{\mathbf{u}}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = \left(-\dot{\tilde{\mathbf{u}}}_k\right)\begin{bmatrix}\frac{\partial \mathbf{w}_{j1}}{\partial \mathbf{z}} & \frac{\partial \mathbf{w}_{j2}}{\partial \mathbf{z}}\end{bmatrix}\mathbf{x}_j + \left(-\tilde{\mathbf{u}}_k\right)\begin{bmatrix}\frac{\partial \dot{\mathbf{w}}_{j1}}{\partial \mathbf{z}} & \frac{\partial \dot{\mathbf{w}}_{j2}}{\partial \mathbf{z}}\end{bmatrix}\mathbf{x}_j \\ + \begin{bmatrix}\dot{\tilde{\mathbf{w}}}_{j1}\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}} & \dot{\tilde{\mathbf{w}}}_{j2}\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}}\end{bmatrix}\mathbf{x}_j + \begin{bmatrix}\tilde{\mathbf{w}}_{j1}\frac{\partial \dot{\mathbf{u}}_k}{\partial \mathbf{z}} & \tilde{\mathbf{w}}_{j2}\frac{\partial \dot{\mathbf{u}}_k}{\partial \mathbf{z}}\end{bmatrix}\mathbf{x}_j\end{aligned} \tag{4.267}$$

- **Cylindrical joint**
  For a point:

$$\begin{aligned}\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \dot{\mathbf{r}}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = &\left[\frac{\partial \dot{\mathbf{w}}_j}{\partial \mathbf{z}} \quad \left(\dot{\tilde{\mathbf{r}}}_j - \dot{\tilde{\mathbf{r}}}_k\right)\frac{\partial \mathbf{w}_j}{\partial \mathbf{z}} + \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\frac{\partial \dot{\mathbf{w}}_j}{\partial \mathbf{z}}\right]\mathbf{x}_j \\ &+ \left[\mathbf{0} \quad \dot{\tilde{\mathbf{w}}}_j\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}} - \frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}\right) + \tilde{\mathbf{w}}_j\left(\frac{\partial \dot{\mathbf{r}}_k}{\partial \mathbf{z}} - \frac{\partial \dot{\mathbf{r}}_j}{\partial \mathbf{z}}\right)\right]\mathbf{x}_j\end{aligned} \tag{4.268}$$

  For a vector:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \dot{\mathbf{u}}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = \left[\mathbf{0} \quad -\dot{\tilde{\mathbf{u}}}_k\frac{\partial \mathbf{w}_j}{\partial \mathbf{z}} - \tilde{\mathbf{u}}_k\frac{\partial \dot{\mathbf{w}}_j}{\partial \mathbf{z}}\right]\mathbf{x}_j + \left[\mathbf{0} \quad \dot{\tilde{\mathbf{w}}}_j\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}} + \tilde{\mathbf{w}}_j\frac{\partial \dot{\mathbf{u}}_k}{\partial \mathbf{z}}\right]\mathbf{x}_j \tag{4.269}$$

- **Spherical joint**
  For a point:

$$\begin{aligned}\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \dot{\mathbf{r}}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = &\,2\left(\dot{\tilde{\mathbf{r}}}_j - \dot{\tilde{\mathbf{r}}}_k\right)\frac{\partial \mathbf{E}}{\partial \mathbf{z}}\mathbf{x}_j + 2\left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\frac{\partial \dot{\mathbf{E}}}{\partial \mathbf{z}}\mathbf{x}_j + \\ &2\dot{\tilde{\mathbf{E}}}\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}} - \frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}\right)\mathbf{x}_j + 2\tilde{\mathbf{E}}\left(\frac{\partial \dot{\mathbf{r}}_k}{\partial \mathbf{z}} - \frac{\partial \dot{\mathbf{r}}_j}{\partial \mathbf{z}}\right)\mathbf{x}_j\end{aligned} \tag{4.270}$$

  For a vector:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \dot{\mathbf{u}}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = -2\dot{\tilde{\mathbf{u}}}_k\frac{\partial \mathbf{E}}{\partial \mathbf{z}}\mathbf{x}_j - 2\tilde{\mathbf{u}}_k\frac{\partial \dot{\mathbf{E}}}{\partial \mathbf{z}}\mathbf{x}_j + 2\dot{\tilde{\mathbf{E}}}\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}}\mathbf{x}_j + 2\tilde{\mathbf{E}}\frac{\partial \dot{\mathbf{u}}_k}{\partial \mathbf{z}}\mathbf{x}_j \tag{4.271}$$

- **Floating joint**
  For a point:

$$\begin{aligned}\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \dot{\mathbf{r}}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = &\left[\mathbf{0} \quad 2\left(\dot{\tilde{\mathbf{r}}}_j - \dot{\tilde{\mathbf{r}}}_k\right)\frac{\partial \mathbf{E}}{\partial \mathbf{z}}\mathbf{x}_j + 2\left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_k\right)\frac{\partial \dot{\mathbf{E}}}{\partial \mathbf{z}}\mathbf{x}_j\right] + \\ &\left[\mathbf{0} \quad +2\tilde{\mathbf{E}}\left(\frac{\partial \dot{\mathbf{r}}_k}{\partial \mathbf{z}} - \frac{\partial \dot{\mathbf{r}}_j}{\partial \mathbf{z}}\right)\mathbf{x}_j + 2\dot{\tilde{\mathbf{E}}}\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}} - \frac{\partial \mathbf{r}_j}{\partial \mathbf{z}}\right)\mathbf{x}_j\right]\end{aligned} \tag{4.272}$$

  For a vector:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \dot{\mathbf{u}}_k}{\partial \mathbf{z}_j}\right)\mathbf{x}_j = \left[\mathbf{0} \quad -2\dot{\tilde{\mathbf{u}}}_k\frac{\partial \mathbf{E}}{\partial \mathbf{z}}\mathbf{x}_j - 2\tilde{\mathbf{u}}_k\frac{\partial \dot{\mathbf{E}}}{\partial \mathbf{z}}\mathbf{x}_j + \dot{\tilde{\mathbf{E}}}\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}}\mathbf{x}_j + \tilde{\mathbf{E}}\frac{\partial \dot{\mathbf{u}}_k}{\partial \mathbf{z}}\mathbf{x}_j\right] \tag{4.273}$$

  In this case, $\mathbf{x}_j$ is assumed to be the part of the vector $\mathbf{x}$ corresponding to the position of the Euler parameters of the floating joint. The derivative with respect to the three first coordinates of the floating joint corresponding to the translations are zero.

- **Planar joint**
  For a point:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \dot{\mathbf{r}}_k}{\partial \mathbf{z}_i}\right)\mathbf{x}_j = \begin{bmatrix}\frac{\partial \dot{\mathbf{u}}_j}{\partial \mathbf{z}_i}\mathbf{x}_{j1} & \frac{\partial \dot{\mathbf{v}}_j}{\partial \mathbf{z}_i}\mathbf{x}_{j2} & \left(\dot{\tilde{\mathbf{r}}}_G^i - \dot{\tilde{\mathbf{r}}}_k\right)\frac{\partial \mathbf{w}_k}{\partial \mathbf{z}_i}\mathbf{x}_{j3} + \dot{\tilde{\mathbf{w}}}_k\left(\frac{\partial \mathbf{r}_k}{\partial \mathbf{z}_i} - \frac{\partial \mathbf{r}_G^i}{\partial \mathbf{z}_i}\right)\mathbf{x}_{j3}\end{bmatrix} + \begin{bmatrix}\mathbf{0} & \mathbf{0} & \left(\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_k\right)\frac{\partial \dot{\mathbf{w}}_k}{\partial \mathbf{z}_i}\mathbf{x}_{j3} + \tilde{\mathbf{w}}_k\left(\frac{\partial \dot{\mathbf{r}}_k}{\partial \mathbf{z}_i} - \frac{\partial \dot{\mathbf{r}}_G^i}{\partial \mathbf{z}_i}\right)\mathbf{x}_{j3}\end{bmatrix} \tag{4.274}$$

  For a vector:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \dot{\mathbf{u}}_k}{\partial \mathbf{z}_i}\right)\mathbf{x}_j = \begin{bmatrix}\mathbf{0} & \mathbf{0} & \dot{\tilde{\mathbf{w}}}_k\frac{\partial \mathbf{u}_k}{\partial \mathbf{z}}\mathbf{x}_{j3} + \tilde{\mathbf{w}}_k\frac{\partial \dot{\mathbf{u}}_k}{\partial \mathbf{z}}\mathbf{x}_{j3} - \dot{\tilde{\mathbf{u}}}_k\frac{\partial \mathbf{w}_k}{\partial \mathbf{z}}\mathbf{x}_{j3} - \tilde{\mathbf{u}}_k\frac{\partial \dot{\mathbf{w}}_k}{\partial \mathbf{z}}\mathbf{x}_{j3}\end{bmatrix} \tag{4.275}$$

  where

$$\mathbf{x} = \begin{bmatrix}\mathbf{x}_j & \mathbf{x}_{j2} & \mathbf{x}_{j3}\end{bmatrix} \tag{4.276}$$

The terms described in this section are the most expensive from a computational point of view due to the number of products and different derivatives involved. In the sensitivity analysis in which this term is required, its efficient computation would determine the global performance of the whole analysis.

### 4.5.7 Evaluation of $\mathbf{q}_{\boldsymbol{\rho}}$

One of the premises of relative coordinate models is that any position of any point or vector within a kinematic chain can be computed in terms of a set of relative coordinates. However, those positions are also conditioned by a set of constant terms which play an essential role in a sensitivity analysis.

The kinematic recursive relations presented in chapter 2 depend exclusively on relative coordinates, on the local coordinates of the points and vectors of the model and on the topology of the mechanism, i.e. on the sequence and type of joints defining the relative motion between bodies. The local coordinates of the model are, thus, the set of parameters that directly affect the global position coordinates of a point or a vector. If lengths or angles defined within the local reference frame of a body are intended to be used as parameters, then a transformation of the resulting sensitivities with respect to local coordinates into the derivatives of these new parameters have to be accomplished, because any local length or angle can be described by a combination of local coordinates of different points or vectors.

Using the expressions of positions formulated in terms of rotation matrices, it can be concluded that the global position of any point of the kinematic chain can be obtained by means of the addition of vectors defining the position of the points involved in each one of the preceding joints in the kinematic chain plus a term related to the local position of the point within the body in which the point is defined. This can be formulated for the types of joints where the bodies share one point, such as revolute, spherical and Cardan joints as:

$$\mathbf{r}_k = \mathbf{r}_i + \mathbf{A}_i\left(\bar{\mathbf{r}}_k^i - \bar{\mathbf{r}}_i^i\right) \tag{4.277}$$

where $\mathbf{A}_i$ is the rotation matrix of body $i$.

Applying recursive relations, the global position $\mathbf{r}_i$ can be expressed in terms of the points and vectors of the previous joints. Using the same types of joints:

$$\mathbf{r}_k^i = \sum_{h=1}^{i} \left( \mathbf{A}_h \left( \bar{\mathbf{r}}_h^h - \bar{\mathbf{r}}_{h-1}^h \right) \right) + \mathbf{A}_i \left( \bar{\mathbf{r}}_k^i - \bar{\mathbf{r}}_i^i \right) \tag{4.278}$$

being $h$ each one of the preceding joints of the body $i$ within the kinematic chain.

From (4.278) it can be deduced that the local coordinates affecting $\mathbf{r}_k^i$ are the ones used in the definition of the preceding joints in the kinematic chain, this is, the group of joints that are between the base body and the body $i$, as well as the local coordinates of point $k$ in body $i$. Considering that the rotation matrices depend only on local coordinates of vectors (see (2.20), (2.21), (2.22), (2.31), (2.40), (2.41), (2.43), (2.87), (2.88)), their derivatives with respect to the local coordinates of a point are null. Behold that none of the rotation matrix expressions introduced in section 2.1 involve any local coordinate of a point, neither in the definition of the constant ($\mathbf{A}_1$) or variable rotation matrices.

At this point, it is convenient to identify the two bodies related by a kinematic joint. Let us denote as first body in a kinematic joint the body that is closer to the base body, while the second body will be the one that is closer to the tips. Depending on in which of the two bodies a local coordinate is defined as a parameter, the derivative will be different.

According to the independence of rotation matrices from local coordinates of points, the partial derivative of the position of any point of the kinematic chain can be obtained with the following expression:

$$\frac{\partial \mathbf{r}_k}{\partial \bar{\mathbf{r}}_j^l} = \begin{cases} \mathbf{A}_l & \text{in case 1} \\ -\mathbf{A}_l & \text{in case 2} \\ \mathbf{0} & \text{in case 3} \end{cases} \tag{4.279}$$

being $l$ the body where the local coordinates of point $k$ are defined as parameters. The cases defined are:

- **Case 1**: point $j$ is contained in the definition of a preceding joint and $l$ is the first body of the joint, or $k = j$.

- **Case 2**: point $j$ is contained in the definition of a preceding joint and $l$ is the second body of the joint.

- **Case 3**: point $j$ does not affect any joint previous to body $i$ and $k \neq j$.

The definition is valid for any of the joint types commented. Let us see what happens when a prismatic joint is considered. Recalling (2.28):

$$\mathbf{r}_{j+2} = \mathbf{r}_{j+1} + z_{k+1}\mathbf{u}_{j+1} \tag{4.280}$$

Considering now prismatic, cylindrical and planar joints (they have the same behavior with respect to the translation coordinate), (4.278) can be reformulated as:

$$\mathbf{r}_k^i = \sum_{h=1}^{i} \left(\mathbf{A}_h \left(\bar{\mathbf{r}}_h^h - \bar{\mathbf{r}}_{h-1}^h\right)\right) + \mathbf{A}_i \left(\bar{\mathbf{r}}_k^i - \bar{\mathbf{r}}_i^i\right) + \sum_{h=1}^{i} \left(\mathbf{u}_h z_h\right) \tag{4.281}$$

being $\mathbf{u}_h$ the translation joint axis, and $z_h$ the translation coordinate. Observe that neither the axis nor the coordinate have direct relations with the local coordinate of any point, and therefore, equation (4.279) is still valid for any type of joint.

Following the same scheme, the derivative of the position of a vector with respect to the local coordinates of a point is always zero, because no local coordinates of points are included into an expression of a rotation matrix:

$$\frac{\partial \mathbf{u}_k}{\partial \bar{\mathbf{r}}_j^l} = \mathbf{0} \tag{4.282}$$

This derivative will be different than zero if two points are used as the axis of a rotation, but this possibility is not considered neither theoretically nor within the implementation of MBSLIM.

Note that, up to this point, only partial derivatives with respect to local coordinates of points have been explored. Now, the derivative of a vector with respect to the local coordinates of other or the same vector is accomplished. The position coordinates of a vector are much easier to obtain than the ones of points, because they depend exclusively on the rotation matrix of the body the vector belongs to and on the local coordinates of the vector:

$$\mathbf{u}_k = \mathbf{A}_i \bar{\mathbf{u}}_k^i \tag{4.283}$$

Despite the apparent simplicity, the derivative becomes more complex than the previous one with respect to local coordinates of points. The local coordinates of vectors have a direct impact on the evaluation of the rotation matrices, so in this differentiation this computation have to be taken into account.

The derivative of the position of a vector with respect to the local coordinates of other or the same vector are:

$$\frac{\partial \mathbf{u}_k}{\partial \bar{\mathbf{u}}_j^l} = \begin{cases} \mathbf{A}_l & \text{if } j = k \\[2ex] \dfrac{\partial \mathbf{A}_l}{\partial \bar{\mathbf{u}}_j^l} \bar{\mathbf{u}}_k^i & \text{if } \bar{\mathbf{u}}_j^l \text{ defines the relative rotation of one of the previous joints} \\[2ex] \mathbf{0} & \text{if } j \neq k \text{ and } \bar{\mathbf{u}}_j^l \text{ does not affect any relative rotation matrix} \end{cases} \tag{4.284}$$

The last possible case involves the derivative of a point with respect to the local coordinates of a vector. The expression of this derivative takes the form:

$$\frac{\partial \mathbf{r}_k}{\partial \bar{\mathbf{u}}_j^l} = \sum_{h=1}^{i} \left(\frac{\partial \mathbf{A}_h}{\partial \bar{\mathbf{u}}_j^l} \left(\bar{\mathbf{r}}_h^h - \bar{\mathbf{r}}_{h-1}^h\right)\right) + \frac{\partial \mathbf{A}_i}{\partial \bar{\mathbf{u}}_j^l} \left(\bar{\mathbf{r}}_k^i - \bar{\mathbf{r}}_i^i\right) + \sum_{h=1}^{i} \left(\frac{\partial \mathbf{u}_h}{\partial \bar{\mathbf{u}}_j^l} z_h\right) \tag{4.285}$$

In the following sections, only the local coordinates of points are considered as parameters in order to simplify the notation and allow a more direct understanding of the effects of the geometrical parameters into relative coordinate models. Local coordinates of vectors as parameters can also be considered with this approach by means of the previous equations and their time derivatives, but the computations become particularly complex during the calculation of $\mathbf{q}_{\mathbf{z}\boldsymbol{\rho}}$ and $\dot{\mathbf{q}}_{\mathbf{z}\boldsymbol{\rho}}$.

### 4.5.8 Evaluation of $\dot{\mathbf{q}}_{\boldsymbol{\rho}}$

Velocities of points and vectors can be expressed in terms of the local coordinates of the points and vectors of preceding joints, as well as other terms. Using those expressions, the partial derivative with respect to any local coordinate of a point can be directly obtained. However, it can be proved that the same result can be reached from the time differentiation of the expressions of section 4.5.7:

$$\frac{\partial \dot{\mathbf{r}}_k}{\partial \bar{\mathbf{r}}_j^l} = \begin{cases} \dot{\mathbf{A}}_l & \text{in case 1} \\ -\dot{\mathbf{A}}_l & \text{in case 2} \\ \mathbf{0} & \text{in case 3} \end{cases} \tag{4.286}$$

being $l$ the body where the local coordinates of point $k$ are defined as parameters, and the cases have been defined in section 4.5.7.

The derivative of the velocity of a vector is, therefore:

$$\frac{\partial \dot{\mathbf{u}}_k}{\partial \bar{\mathbf{r}}_j^l} = \mathbf{0} \tag{4.287}$$

### 4.5.9 Evaluation of $\ddot{\mathbf{q}}_{\boldsymbol{\rho}}$

Similarly to section 4.5.8, it can be proved that the derivatives of the accelerations of a point or a vector with respect to the local coordinates of a point can be obtained differentiating twice the expressions of $\mathbf{q}_{\boldsymbol{\rho}}$ presented in section 4.5.7 with respect to time:

$$\frac{\partial \ddot{\mathbf{r}}_k}{\partial \bar{\mathbf{r}}_j^l} = \begin{cases} \ddot{\mathbf{A}}_l & \text{in case 1} \\ -\ddot{\mathbf{A}}_l & \text{in case 2} \\ \mathbf{0} & \text{in case 3} \end{cases} \tag{4.288}$$

with $l$ the body in which the local coordinates of point $k$ are defined as parameters, and with cases defined in section 4.5.7.

The derivative of the velocity of a vector is in this case, once again:

$$\frac{\partial \ddot{\mathbf{u}}_k}{\partial \bar{\mathbf{r}}_j^l} = \mathbf{0} \tag{4.289}$$

### 4.5.10 Evaluation of $\mathbf{q}_{\mathbf{z}\boldsymbol{\rho}}$

Following the scheme of derivation of the previous sections, the second derivative of $\mathbf{q}$ with respect to the relative coordinates in positions and with respect to the parameters of the system, can be obtained by taking derivatives for the expressions obtained for $\mathbf{q}_{\boldsymbol{\rho}}$ with respect to $\mathbf{z}$. The results are:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{r}_k}{\partial \bar{\mathbf{r}}_j^l}\right) = \begin{cases} \dfrac{\partial \mathbf{A}_l}{\partial \mathbf{z}} & \text{in case 1} \\ -\dfrac{\partial \mathbf{A}_l}{\partial \mathbf{z}} & \text{in case 2} \\ \mathbf{0} & \text{in case 3} \end{cases} \tag{4.290}$$

wherein, once again, $l$ is the body where the local coordinates of point $k$ are defined as parameters. The cases have been defined in section 4.5.7.

The derivatives for a vector are, accordingly, all zero:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \mathbf{u}_k}{\partial \bar{\mathbf{r}}_j^l}\right) = \mathbf{0} \tag{4.291}$$

### 4.5.11 Evaluation of $\dot{\mathbf{q}}_{\mathbf{z}\boldsymbol{\rho}}$

It can be proved that $\dot{\mathbf{q}}_{\mathbf{z}\boldsymbol{\rho}}$ can be obtained as the time derivative of $\mathbf{q}_{\mathbf{z}\boldsymbol{\rho}}$. Applying again a new time derivative on the expressions of 4.5.10, $\dot{\mathbf{q}}_{\mathbf{z}\hat{\boldsymbol{\rho}}}$ is reached:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \dot{\mathbf{r}}_k}{\partial \bar{\mathbf{r}}_j^l}\right) = \begin{cases} \dfrac{\partial \dot{\mathbf{A}}_l}{\partial \mathbf{z}} & \text{in case 1} \\ -\dfrac{\partial \dot{\mathbf{A}}_l}{\partial \mathbf{z}} & \text{in case 2} \\ \mathbf{0} & \text{in case 3} \end{cases} \tag{4.292}$$

where $l$ is the body where the local coordinates of point $k$ are defined as parameters, and, once again, the cases were defined in section 4.5.7.

The derivatives for a vector are null in all the cases:

$$\frac{\partial}{\partial \mathbf{z}}\left(\frac{\partial \dot{\mathbf{u}}_k}{\partial \bar{\mathbf{r}}_j^l}\right) = \mathbf{0} \tag{4.293}$$

Regarding the resulting expressions of the derivatives of $\mathbf{q}$ with respect to $\boldsymbol{\rho}$, local coordinates of points as parameters can be simply added to a general sensitivity analysis code using simple and easy-to-compute expressions.

# Chapter 5

# Sensitivity analysis of closed-loop systems

In this chapter, constrained multibody systems are considered in order to complete the general sensitivity analysis of topological models. One of the advantages of topological models with respect to global ones is that they tend to generate a problem with a smaller number of coordinates. Moreover, in the case of open-loop systems in minimal coordinates (see chapter 2), no constraints are needed for the equations of motion. For closed-loop systems, or systems modeled in non-minimal coordinates (see chapter 3), constraint equations appear, but their number tend to be also smaller than in the case of global formulations, because most of the joints between bodies do not need to be modeled using constraint equations. Hence, constraints have to be combined with the recursive joint relations and with the semi-recursive unconstrained equations of motion.

In chapter 3, the constraints have been successfully combined with the recursive relations, yielding the kinematic position, velocity and acceleration problems in relative coordinates. In addition, constraints have been also applied to the equations of motion using two very different schemes: the semi-recursive Matrix R formulation, consisting on the use of a second projection of velocities which conduces to an ODE system in independent coordinates; and the semi-recursive ALI3-P formulations, based on an Augmented Lagrangian scheme in positions with projections in velocities and accelerations, and which produces an index-3 DAE system in dependent coordinates.

In the present chapter, the sensitivity analyses of the three kinematic problems and the two semi-recursive dynamic formulations are accomplished. The sensitivity analyses developed are based on the differentiation of the analytical kinematic and dynamic expressions, following two different approaches: the direct differentiation method and the adjoint variable method.

The direct differentiation of the kinematic or dynamic expressions of a multibody system leads to a set of equations where the sensitivities of the states are the main variables. It means that if the dynamic problem has $s$ unknowns, the sensitivity analysis of this dynamics with respect to $p$ parameters will lead to a problem with $s \times p$ unknown variables. The advantages of this method are the analogy of the sensitivity system to the dynamics one, making relatively easy a sensitivity implementation, and the fact that the matrix of the system is the same for every parameter.

The adjoint method is, otherwise, a more complex approach, especially in systems of equations involving numerical integration as the dynamic problem. The real power of this method is in the reduced number of variables that need to be solved compared to the direct differentiation method for large numbers of parameters, and despite the more complex calculations and storage, it could be very profitable in some cases. With this method, the number of resulting unknown variables will be independent of the number of parameters selected, but it will be determined by the kind and size of the dynamic problem and by the number of objective functions (or optimization constraints) whose gradient is sought.

This chapter is divided in two blocks: one related to the sensitivity analysis of the kinematics of topological models, and a second block related to the sensitivity analysis of the dynamics of closed-loop systems.

Firstly, the sensitivity analysis of the kinematics in positions is outlined, considering the possibility of degrees of freedom not belonging to the set of joint coordinates. Then, the sensitivity analysis of the kinematic velocity and acceleration problems are described, paying special attention to the derivatives of the constraints. In general, the sensitivities of kinematic problems are much easier to solve and involve significantly less derivatives than the dynamic problem, but they are included in this chapter because they are needed to initiate the simulation in the form of the sensitivities of the initial position and velocity problems, and, moreover, because the problems have an interest by themselves. In general multibody systems, natural coordinates models offer the simplest solution to the kinematic problems, whereas the same problem in relative coordinates seems to be more entangled due to the particular treatment of each type of joint and to the accumulations required. However, the kinematics of closed-loop systems in relative coordinates has general known expressions which can be simply solved for any mechanism by using kinematic recursive relations and the derivatives of the constraint equations, as it was presented in chapter 3. In the current chapter, the differentiation of kinematic problems in relative coordinates is addressed, obtaining a general approach for the calculation of the sensitivity analysis of any multibody model using both the direct differentiation and the adjoint method.

In a second block, the sensitivities of the dynamics of closed-loop systems are presented. First, the sensitivity analysis of the semi-recursive Matrix R formulation is introduced, and then the semi-recursive ALI3-P sensitivity formulations are addressed. The two schemes of solution of the constrained equations of motion are successfully differentiated, yielding the forward sensitivity and adjoint variable expressions for both problems.

In chapter 4, the sensitivity analysis of open-loop systems has been described paying special attention to the differences between the general formulation for any reference point and the particular formulations RTdyn0 and RTdyn1. In the present chapter, all the expressions will be referred to the general formulation for any reference point. The evaluation of the new constraint derivatives required in the sensitivity formulations developed will be explained in detail in section 5.5. The rest of the terms of forces and masses needed for the sensitivities of the constrained dynamic equations have been introduced in chapter 4, and therefore they will be just referenced in this chapter.

## 5.1 Kinematic sensitivity analysis

In this section, the sensitivities of an objective function during a kinematic simulation over time will be studied. Let us consider a vector of objective functions[1] $\boldsymbol{\psi} \in \mathbb{R}^o$, dependent on positions, velocities and accelerations of points, vectors and relative coordinates as well as a set of parameters $\boldsymbol{\rho}$, being $o$ the number of scalar objective functions:

$$\boldsymbol{\psi} = \int_{t_0}^{t_F} \mathbf{g}\left(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbf{z}, \dot{\mathbf{z}}, \ddot{\mathbf{z}}, \boldsymbol{\rho}\right) \mathrm{d}t \tag{5.1}$$

wherein $\mathbf{g} \in \mathbb{R}^o$ is a vector of functions with a known analytical expression.

### 5.1.1 Forward sensitivity

Differentiating (5.1) with respect to the set of sensitivity parameters, the gradient of the objective function can be determined according to (4.2) in terms of $\mathbf{q}'$, $\dot{\mathbf{q}}'$, $\ddot{\mathbf{q}}'$, $\mathbf{z}'$, $\dot{\mathbf{z}}'$, $\ddot{\mathbf{z}}'$. However, considering the implicit dependencies of natural coordinates in relative coordinates, this gradient can be reformulated as:

$$\boldsymbol{\psi}' = \int_{t_0}^{t_F} \left(\mathbf{g}_{\hat{\mathbf{z}}}\mathbf{z}' + \mathbf{g}_{\hat{\dot{\mathbf{z}}}}\dot{\mathbf{z}}' + \mathbf{g}_{\hat{\ddot{\mathbf{z}}}}\ddot{\mathbf{z}}' + \mathbf{g}_{\hat{\boldsymbol{\rho}}}\right) \mathrm{d}t \tag{5.2}$$

where $\mathbf{g}_{\hat{\mathbf{z}}}$, $\mathbf{g}_{\hat{\dot{\mathbf{z}}}}$, $\mathbf{g}_{\hat{\ddot{\mathbf{z}}}}$ and $\mathbf{g}_{\hat{\boldsymbol{\rho}}}$ are given by (4.8).

From (5.2) and (4.8), it can be deduced that the composition of the gradient of the objective function in a kinematic sensitivity analysis of a relative coordinate model is significantly more complex than in natural coordinates models since it involves more terms and products, though it could be a deceptive impression. Consider for example the case of a closed-loop system: due to the reduced number of coordinates and constraints of relative coordinate models, the kinematic problems are usually compact and significantly smaller than the ones generated by natural coordinates models, and accordingly, their computation is usually faster, especially when the mechanism is composed of a high number of bodies.

Regarding the kinematic problems of position, velocity and acceleration, it can be observed that they have a strong interrelation, since the position problem has to be solved before the computation of the velocities, and the velocities of the states are also needed for the assessment of the kinematic acceleration analysis. Analogously, the sensitivities have the same dependencies, so in order to obtain the sensitivity of the acceleration of a state, for instance, the sensitivities of the positions and velocities have to be previously evaluated.

[1]Not all of them need to be objective functions, some of them could be design or optimization constraints for which the gradient with respect to the parameters is also needed.

In the following developments, a non-constant $\mathbf{B}$ matrix will be considered in order to embrace all the generality. The case of a constant $\mathbf{B}$ matrix can be inferred from these expressions by eliminating all its derivatives.

For the sake of clarity, let us recall here the expressions of the kinematic problems (3.15), (3.19a) and (3.19b):

$$\begin{bmatrix} \mathbf{\Phi}_{\hat{\mathbf{z}}}^{\{n\}} \\ \mathbf{B}^{\{n\}} \end{bmatrix} \Delta\mathbf{z}^{\{n+1\}} = \begin{bmatrix} -\mathbf{\Phi}^{\{n\}} \\ \mathbf{d} - \left(\mathbf{z}^{i}\right)^{\{n\}} \end{bmatrix} \tag{5.3a}$$

$$\begin{bmatrix} \mathbf{\Phi}_{\hat{\mathbf{z}}} \\ \mathbf{B} \end{bmatrix} \dot{\mathbf{z}} = \begin{bmatrix} -\mathbf{\Phi}_t \\ \dot{\mathbf{d}} \end{bmatrix} \equiv \begin{bmatrix} \mathbf{b} \\ \dot{\mathbf{d}} \end{bmatrix} \tag{5.3b}$$

$$\begin{bmatrix} \mathbf{\Phi}_{\hat{\mathbf{z}}} \\ \mathbf{B} \end{bmatrix} \ddot{\mathbf{z}} = \begin{bmatrix} -\dot{\mathbf{\Phi}}_t - \dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}\dot{\mathbf{z}} \\ \ddot{\mathbf{d}} - \dot{\mathbf{B}}\dot{\mathbf{z}} \end{bmatrix} \equiv \begin{bmatrix} \mathbf{c} \\ \ddot{\mathbf{d}} - \dot{\mathbf{B}}\dot{\mathbf{z}} \end{bmatrix} \tag{5.3c}$$

Differentiating with respect to a set of parameters $\boldsymbol{\rho}$, the sensitivities of the position, velocity and acceleration kinematic problems become:

$$\begin{bmatrix} \mathbf{\Phi}_{\hat{\mathbf{z}}} \\ \mathbf{B} \end{bmatrix} \mathbf{z}' = \begin{bmatrix} -\mathbf{\Phi}_{\rho} \\ \mathbf{d}' \end{bmatrix} \tag{5.4a}$$

$$\begin{bmatrix} \mathbf{\Phi}_{\hat{\mathbf{z}}} \\ \mathbf{B} \end{bmatrix} \dot{\mathbf{z}}' = \begin{bmatrix} -\mathbf{b}^{\rho} \\ \dot{\mathbf{d}}' - \bar{\mathbf{b}}^{\rho} \end{bmatrix} \tag{5.4b}$$

$$\begin{bmatrix} \mathbf{\Phi}_{\hat{\mathbf{z}}} \\ \mathbf{B} \end{bmatrix} \ddot{\mathbf{z}}' = \begin{bmatrix} -\mathbf{c}^{\rho} \\ \ddot{\mathbf{d}}' - \bar{\mathbf{c}}^{\rho} \end{bmatrix} \tag{5.4c}$$

in which $\mathbf{d}', \dot{\mathbf{d}}', \ddot{\mathbf{d}}' \in \mathbb{R}^{d \times p}$ are the sensitivities of the degrees of freedom at position, velocity and acceleration level, and the grouped terms introduced are:

$$\mathbf{b}^{\rho} = \dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}\mathbf{z}' + \dot{\mathbf{\Phi}}_{\hat{\rho}} \tag{5.5a}$$

$$\bar{\mathbf{b}}^{\rho} = \left(\mathbf{B}_{\hat{\mathbf{z}}}\mathbf{z}' + \mathbf{B}_{\hat{\rho}}\right)\dot{\mathbf{z}} = \dot{\mathbf{B}}\mathbf{z}' + \mathbf{B}_{\hat{\rho}}\dot{\mathbf{z}} \tag{5.5b}$$

$$\mathbf{c}^{\rho} = 2\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}\dot{\mathbf{z}}' + \ddot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}\mathbf{z}' + \ddot{\mathbf{\Phi}}_{\hat{\rho}} \tag{5.5c}$$

$$\begin{aligned} \bar{\mathbf{c}}^{\rho} &= \left(\mathbf{B}_{\hat{\mathbf{z}}}\mathbf{z}' + \mathbf{B}_{\hat{\rho}}\right)\ddot{\mathbf{z}} + \left(\dot{\mathbf{B}}_{\hat{\dot{\mathbf{z}}}}\dot{\mathbf{z}}' + \dot{\mathbf{B}}_{\hat{\mathbf{z}}}\mathbf{z}' + \dot{\mathbf{B}}_{\hat{\rho}}\right)\dot{\mathbf{z}} + \dot{\mathbf{B}}\dot{\mathbf{z}}' \\ &= 2\dot{\mathbf{B}}\dot{\mathbf{z}}' + \ddot{\mathbf{B}}\mathbf{z}' + \mathbf{B}_{\hat{\rho}}\ddot{\mathbf{z}} + \dot{\mathbf{B}}_{\hat{\rho}}\dot{\mathbf{z}} \end{aligned} \tag{5.5d}$$

where the following simplifications have been applied:

$$\mathbf{B}_{\hat{\mathbf{z}}}\dot{\mathbf{z}} = \dot{\mathbf{B}} \tag{5.6a}$$

$$\mathbf{B}_{\hat{\mathbf{z}}}\ddot{\mathbf{z}} + \dot{\mathbf{B}}_{\hat{\mathbf{z}}}\dot{\mathbf{z}} = \dot{\mathbf{B}}_{\hat{\dot{\mathbf{z}}}}\ddot{\mathbf{z}} + \dot{\mathbf{B}}_{\hat{\mathbf{z}}}\dot{\mathbf{z}} = \ddot{\mathbf{B}} \tag{5.6b}$$

Although constraint relations are almost immediate, the simplified expressions (5.6) require further explanation. Let us expand the expression of the time derivative of $\mathbf{B}$ according to its original definition given by (3.17):

$$\dot{\mathbf{B}} = \frac{\mathrm{d}\mathbf{B}}{\mathrm{d}t} = \left(\mathbf{B}\right)_{\hat{\mathbf{z}}} \otimes_3 \dot{\mathbf{z}} = \left(\left(f(\mathbf{z})\right)_{\hat{\mathbf{z}}}\right)_{\hat{\mathbf{z}}} \otimes_3 \dot{\mathbf{z}} \tag{5.7}$$

Now, consider the following property of tensor-vector products and second derivatives of a function $\mathbf{F} \in \mathbb{R}^f$ dependent on 2 vectors of independent variables $\mathbf{x} \in \mathbb{R}^x$ and $\mathbf{y} \in \mathbb{R}^y$:

$$\Big((f(\mathbf{x},\mathbf{y}))_{\mathbf{x}}\Big)_{\mathbf{y}} \otimes_2 \mathbf{x} = \Big((f(\mathbf{x},\mathbf{y}))_{\mathbf{y}}\Big)_{\mathbf{x}} \otimes_3 \mathbf{x} \tag{5.8}$$

Applying (5.8) to (5.9), the following equality is inferred:

$$\dot{\mathbf{B}} = ((f(\mathbf{z}))_{\hat{\mathbf{z}}})_{\hat{\mathbf{z}}} \otimes_3 \dot{\mathbf{z}} = ((f(\mathbf{z}))_{\hat{\mathbf{z}}})_{\hat{\mathbf{z}}} \otimes_2 \dot{\mathbf{z}} = \mathbf{B}_{\hat{\mathbf{z}}} \otimes_2 \dot{\mathbf{z}} \tag{5.9}$$

thus equation (5.6a) has been proved.

The expansion of $\ddot{\mathbf{B}}$ in terms partial derivatives holds:

$$\ddot{\mathbf{B}} = \frac{\mathrm{d}\dot{\mathbf{B}}}{\mathrm{d}t} = \frac{\mathrm{d}\mathbf{B}_{\hat{\mathbf{z}}}}{\mathrm{d}t} \otimes_3 \dot{\mathbf{z}} + (\mathbf{B})_{\hat{\mathbf{z}}} \otimes_3 \ddot{\mathbf{z}} \tag{5.10}$$

Let us explore now the time derivative of $\mathbf{B}_{\hat{\mathbf{z}}}$ considering its dependencies:

$$\frac{\mathrm{d}\mathbf{B}_{\hat{\mathbf{z}}}}{\mathrm{d}t} = (\mathbf{B}_{\hat{\mathbf{z}}})_{\hat{\mathbf{z}}} \otimes_4 \dot{\mathbf{z}} = (\mathbf{B}_{\hat{\mathbf{z}}} \otimes_3 \dot{\mathbf{z}})_{\hat{\mathbf{z}}} = \Big(\dot{\mathbf{B}}_{\hat{\mathbf{z}}}\Big)_{\hat{\mathbf{z}}} \tag{5.11}$$

Now, combining (5.8), (5.10) and (5.11), the following expression is achieved:

$$\begin{aligned}\ddot{\mathbf{B}} = \Big(\dot{\mathbf{B}}_{\hat{\mathbf{z}}}\Big)_{\hat{\mathbf{z}}} \otimes_3 \dot{\mathbf{z}} + (\mathbf{B})_{\hat{\mathbf{z}}} \otimes_3 \ddot{\mathbf{z}} = (\mathbf{B}_{\hat{\mathbf{z}}} \otimes_3 \dot{\mathbf{z}})_{\hat{\mathbf{z}}} \otimes_3 \dot{\mathbf{z}} + (\mathbf{B})_{\hat{\mathbf{z}}} \otimes_3 \ddot{\mathbf{z}} = \\ (\mathbf{B}_{\hat{\mathbf{z}}} \otimes_3 \dot{\mathbf{z}})_{\hat{\mathbf{z}}} \otimes_2 \dot{\mathbf{z}} + (\mathbf{B})_{\hat{\mathbf{z}}} \otimes_2 \ddot{\mathbf{z}}\end{aligned} \tag{5.12}$$

Therefore, (5.6b) has been validated.

The constraint derivatives involved in (5.5) can be obtained using the differentiation rule declared in (4.9). The terms used to compose them will be detailed in section 5.5.

The sensitivity of the position problem is unaffected by the variability of the matrix $\mathbf{B}$, as it can be seen in (5.4a), but the velocity and acceleration problems are penalized by the definition of degrees of freedom out from the vector of joint-coordinates $\mathbf{z}$. However, the derivatives required in those kinematic analyses are equivalent to the ones involved in the evaluation of $\mathbf{b}^{\rho}$ and $\mathbf{c}^{\rho}$ in the velocity and acceleration problems respectively. If the imposition of the degrees of freedom is regarded as a group of constraints, the same differentiation structure of the constraints can be reused. For instance, if two new constraint vectors $\bar{\dot{\mathbf{\Phi}}} \in \mathbb{R}^d$ and $\bar{\ddot{\mathbf{\Phi}}} \in \mathbb{R}^d$ are considered, being $d$ the number of degrees of freedom, their right-hand-side terms in the sensitivity expressions (5.4b) and (5.4c) will be:

$$\bar{\dot{\mathbf{\Phi}}} = \mathbf{B}\dot{\mathbf{z}} - \dot{\mathbf{z}}^i \Rightarrow \bar{\mathbf{b}}^{\rho} - \dot{\mathbf{z}}^{i\prime} = \bar{\dot{\mathbf{\Phi}}}_{\hat{\mathbf{z}}}\mathbf{z}' + \bar{\dot{\mathbf{\Phi}}}_{\hat{\rho}} \tag{5.13a}$$

$$\bar{\ddot{\mathbf{\Phi}}} = \mathbf{B}\ddot{\mathbf{z}} + \dot{\mathbf{B}}\dot{\mathbf{z}} - \ddot{\mathbf{z}}^{i} \Rightarrow \bar{\mathbf{c}}^{\boldsymbol{\rho}} - \ddot{\mathbf{z}}^{i\prime} = 2\bar{\dot{\mathbf{\Phi}}}_{\hat{\mathbf{z}}}\dot{\mathbf{z}}' + \bar{\ddot{\mathbf{\Phi}}}_{\hat{\mathbf{z}}}\mathbf{z}' + \bar{\ddot{\mathbf{\Phi}}}_{\hat{\boldsymbol{\rho}}} \tag{5.13b}$$

In brief, the difference between a constant and non-constant $\mathbf{B}$ matrix in the sensitivities of the kinematic problems is similar to the addition of a number of constraints equal to the number of degrees of freedom not included in the relative coordinates vector.

### 5.1.2 Adjoint sensitivity

The adjoint variable method is particularly well-suited for problems with a high number of parameters and low number of objective functions[2], since the number of sensitivity variables depends on the number of objective functions rather than on the number of parameters. Nevertheless, it could convey several drawbacks, specially for differential equations in which the imposition of initial conditions forces to use a backward-in-time integration scheme. Regarding that any of the kinematic problems introduced in chapter 3 is formulated as a set of algebraic equations, no numerical integration or initialization problem will be required, as it will be exposed in the following lines.

The gradient of an integral objective function $\boldsymbol{\psi} \in \mathbb{R}^{o}$ given by (5.1) can be computed using the following Lagrangian function:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\rho}) = \boldsymbol{\psi} &- \int_{t_0}^{t_F} \boldsymbol{\mu}_{\mathbf{\Phi}}^{\mathrm{T}}\mathbf{\Phi}\mathrm{d}t - \int_{t_0}^{t_F} \boldsymbol{\mu}_{\dot{\mathbf{\Phi}}}^{\mathrm{T}}\dot{\mathbf{\Phi}}\mathrm{d}t - \int_{t_0}^{t_F} \boldsymbol{\mu}_{\ddot{\mathbf{\Phi}}}^{\mathrm{T}}\ddot{\mathbf{\Phi}}\mathrm{d}t - \int_{t_0}^{t_F} \boldsymbol{\mu}_{\mathbf{z}}^{\mathrm{T}}\left(\mathbf{z}^{i} - \mathbf{d}\right)\mathrm{d}t \\ &- \int_{t_0}^{t_F} \boldsymbol{\mu}_{\dot{\mathbf{z}}}^{\mathrm{T}}\left(\mathbf{B}\dot{\mathbf{z}} - \dot{\mathbf{d}}\right)\mathrm{d}t - \int_{t_0}^{t_F} \boldsymbol{\mu}_{\ddot{\mathbf{z}}}^{\mathrm{T}}\left(\mathbf{B}\ddot{\mathbf{z}} - \dot{\mathbf{B}}\dot{\mathbf{z}} - \ddot{\mathbf{d}}\right)\mathrm{d}t\end{aligned} \tag{5.14}$$

where $\boldsymbol{\mu}_{\mathbf{\Phi}}$, $\boldsymbol{\mu}_{\dot{\mathbf{\Phi}}}$ and $\boldsymbol{\mu}_{\ddot{\mathbf{\Phi}}} \in \mathbb{R}^{m\times o}$ are the adjoint variables associated to the kinematic constraints in positions, velocities and accelerations respectively, and $\boldsymbol{\mu}_{\mathbf{z}}$, $\boldsymbol{\mu}_{\dot{\mathbf{z}}}$ and $\boldsymbol{\mu}_{\ddot{\mathbf{z}}} \in \mathbb{R}^{d\times o}$ are the adjoint variables associated to the imposition of the values of the DoF of the system. Observe that as long as the kinematic problems in positions, velocities and accelerations are solved, the Lagrangian will have the same value of the objective function (thus the same gradient) regardless of the values of the adjoint variables.

Taking derivatives on (5.14) with respect to the set of sensitivity parameters, the gradient of the Lagrangian yields:

$$\begin{aligned}\mathcal{L}' = \boldsymbol{\psi}' &- \int_{t_0}^{t_F} \boldsymbol{\mu}_{\mathbf{\Phi}}^{\mathrm{T}}\left(\mathbf{\Phi}_{\hat{\mathbf{z}}}\mathbf{z}' + \mathbf{\Phi}_{\hat{\boldsymbol{\rho}}}\right)\mathrm{d}t - \int_{t_0}^{t_F} \boldsymbol{\mu}_{\dot{\mathbf{\Phi}}}^{\mathrm{T}}\left(\mathbf{\Phi}_{\hat{\mathbf{z}}}\dot{\mathbf{z}}' + \mathbf{b}^{\boldsymbol{\rho}}\right)\mathrm{d}t - \int_{t_0}^{t_F} \boldsymbol{\mu}_{\ddot{\mathbf{\Phi}}}^{\mathrm{T}}\left(\mathbf{\Phi}_{\hat{\mathbf{z}}}\ddot{\mathbf{z}}' + \mathbf{c}^{\boldsymbol{\rho}}\right)\mathrm{d}t \\ &- \int_{t_0}^{t_F} \boldsymbol{\mu}_{\mathbf{z}}^{\mathrm{T}}\left(\mathbf{B}\mathbf{z}' - \mathbf{d}'\right)\mathrm{d}t - \int_{t_0}^{t_F} \boldsymbol{\mu}_{\dot{\mathbf{z}}}^{\mathrm{T}}\left(\mathbf{B}\dot{\mathbf{z}}' + \bar{\mathbf{b}}^{\boldsymbol{\rho}} - \dot{\mathbf{d}}'\right)\mathrm{d}t - \int_{t_0}^{t_F} \boldsymbol{\mu}_{\ddot{\mathbf{z}}}^{\mathrm{T}}\left(\mathbf{B}\ddot{\mathbf{z}}' + \bar{\mathbf{c}}^{\boldsymbol{\rho}} - \ddot{\mathbf{d}}'\right)\mathrm{d}t\end{aligned} \tag{5.15}$$

[2]Or constraint functions whose gradient is sought.

wherein the terms involving the derivatives of the newly added adjoint variables appear multiplied by a term that is null, so they are directly eliminated from the final derivative expression.

Now, the gradient of $\boldsymbol{\psi}$ given by (5.2) can be substituted in (5.15), and the compact terms $\mathbf{b}^{\boldsymbol{\rho}}$, $\mathbf{c}^{\boldsymbol{\rho}}$, $\bar{\mathbf{b}}^{\boldsymbol{\rho}}$, and $\bar{\mathbf{c}}^{\boldsymbol{\rho}}$ can be expanded to include their dependencies on $\mathbf{z}'$, $\dot{\mathbf{z}}'$ and $\ddot{\mathbf{z}}'$ using expressions (5.5):

$$\begin{aligned}
\mathcal{L}' = \int_{t_0}^{t_F} \left(\mathbf{g}_{\hat{\mathbf{z}}}\mathbf{z}' + \mathbf{g}_{\hat{\dot{\mathbf{z}}}}\dot{\mathbf{z}}' + \mathbf{g}_{\hat{\ddot{\mathbf{z}}}}\ddot{\mathbf{z}}' + \mathbf{g}_{\hat{\boldsymbol{\rho}}}\right) \mathrm{d}t - \int_{t_0}^{t_F} \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}} \left(\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\mathbf{z}' + \boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}}\right) \mathrm{d}t \\
- \int_{t_0}^{t_F} \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left(\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\dot{\mathbf{z}}' + \dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\mathbf{z}' + \dot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}}\right) \mathrm{d}t - \int_{t_0}^{t_F} \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left(\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\ddot{\mathbf{z}}' + 2\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\dot{\mathbf{z}}' + \ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\mathbf{z}' + \ddot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}}\right) \mathrm{d}t \\
- \int_{t_0}^{t_F} \boldsymbol{\mu}_{\mathbf{z}}^{\mathrm{T}} \left(\mathbf{B}\mathbf{z}' - \mathbf{d}'\right) \mathrm{d}t - \int_{t_0}^{t_F} \boldsymbol{\mu}_{\dot{\mathbf{z}}}^{\mathrm{T}} \left(\mathbf{B}\dot{\mathbf{z}}' + \left(\mathbf{B}_{\hat{\mathbf{z}}}\mathbf{z}' + \mathbf{B}_{\hat{\boldsymbol{\rho}}}\right)\dot{\mathbf{z}} - \dot{\mathbf{d}}'\right) \mathrm{d}t \\
- \int_{t_0}^{t_F} \boldsymbol{\mu}_{\ddot{\mathbf{z}}}^{\mathrm{T}} \left(\mathbf{B}\ddot{\mathbf{z}}' + \left(\mathbf{B}_{\hat{\mathbf{z}}}\mathbf{z}' + \mathbf{B}_{\hat{\boldsymbol{\rho}}}\right)\ddot{\mathbf{z}} + \left(\dot{\mathbf{B}}_{\hat{\dot{\mathbf{z}}}}\dot{\mathbf{z}}' + \dot{\mathbf{B}}_{\hat{\mathbf{z}}}\mathbf{z}' + \dot{\mathbf{B}}_{\hat{\boldsymbol{\rho}}}\right)\dot{\mathbf{z}} + \dot{\mathbf{B}}\dot{\mathbf{z}}' - \ddot{\mathbf{d}}'\right) \mathrm{d}t
\end{aligned} \tag{5.16}$$

Grouping terms with respect to the sensitivities of the states, and applying the identities introduced in (5.6a) and (5.6b), the sensitivity of the adjoint Lagrangian becomes:

$$\begin{aligned}
\mathcal{L}' = \int_{t_0}^{t_F} \left(\mathbf{g}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}_{\mathbf{z}}^{\mathrm{T}}\mathbf{B} - \boldsymbol{\mu}_{\dot{\mathbf{z}}}^{\mathrm{T}}\dot{\mathbf{B}} - \boldsymbol{\mu}_{\ddot{\mathbf{z}}}^{\mathrm{T}}\ddot{\mathbf{B}}\right) \mathbf{z}'\mathrm{d}t \\
+ \int_{t_0}^{t_F} \left(\mathbf{g}_{\hat{\dot{\mathbf{z}}}} - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}} - 2\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}_{\dot{\mathbf{z}}}^{\mathrm{T}}\mathbf{B} - 2\boldsymbol{\mu}_{\ddot{\mathbf{z}}}^{\mathrm{T}}\dot{\mathbf{B}}\right) \dot{\mathbf{z}}'\mathrm{d}t \\
+ \int_{t_0}^{t_F} \left(\mathbf{g}_{\hat{\ddot{\mathbf{z}}}} - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}_{\ddot{\mathbf{z}}}^{\mathrm{T}}\mathbf{B}\right) \ddot{\mathbf{z}}'\mathrm{d}t \\
+ \int_{t_0}^{t_F} \Big(\mathbf{g}_{\hat{\boldsymbol{\rho}}} + \boldsymbol{\mu}_{\mathbf{z}}^{\mathrm{T}}\mathbf{d}' + \boldsymbol{\mu}_{\dot{\mathbf{z}}}^{\mathrm{T}}\left(\dot{\mathbf{d}}' - \mathbf{B}_{\hat{\boldsymbol{\rho}}}\dot{\mathbf{z}}\right) + \boldsymbol{\mu}_{\ddot{\mathbf{z}}}^{\mathrm{T}}\left(\ddot{\mathbf{d}}' - \mathbf{B}_{\hat{\boldsymbol{\rho}}}\ddot{\mathbf{z}} - \dot{\mathbf{B}}_{\hat{\boldsymbol{\rho}}}\dot{\mathbf{z}}\right) \\
- \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}} - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}}\dot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}} - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\ddot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}}\Big) \mathrm{d}t
\end{aligned} \tag{5.17}$$

in which the relation $\dot{\mathbf{B}}_{\hat{\dot{\mathbf{z}}}} = \mathbf{B}_{\hat{\mathbf{z}}}$ has been employed.

In (5.17), the gradient of the adjoint Lagrangian is identical to the gradient of the objective function regardless of the adjoint variable values. Therefore, these adjoint variables can be selected such as all the terms multiplying the sensitivities of the states are null, hence their calculation can be avoided. The following systems of equations are therefore obtained:

$$\begin{bmatrix}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} & \mathbf{B}^{\mathrm{T}}\end{bmatrix} \begin{bmatrix}\boldsymbol{\mu}_{\boldsymbol{\Phi}} \\ \boldsymbol{\mu}_{\mathbf{z}}\end{bmatrix} = \mathbf{g}_{\hat{\mathbf{z}}}^{\mathrm{T}} - \dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}} - \ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}} - \dot{\mathbf{B}}^{\mathrm{T}}\boldsymbol{\mu}_{\dot{\mathbf{z}}} - \ddot{\mathbf{B}}^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\mathbf{z}}} \tag{5.18a}$$

$$\begin{bmatrix}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} & \mathbf{B}^{\mathrm{T}}\end{bmatrix} \begin{bmatrix}\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}} \\ \boldsymbol{\mu}_{\dot{\mathbf{z}}}\end{bmatrix} = \mathbf{g}_{\hat{\dot{\mathbf{z}}}}^{\mathrm{T}} - 2\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}} - 2\dot{\mathbf{B}}^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\mathbf{z}}} \tag{5.18b}$$

$$\begin{bmatrix}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} & \mathbf{B}^{\mathrm{T}}\end{bmatrix} \begin{bmatrix}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}} \\ \boldsymbol{\mu}_{\ddot{\mathbf{z}}}\end{bmatrix} = \mathbf{g}_{\hat{\ddot{\mathbf{z}}}}^{\mathrm{T}} \tag{5.18c}$$

These systems of equations have to be solved starting form the acceleration equations ($\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}$ and $\boldsymbol{\mu}_{\ddot{\mathbf{z}}}$), continuing with velocities ($\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}$ and $\boldsymbol{\mu}_{\dot{\mathbf{z}}}$) and ending with positions ($\boldsymbol{\mu}_{\boldsymbol{\Phi}}$ and $\boldsymbol{\mu}_{\mathbf{z}}$). In order to obtain the $3(m+d) \times o$ adjoint variables, the system (5.18) of $3n \times o$ equations have to be solved, but since the number of variables is lower than or equal to the number of constraints plus the number of degrees of freedom ($n \leq m+d$), this adjoint system could have infinite solutions, being one of them the minimum norm solution.

Once solved the system of adjoint equations and computed the adjoint variables, the gradient of the objective function can be calculated with the remaining terms of expression (5.17):

$$\begin{aligned}\mathcal{L}' = \int_{t_0}^{t_F} \Big(&\mathbf{g}_{\hat{\rho}} + \boldsymbol{\mu}_{\mathbf{z}}^{\mathrm{T}}\mathbf{d}' + \boldsymbol{\mu}_{\dot{\mathbf{z}}}^{\mathrm{T}}\left(\dot{\mathbf{d}}' - \mathbf{B}_{\hat{\rho}}\dot{\mathbf{z}}\right) + \boldsymbol{\mu}_{\ddot{\mathbf{z}}}^{\mathrm{T}}\left(\ddot{\mathbf{d}}' - \mathbf{B}_{\hat{\rho}}\ddot{\mathbf{z}} - \dot{\mathbf{B}}_{\hat{\rho}}\dot{\mathbf{z}}\right) \\ &- \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\rho}} - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}}\dot{\boldsymbol{\Phi}}_{\hat{\rho}} - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\ddot{\boldsymbol{\Phi}}_{\hat{\rho}}\Big)\,\mathrm{d}t\end{aligned} \tag{5.19}$$

The application of the adjoint variable method to the set of algebraic equations of the kinematic position, velocity and acceleration problems delivers, as presented in (5.18), a new set of algebraic equations. Comparing the set of equations yielded by the adjoint variable method (5.18) and the equations of the direct differentiation method (5.4), some analogies can be observed, like that the leading matrices of the adjoint problem are the transpose of the system matrices of the direct problems, of that both expressions require exactly the same terms, even though they are involved in different operations.

## 5.2 Sensitivity analysis of semi-recursive Matrix R formulations

The enforcement of kinematic constraints in dynamic analysis can be addressed through different approaches, three of them listed in chapter 3. The complexity in the differentiation process is strongly tied to the construction of the dynamic equations and to the set of expressions produced. In this sense, a set of dynamic equations formulated by means of an ODE system is substantially simpler to differentiate than problems expressed by a DAE system.

The semi-recursive Matrix R formulation extended to independent variables not included in the vector of joint coordinates $\mathbf{z}$ (variable $\mathbf{B}$ matrix) is analytically differentiated in this section with respect to a set of parameters $\boldsymbol{\rho} \in \mathbb{R}^p$ applying a direct differentiation scheme and the adjoint variable method.

Let us consider an integral objective function dependent on a set of Cartesian coordinates $\mathbf{q}$, $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$, a set of joint-coordinates $\mathbf{z}$, $\dot{\mathbf{z}}$ and $\ddot{\mathbf{z}}$, a set of degrees of freedom $\mathbf{z}^i$, $\dot{\mathbf{z}}^i$ and $\ddot{\mathbf{z}}^i$ and a set of parameters $\boldsymbol{\rho}$:

$$\psi = \int_{t_0}^{t_F} \mathbf{g}\left(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbf{z}, \dot{\mathbf{z}}, \ddot{\mathbf{z}}, \mathbf{z}^i, \dot{\mathbf{z}}^i, \ddot{\mathbf{z}}^i, \boldsymbol{\rho}\right)\mathrm{d}t \tag{5.20}$$

Behold that each degree of freedom is part of the Cartesian coordinates vector or the joint coordinates vector. In this regard, explicit dependencies of the objective function $\mathbf{g}$ on the degrees of freedom are not necessary since they are included in the Cartesian and joint coordinate dependencies, but they are considered separately due to particular simplifications conquered for these dependencies.

### 5.2.1 Forward sensitivity

Taking derivatives on (5.20) with respect to a set of parameters $\boldsymbol{\rho}$ and considering the explicit dependencies of $\mathbf{g}$, the sensitivity of the objective function (5.20) can be represented by the following gradient:

$$\boldsymbol{\psi}' = \int_{t_0}^{t_F} \left(\mathbf{g}_{\mathbf{z}^i}\mathbf{z}^{i\prime} + \mathbf{g}_{\dot{\mathbf{z}}^i}\dot{\mathbf{z}}^{i\prime} + \mathbf{g}_{\ddot{\mathbf{z}}^i}\ddot{\mathbf{z}}^{i\prime} + \mathbf{g}_{\mathbf{z}}\mathbf{z}' + \mathbf{g}_{\dot{\mathbf{z}}}\dot{\mathbf{z}}' + \mathbf{g}_{\ddot{\mathbf{z}}}\ddot{\mathbf{z}}' + \mathbf{g}_{\mathbf{q}}\mathbf{q}' + \mathbf{g}_{\dot{\mathbf{q}}}\dot{\mathbf{q}}' + \mathbf{g}_{\ddot{\mathbf{q}}}\ddot{\mathbf{q}}' + \mathbf{g}_{\boldsymbol{\rho}}\right) \mathrm{d}t \tag{5.21}$$

where the $(\cdot)'$ denotes total derivatives with respect to the design parameters and the subscripts indicate partial derivatives.

The Matrix R formulation imposes the fulfillment of the vector of constraints at position, velocity and acceleration levels. Considering that the constraint equations at those levels are enforced, their differentiation with respect to the parameters of the system delivers the following sensitivity relations:

$$\boldsymbol{\Phi}' = \frac{\mathrm{d}\boldsymbol{\Phi}}{\mathrm{d}\boldsymbol{\rho}} = \mathbf{0} \Rightarrow \boldsymbol{\Phi}_{\hat{\mathbf{z}}}\mathbf{z}' = -\boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}} \tag{5.22a}$$

$$\dot{\boldsymbol{\Phi}}' = \frac{\mathrm{d}\dot{\boldsymbol{\Phi}}}{\mathrm{d}\boldsymbol{\rho}} = \mathbf{0} \Rightarrow \boldsymbol{\Phi}_{\hat{\mathbf{z}}}\dot{\mathbf{z}}' = -\mathbf{b}^{\boldsymbol{\rho}} \tag{5.22b}$$

$$\ddot{\boldsymbol{\Phi}}' = \frac{\mathrm{d}\ddot{\boldsymbol{\Phi}}}{\mathrm{d}\boldsymbol{\rho}} = \mathbf{0} \Rightarrow \boldsymbol{\Phi}_{\hat{\mathbf{z}}}\ddot{\mathbf{z}}' = -\mathbf{c}^{\boldsymbol{\rho}} \tag{5.22c}$$

where $\mathbf{b}^{\boldsymbol{\rho}}$ and $\mathbf{c}^{\boldsymbol{\rho}}$, which are not partial derivatives, are described in (5.5).

Equations (5.22a)-(5.22c) have to be completed with $d$ additional constraints like in (5.4), defining the value of sensitivities of the degrees of freedom,

$$\begin{bmatrix}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\\ \mathbf{B}\end{bmatrix}\mathbf{z}' = \begin{bmatrix}-\boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}}\\ \mathbf{z}^{i\prime}\end{bmatrix} \Rightarrow \mathbf{z}' = \mathbf{R}^{\boldsymbol{\Phi}}\mathbf{z}^{i\prime} - \mathbf{S}^{\boldsymbol{\Phi}}\boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}} \tag{5.23a}$$

$$\begin{bmatrix}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\\ \mathbf{B}\end{bmatrix}\dot{\mathbf{z}}' = \begin{bmatrix}-\mathbf{b}^{\boldsymbol{\rho}}\\ \dot{\mathbf{z}}^{i\prime} - \bar{\mathbf{b}}^{\boldsymbol{\rho}}\end{bmatrix} \Rightarrow \dot{\mathbf{z}}' = \mathbf{R}^{\boldsymbol{\Phi}}\left(\dot{\mathbf{z}}^{i\prime} - \bar{\mathbf{b}}^{\boldsymbol{\rho}}\right) - \mathbf{S}^{\boldsymbol{\Phi}}\mathbf{b}^{\boldsymbol{\rho}} \tag{5.23b}$$

$$\begin{bmatrix}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\\ \mathbf{B}\end{bmatrix}\ddot{\mathbf{z}}' = \begin{bmatrix}-\mathbf{c}^{\boldsymbol{\rho}}\\ \ddot{\mathbf{z}}^{i\prime} - \bar{\mathbf{c}}^{\boldsymbol{\rho}}\end{bmatrix} \Rightarrow \ddot{\mathbf{z}}' = \mathbf{R}^{\boldsymbol{\Phi}}\left(\ddot{\mathbf{z}}^{i\prime} - \bar{\mathbf{c}}^{\boldsymbol{\rho}}\right) - \mathbf{S}^{\boldsymbol{\Phi}}\mathbf{c}^{\boldsymbol{\rho}} \tag{5.23c}$$

being $\bar{\mathbf{b}}^{\boldsymbol{\rho}}$ and $\bar{\mathbf{c}}^{\boldsymbol{\rho}}$ defined in (5.5).

Expanding terms to make explicit the dependencies on the sensitivities of the independent coordinates in positions, velocities and accelerations, equations (5.23) are transformed:

$$\mathbf{z}' = \mathbf{R}^{\boldsymbol{\Phi}}\mathbf{z}^{i\prime} - \mathbf{S}^{\boldsymbol{\Phi}}\boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}} \tag{5.24a}$$

$$\dot{\mathbf{z}}' = \mathbf{R}^{\boldsymbol{\Phi}}\left(\dot{\mathbf{z}}^{i\prime} - \dot{\mathbf{B}}\mathbf{z}' + \mathbf{B}_{\hat{\rho}}\dot{\mathbf{z}}\right) - \mathbf{S}^{\boldsymbol{\Phi}}\left(\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\mathbf{z}' + \dot{\boldsymbol{\Phi}}_{\hat{\rho}}\right) \tag{5.24b}$$

$$\ddot{\mathbf{z}}' = \mathbf{R}^{\boldsymbol{\Phi}}\left(\ddot{\mathbf{z}}^{i\prime} - 2\dot{\mathbf{B}}\dot{\mathbf{z}}' + \ddot{\mathbf{B}}\mathbf{z}' + \mathbf{B}_{\hat{\rho}}\ddot{\mathbf{z}} + \dot{\mathbf{B}}_{\hat{\rho}}\dot{\mathbf{z}}\right) - \mathbf{S}^{\boldsymbol{\Phi}}\left(2\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\dot{\mathbf{z}}' + \ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\mathbf{z}' + \ddot{\boldsymbol{\Phi}}_{\hat{\rho}}\right) \tag{5.24c}$$

Every kinematic problem implicitly involved in semi-recursive Matrix R formulations (3.15), (3.19a) and (3.19b) evidences that joint coordinates can be expressed as a combination of terms referred to degrees of freedom and constraint equations. For a given instant of time, the implicit dependency $\mathbf{z} = \mathbf{z}\left(\mathbf{z}^i, \boldsymbol{\Phi}\left(\boldsymbol{\rho}\right)\right)$ can be inferred from the application of the implicit function theorem to equations (5.24) augmented with an appropriate selection of degrees of freedom through matrix $\mathbf{B}$:

$$\delta\mathbf{z} = \frac{\partial\mathbf{z}}{\partial\mathbf{z}^i}\delta\mathbf{z}^i + \frac{\partial\mathbf{z}}{\partial\boldsymbol{\Phi}}\boldsymbol{\Phi}_{\hat{\rho}}\delta\boldsymbol{\rho} \Rightarrow \mathbf{z}' = \mathbf{z}_{\mathbf{z}^i}\mathbf{z}^{i\prime} + \mathbf{z}_{\boldsymbol{\Phi}}\boldsymbol{\Phi}_{\hat{\rho}} \tag{5.25}$$

Analogously, the implicit relations $\dot{\mathbf{z}} = \dot{\mathbf{z}}\left(\dot{\mathbf{z}}^i, \dot{\boldsymbol{\Phi}}\left(\mathbf{z}, \boldsymbol{\rho}\right)\right)$ for velocities and $\ddot{\mathbf{z}} = \ddot{\mathbf{z}}\left(\ddot{\mathbf{z}}^i, \ddot{\boldsymbol{\Phi}}\left(\dot{\mathbf{z}}, \mathbf{z}, \boldsymbol{\rho}\right)\right)$ for joint coordinate accelerations can be deduced, yielding:

$$\delta\dot{\mathbf{z}} = \frac{\partial\dot{\mathbf{z}}}{\partial\dot{\mathbf{z}}^i}\delta\dot{\mathbf{z}}^i + \frac{\partial\dot{\mathbf{z}}}{\partial\mathbf{z}}\delta\mathbf{z} + \frac{\partial\dot{\mathbf{z}}}{\partial\dot{\boldsymbol{\Phi}}}\left(\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\mathbf{z}' + \dot{\boldsymbol{\Phi}}_{\hat{\rho}}\delta\boldsymbol{\rho}\right) \Rightarrow \dot{\mathbf{z}}' = \dot{\mathbf{z}}_{\dot{\mathbf{z}}^i}\dot{\mathbf{z}}^{i\prime} + \dot{\mathbf{z}}_{\mathbf{z}}\mathbf{z}' + \dot{\mathbf{z}}_{\dot{\boldsymbol{\Phi}}}\left(\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\mathbf{z}' + \dot{\boldsymbol{\Phi}}_{\hat{\rho}}\right) \tag{5.26a}$$

$$\begin{gathered}\delta\ddot{\mathbf{z}} = \frac{\partial\ddot{\mathbf{z}}}{\partial\ddot{\mathbf{z}}^i}\delta\ddot{\mathbf{z}}^i + \frac{\partial\ddot{\mathbf{z}}}{\partial\dot{\mathbf{z}}}\delta\dot{\mathbf{z}} + \frac{\partial\ddot{\mathbf{z}}}{\partial\mathbf{z}}\delta\mathbf{z} + \frac{\partial\ddot{\mathbf{z}}}{\partial\ddot{\boldsymbol{\Phi}}}\left(\ddot{\boldsymbol{\Phi}}_{\hat{\dot{\mathbf{z}}}}\dot{\mathbf{z}}' + \ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\mathbf{z}' + \ddot{\boldsymbol{\Phi}}_{\hat{\rho}}\delta\boldsymbol{\rho}\right) \Rightarrow \\ \ddot{\mathbf{z}}' = \ddot{\mathbf{z}}_{\ddot{\mathbf{z}}^i}\ddot{\mathbf{z}}^{i\prime} + \ddot{\mathbf{z}}_{\dot{\mathbf{z}}}\dot{\mathbf{z}}' + \ddot{\mathbf{z}}_{\mathbf{z}}\mathbf{z}' + \ddot{\mathbf{z}}_{\ddot{\boldsymbol{\Phi}}}\left(\ddot{\boldsymbol{\Phi}}_{\hat{\dot{\mathbf{z}}}}\dot{\mathbf{z}}' + \ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\mathbf{z}' + \ddot{\boldsymbol{\Phi}}_{\hat{\rho}}\right)\end{gathered} \tag{5.26b}$$

Comparing (5.25), (5.26a) and (5.26b) with (5.24), the following relations can be deduced:

$$\frac{\partial\mathbf{z}}{\partial\mathbf{z}^i} = \mathbf{z}_{\mathbf{z}^i} = \frac{\partial\dot{\mathbf{z}}}{\partial\dot{\mathbf{z}}^i} = \dot{\mathbf{z}}_{\dot{\mathbf{z}}^i} = \frac{\partial\ddot{\mathbf{z}}}{\partial\ddot{\mathbf{z}}^i} = \ddot{\mathbf{z}}_{\ddot{\mathbf{z}}^i} = \mathbf{R}^{\boldsymbol{\Phi}} \tag{5.27}$$

$$\frac{\partial\mathbf{z}}{\partial\boldsymbol{\Phi}} = \mathbf{z}_{\boldsymbol{\Phi}} = \frac{\partial\dot{\mathbf{z}}}{\partial\dot{\boldsymbol{\Phi}}} = \dot{\mathbf{z}}_{\dot{\boldsymbol{\Phi}}} = \frac{\partial\ddot{\mathbf{z}}}{\partial\ddot{\boldsymbol{\Phi}}} = \ddot{\mathbf{z}}_{\ddot{\boldsymbol{\Phi}}} = -\mathbf{S}^{\boldsymbol{\Phi}} \tag{5.28}$$

$$\frac{\partial\mathbf{z}}{\partial\boldsymbol{\rho}} = \mathbf{z}_{\boldsymbol{\rho}} = -\mathbf{S}^{\boldsymbol{\Phi}}\boldsymbol{\Phi}_{\hat{\rho}} \tag{5.29}$$

$$\frac{\partial\dot{\mathbf{z}}}{\partial\mathbf{z}} = \dot{\mathbf{z}}_{\mathbf{z}} = -\mathbf{S}^{\boldsymbol{\Phi}}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} - \mathbf{R}^{\boldsymbol{\Phi}}\dot{\mathbf{B}} \tag{5.30}$$

$$\frac{\partial\dot{\mathbf{z}}}{\partial\boldsymbol{\rho}} = \dot{\mathbf{z}}_{\boldsymbol{\rho}} = -\mathbf{S}^{\boldsymbol{\Phi}}\dot{\boldsymbol{\Phi}}_{\hat{\rho}} - \mathbf{R}^{\boldsymbol{\Phi}}\mathbf{B}_{\hat{\rho}}\dot{\mathbf{z}} \tag{5.31}$$

$$\frac{\partial\ddot{\mathbf{z}}}{\partial\dot{\mathbf{z}}} = \ddot{\mathbf{z}}_{\dot{\mathbf{z}}} = -2\mathbf{S}^{\boldsymbol{\Phi}}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} - 2\mathbf{R}^{\boldsymbol{\Phi}}\dot{\mathbf{B}} \tag{5.32}$$

$$\frac{\partial\ddot{\mathbf{z}}}{\partial\mathbf{z}} = \ddot{\mathbf{z}}_{\mathbf{z}} = -\mathbf{S}^{\boldsymbol{\Phi}}\ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} - \mathbf{R}^{\boldsymbol{\Phi}}\ddot{\mathbf{B}} \tag{5.33}$$

$$\frac{\partial\ddot{\mathbf{z}}}{\partial\boldsymbol{\rho}} = \ddot{\mathbf{z}}_{\boldsymbol{\rho}} = -\mathbf{S}^{\boldsymbol{\Phi}}\ddot{\boldsymbol{\Phi}}_{\hat{\rho}} - \mathbf{R}^{\boldsymbol{\Phi}}\left(\mathbf{B}_{\hat{\rho}}\ddot{\mathbf{z}} + \dot{\mathbf{B}}_{\hat{\rho}}\dot{\mathbf{z}}\right) \tag{5.34}$$

The expressions whose sensitivity analysis is been addressed include the definition of a non-constant $\mathbf{B}$ matrix, i.e. the possibility of independent coordinates non belonging to the joint-coordinates vector. Behold that the expressions developed are generic for any selection of degrees of freedom. In fact, if the vector of independent coordinates are selected such as the matrix $\mathbf{B}$ is constant, the expressions presented are still valid (with the derivatives of $\mathbf{B}$ null).

Once established the derivative expressions of $\mathbf{z}$, $\dot{\mathbf{z}}$ and $\ddot{\mathbf{z}}$ in (5.27), the dependent variables $\mathbf{q}'$, $\dot{\mathbf{q}}'$, $\ddot{\mathbf{q}}'$, $\mathbf{z}'$, $\dot{\mathbf{z}}'$, $\ddot{\mathbf{z}}'$ can be removed from the gradient of the objective function (5.21). Then, all the contributions involving $\mathbf{z}^{i\prime}$, $\dot{\mathbf{z}}^{i\prime}$ and $\ddot{\mathbf{z}}^{i\prime}$ can be gathered together:

$$\begin{aligned}\mathbf{g}_{\check{\mathbf{z}}^i} = \mathbf{g}_{\mathbf{z}^i} + \left[\mathbf{g}_{\hat{\mathbf{z}}} + \mathbf{g}_{\hat{\dot{\mathbf{z}}}}\dot{\mathbf{z}}_{\mathbf{z}} + \mathbf{g}_{\hat{\ddot{\mathbf{z}}}}\left(\ddot{\mathbf{z}}_{\dot{\mathbf{z}}}\dot{\mathbf{z}}_{\mathbf{z}} + \ddot{\mathbf{z}}_{\mathbf{z}}\right)\right]\mathbf{z}_{\mathbf{z}^i} = \mathbf{g}_{\mathbf{z}^i} + \Big[\mathbf{g}_{\hat{\mathbf{z}}} + \mathbf{g}_{\hat{\dot{\mathbf{z}}}}\left(-\mathbf{S}^{\boldsymbol{\Phi}}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} - \mathbf{R}^{\boldsymbol{\Phi}}\dot{\mathbf{B}}\right) \\ +\mathbf{g}_{\hat{\ddot{\mathbf{z}}}}\left(\left(-2\mathbf{S}^{\boldsymbol{\Phi}}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} - 2\mathbf{R}^{\boldsymbol{\Phi}}\dot{\mathbf{B}}\right)\left(-\mathbf{S}^{\boldsymbol{\Phi}}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} - \mathbf{R}^{\boldsymbol{\Phi}}\dot{\mathbf{B}}\right) + \left(-\mathbf{S}^{\boldsymbol{\Phi}}\ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} - \mathbf{R}^{\boldsymbol{\Phi}}\ddot{\mathbf{B}}\right)\right)\Big]\mathbf{R}^{\boldsymbol{\Phi}}\end{aligned} \tag{5.35a}$$

$$\mathbf{g}_{\check{\dot{\mathbf{z}}}^i} = \mathbf{g}_{\dot{\mathbf{z}}^i} + \left[\mathbf{g}_{\hat{\dot{\mathbf{z}}}} + \mathbf{g}_{\hat{\ddot{\mathbf{z}}}}\ddot{\mathbf{z}}_{\dot{\mathbf{z}}}\right]\dot{\mathbf{z}}_{\dot{\mathbf{z}}^i} = \mathbf{g}_{\dot{\mathbf{z}}^i} + \left[\mathbf{g}_{\hat{\dot{\mathbf{z}}}} + \mathbf{g}_{\hat{\ddot{\mathbf{z}}}}\left(-2\mathbf{S}^{\boldsymbol{\Phi}}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} - 2\mathbf{R}^{\boldsymbol{\Phi}}\dot{\mathbf{B}}\right)\right]\mathbf{R}^{\boldsymbol{\Phi}} \tag{5.35b}$$

$$\mathbf{g}_{\check{\ddot{\mathbf{z}}}^i} = \mathbf{g}_{\ddot{\mathbf{z}}^i} + \mathbf{g}_{\hat{\ddot{\mathbf{z}}}}\ddot{\mathbf{z}}_{\ddot{\mathbf{z}}^i} = \mathbf{g}_{\ddot{\mathbf{z}}^i} + \mathbf{g}_{\hat{\ddot{\mathbf{z}}}}\mathbf{R}^{\boldsymbol{\Phi}} \tag{5.35c}$$

$$\mathbf{g}_{\check{\boldsymbol{\rho}}} = \mathbf{g}_{\hat{\boldsymbol{\rho}}} + \mathbf{g}_{\hat{\mathbf{z}}}\mathbf{z}_{\boldsymbol{\rho}} + \mathbf{g}_{\hat{\dot{\mathbf{z}}}}\left(\dot{\mathbf{z}}_{\mathbf{z}}\mathbf{z}_{\boldsymbol{\rho}} + \dot{\mathbf{z}}_{\boldsymbol{\rho}}\right) + \mathbf{g}_{\hat{\ddot{\mathbf{z}}}}\left(\ddot{\mathbf{z}}_{\dot{\mathbf{z}}}\left(\dot{\mathbf{z}}_{\mathbf{z}}\mathbf{z}_{\boldsymbol{\rho}} + \dot{\mathbf{z}}_{\boldsymbol{\rho}}\right) + \ddot{\mathbf{z}}_{\mathbf{z}}\mathbf{z}_{\boldsymbol{\rho}} + \ddot{\mathbf{z}}_{\boldsymbol{\rho}}\right) \tag{5.35d}$$

where $(\cdot)_{\check{}}$ identifies a partial derivative including implicit dependencies with respect to joint coordinates:

$$(\cdot)_{\check{\mathbf{x}}} = (\cdot)_{\hat{\mathbf{x}}} + (\cdot)_{\hat{\mathbf{z}}}\mathbf{z}_{\mathbf{x}} + (\cdot)_{\hat{\dot{\mathbf{z}}}}\left(\dot{\mathbf{z}}_{\mathbf{x}} + \dot{\mathbf{z}}_{\mathbf{z}}\mathbf{z}_{\mathbf{x}}\right) + (\cdot)_{\hat{\ddot{\mathbf{z}}}}\left(\ddot{\mathbf{z}}_{\mathbf{x}} + \ddot{\mathbf{z}}_{\dot{\mathbf{z}}}\dot{\mathbf{z}}_{\mathbf{x}} + \left(\ddot{\mathbf{z}}_{\mathbf{z}} + \ddot{\mathbf{z}}_{\dot{\mathbf{z}}}\dot{\mathbf{z}}_{\mathbf{z}}\right)\mathbf{z}_{\mathbf{x}}\right) \tag{5.36}$$

Note that the substitution of the derivatives with respect to $\boldsymbol{\rho}$ in (5.35) is omitted to clarify the resulting equations since the replacement is straightforward and no significant simplifications can be applied. Observe that the derivatives $\mathbf{g}_{\hat{\mathbf{z}}}$, $\mathbf{g}_{\hat{\dot{\mathbf{z}}}}$, $\mathbf{g}_{\hat{\ddot{\mathbf{z}}}}$ and $\mathbf{g}_{\hat{\boldsymbol{\rho}}}$ indicate the differentiation rule exposed in (4.8) in chapter 4.

Finally, gathering the terms with respect to $\mathbf{z}^{i\prime}$, $\dot{\mathbf{z}}^{i\prime}$ and $\ddot{\mathbf{z}}^{i\prime}$, the expression of the gradient results:

$$\boldsymbol{\psi}' = \int_{t_0}^{t_F}\left(\mathbf{g}_{\check{\mathbf{z}}^i}\mathbf{z}^{i\prime} + \mathbf{g}_{\check{\dot{\mathbf{z}}}^i}\dot{\mathbf{z}}^{i\prime} + \mathbf{g}_{\check{\ddot{\mathbf{z}}}^i}\ddot{\mathbf{z}}^{i\prime} + \mathbf{g}_{\check{\boldsymbol{\rho}}}\right)\mathrm{d}t \tag{5.37}$$

The sensitivities $\mathbf{z}^{i\prime}$, $\dot{\mathbf{z}}^{i\prime}$ and $\ddot{\mathbf{z}}^{i\prime}$ can be obtained differentiating (3.27) with respect to the vector of parameters. First of all, let us transform the system (3.27) to use a more compact notation:

$$\bar{\mathbf{M}}\ddot{\mathbf{z}}^i = \bar{\mathbf{Q}} \tag{5.38}$$

with

$$\bar{\mathbf{M}} = \left(\mathbf{R}^{\boldsymbol{\Phi}\mathrm{T}}\mathbf{M}^d\mathbf{R}^{\boldsymbol{\Phi}}\right) \tag{5.39}$$

$$\bar{\mathbf{Q}} = \mathbf{R}^{\boldsymbol{\Phi}\mathrm{T}}\left(\mathbf{Q}^d - \mathbf{M}^d\left(\mathbf{S}^{\boldsymbol{\Phi}}\mathbf{c} - \mathbf{R}^{\boldsymbol{\Phi}}\dot{\mathbf{B}}\dot{\mathbf{z}}\right)\right) \tag{5.40}$$

Then, taking derivatives in (5.38) with respect to the set of parameters $\boldsymbol{\rho}$:

$$\bar{\mathbf{M}}'\ddot{\mathbf{z}}^i + \bar{\mathbf{M}}\ddot{\mathbf{z}}^{i\prime} = \bar{\mathbf{Q}}' \tag{5.41}$$

where:

$$\bar{\mathbf{Q}}' = \frac{\mathrm{d}\bar{\mathbf{Q}}}{\mathrm{d}\boldsymbol{\rho}} = \bar{\mathbf{Q}}_{\check{\mathbf{z}}^i}\mathbf{z}^{i\prime} + \bar{\mathbf{Q}}_{\check{\dot{\mathbf{z}}}^i}\dot{\mathbf{z}}^{i\prime} + \bar{\mathbf{Q}}_{\check{\boldsymbol{\rho}}} = -\bar{\mathbf{K}}\mathbf{z}^{i\prime} - \bar{\mathbf{C}}\dot{\mathbf{z}}^{i\prime} + \bar{\mathbf{Q}}_{\check{\boldsymbol{\rho}}} \tag{5.42}$$

$$\bar{\mathbf{M}}'\ddot{\mathbf{z}}^i = \frac{\mathrm{d}\bar{\mathbf{M}}}{\mathrm{d}\boldsymbol{\rho}}\ddot{\mathbf{z}}^i = \left(\bar{\mathbf{M}}_{\check{\mathbf{z}}^i}\ddot{\mathbf{z}}^i\right)\mathbf{z}^{i\prime} + \bar{\mathbf{M}}_{\check{\boldsymbol{\rho}}}\ddot{\mathbf{z}}^i \tag{5.43}$$

The resulting Tangent Linear Model (hereinafter TLM) takes the form:

$$\bar{\mathbf{M}}\ddot{\mathbf{z}}^{i\prime} + \bar{\mathbf{C}}\dot{\mathbf{z}}^{i\prime} + \left(\bar{\mathbf{K}} + \bar{\mathbf{M}}_{\mathbf{z}^i}\ddot{\mathbf{z}}^i\right)\mathbf{z}^{i\prime} = \bar{\mathbf{Q}}_{\check{\boldsymbol{\rho}}} - \bar{\mathbf{M}}_{\check{\boldsymbol{\rho}}}\ddot{\mathbf{z}}^i \tag{5.44a}$$

$$\mathbf{z}^{i\prime}\left(t_0\right) = \mathbf{z}_0^{i\prime} \tag{5.44b}$$

$$\dot{\mathbf{z}}^{i\prime}\left(t_0\right) = \dot{\mathbf{z}}_0^{i\prime} \tag{5.44c}$$

wherein:

$$\bar{\mathbf{Q}}_{\check{\boldsymbol{\rho}}} - \bar{\mathbf{M}}_{\check{\boldsymbol{\rho}}}\ddot{\mathbf{z}}^i = \bar{\mathbf{Q}}_{\hat{\boldsymbol{\rho}}} - \bar{\mathbf{M}}_{\hat{\boldsymbol{\rho}}}\ddot{\mathbf{z}}^i + \left(\bar{\mathbf{Q}}_{\hat{\mathbf{z}}} + \bar{\mathbf{Q}}_{\hat{\dot{\mathbf{z}}}}\dot{\mathbf{z}}_{\mathbf{z}} - \bar{\mathbf{M}}_{\hat{\mathbf{z}}}\ddot{\mathbf{z}}^i\right)\mathbf{z}_{\boldsymbol{\rho}} + \bar{\mathbf{Q}}_{\hat{\dot{\mathbf{z}}}}\dot{\mathbf{z}}_{\boldsymbol{\rho}} \tag{5.45}$$

$$\bar{\mathbf{K}} = -\bar{\mathbf{Q}}_{\mathbf{z}^i} = -\left(\bar{\mathbf{Q}}_{\hat{\mathbf{z}}} + \bar{\mathbf{Q}}_{\hat{\dot{\mathbf{z}}}}\dot{\mathbf{z}}_{\mathbf{z}}\right)\mathbf{z}_{\mathbf{z}^i} \tag{5.46}$$

$$\bar{\mathbf{C}} = -\bar{\mathbf{Q}}_{\dot{\mathbf{z}}^i} = -\bar{\mathbf{Q}}_{\hat{\dot{\mathbf{z}}}}\dot{\mathbf{z}}_{\dot{\mathbf{z}}^i} \tag{5.47}$$

$$\bar{\mathbf{M}}_{\check{\mathbf{z}}^i}\ddot{\mathbf{z}}^i = \left(\bar{\mathbf{M}}_{\hat{\mathbf{z}}}\ddot{\mathbf{z}}^i\right)\mathbf{z}_{\mathbf{z}^i} \tag{5.48}$$

The terms involving derivatives of $\boldsymbol{\rho}$, i.e. $\bar{\mathbf{Q}}_{\hat{\boldsymbol{\rho}}}$ and $\bar{\mathbf{M}}_{\hat{\boldsymbol{\rho}}}\ddot{\mathbf{z}}^i$, can be computed more efficiently if both are evaluated together. In fact, the terms referred to forces and masses of each body are accumulated following the same scheme, hence it is convenient to execute the differentiation of the accumulation-related terms only once for the whole expression instead of twice, one for forces and another one for masses. In order to display this relation, the semi-recursive Matrix R dynamic expressions can be reformulated as:

$$\mathbf{R}^{\boldsymbol{\Phi}\mathrm{T}}\left(\mathbf{Q}^d - \mathbf{M}^d\left(\mathbf{R}^{\boldsymbol{\Phi}}\left(\ddot{\mathbf{z}}^i - \dot{\mathbf{B}}\dot{\mathbf{z}}\right) + \mathbf{S}^{\boldsymbol{\Phi}}\mathbf{c}\right)\right) = \mathbf{0} \tag{5.49}$$

Taking into account expression (3.26), the equation (5.49) can be transformed into:

$$\mathbf{R}^{\boldsymbol{\Phi}\mathrm{T}}\left(\mathbf{Q}^d - \mathbf{M}^d\ddot{\mathbf{z}}\right) = \mathbf{0} \tag{5.50}$$

Resorting now to the reference point coordinates accelerations vector $\dot{\mathbf{V}}$ and using (2.203), (2.206) and the definition of the mass matrix and generalized forces vector in terms of joint coordinates given by (2.208b) and (2.208c), equation (5.50) becomes:

$$\mathbf{R}^{\boldsymbol{\Phi}\mathrm{T}}\mathbf{R}^{v\mathrm{T}}\left(\mathbf{Q}^v - \mathbf{M}^v\dot{\mathbf{V}}\right) = \mathbf{0} \tag{5.51}$$

Behold that any of the equations presented above is valid for the differentiation, even though they are expressed in terms of independent coordinates, in joint coordinates or in reference point coordinates. Exploring each one of equations (5.49), (5.50) and (5.51), it can be deduced that (5.51) is the most straightforward to compose, thus it is the most suitable to be differentiated. Moreover, as introduced in section (2.4.2), $\mathbf{Q}^v - \mathbf{M}^v\dot{\mathbf{V}}$ can be efficiently evaluated following a fully-recursive scheme. The efficient evaluation of the derivatives of $\mathbf{Q}^v - \mathbf{M}^v\dot{\mathbf{V}}$ will be commented in section (6.3.1) corresponding to implementation considerations.

Taking derivatives on (5.51):

$$\begin{aligned}\bar{\mathbf{Q}}_{\hat{\boldsymbol{\rho}}} - \bar{\mathbf{M}}_{\hat{\boldsymbol{\rho}}}\ddot{\mathbf{z}}^i = \mathbf{R}^{\boldsymbol{\Phi}\mathrm{T}}_{\hat{\boldsymbol{\rho}}}\mathbf{R}^{v\mathrm{T}}\left(\mathbf{Q}^v - \mathbf{M}^v\dot{\mathbf{V}}\right) + \mathbf{R}^{\boldsymbol{\Phi}\mathrm{T}}\mathbf{R}^{v\mathrm{T}}_{\hat{\boldsymbol{\rho}}}\left(\mathbf{Q}^v - \mathbf{M}^v\dot{\mathbf{V}}\right) \\ + \mathbf{R}^{\boldsymbol{\Phi}\mathrm{T}}\mathbf{R}^{v\mathrm{T}}\left(\mathbf{Q}^v_{\hat{\boldsymbol{\rho}}} - \mathbf{M}^v_{\hat{\boldsymbol{\rho}}}\dot{\mathbf{V}} - \mathbf{M}^v\dot{\mathbf{V}}_{\hat{\boldsymbol{\rho}}}\right)\end{aligned} \tag{5.52}$$

The special interest of these expressions relies on the elimination of duplicities on the differentiation process and on the particular assembly using a forward computation of the sensitivities of the kinematics ($\dot{\mathbf{V}}_{\hat{\boldsymbol{\rho}}}$) from the root to the tips of the mechanism, and other evaluation and accumulation of masses and forces derivatives from the tips to the root. With this scheme, either in the differentiation of the kinematic part or of the forces and masses, only the expressions of the previous term of the kinematic chain have to be stored to compute the derivative referred to the subsequent body, hence the use of memory is also improved. A more detailed description is included in section 6.3.1.

Back into expressions (5.45) to (5.48), the following derivatives are needed as well:

$$\begin{aligned}\bar{\mathbf{Q}}_{\hat{\mathbf{z}}} = \mathbf{R}^{\boldsymbol{\Phi}\mathrm{T}}_{\mathbf{z}}\left(\mathbf{Q} - \mathbf{M}\left(\mathbf{S}^{\boldsymbol{\Phi}}\mathbf{c} - \mathbf{R}^{\boldsymbol{\Phi}}\dot{\mathbf{B}}\dot{\mathbf{z}}\right)\right) - \mathbf{R}^{\boldsymbol{\Phi}\mathrm{T}}\left(\mathbf{K} + \mathbf{M}_{\hat{\mathbf{z}}}\left(\mathbf{S}^{\boldsymbol{\Phi}}\mathbf{c} - \mathbf{R}^{\boldsymbol{\Phi}}\dot{\mathbf{B}}\dot{\mathbf{z}}\right) \right. \\ \left. + \mathbf{M}\left(\mathbf{S}^{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\mathbf{c} + \mathbf{S}^{\boldsymbol{\Phi}}\mathbf{c}_{\hat{\mathbf{z}}} - \mathbf{R}^{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\dot{\mathbf{B}}\dot{\mathbf{z}} - \mathbf{R}^{\boldsymbol{\Phi}}\dot{\mathbf{B}}_{\hat{\mathbf{z}}}\dot{\mathbf{z}} - \mathbf{R}^{\boldsymbol{\Phi}}\dot{\mathbf{B}}\dot{\mathbf{z}}_{\hat{\mathbf{z}}}\right)\right)\end{aligned}, \tag{5.53}$$

$$\bar{\mathbf{Q}}_{\hat{\dot{\mathbf{z}}}} = -\mathbf{R}^{\boldsymbol{\Phi}\mathrm{T}}\left(\mathbf{C} + \mathbf{M}\left(\mathbf{S}^{\boldsymbol{\Phi}}\mathbf{c}_{\hat{\dot{\mathbf{z}}}} - 2\mathbf{R}^{\boldsymbol{\Phi}}\dot{\mathbf{B}}\right)\right) \tag{5.54}$$

The derivatives of $\mathbf{S}^{\boldsymbol{\Phi}}$ and $\mathbf{R}^{\boldsymbol{\Phi}}$ can be attained using the properties of the derivative of the inverse matrix applied to (5.55), recalled below:

$$\begin{bmatrix}\mathbf{S}^{\boldsymbol{\Phi}} & \mathbf{R}^{\boldsymbol{\Phi}}\end{bmatrix}\begin{bmatrix}\boldsymbol{\Phi}_{\hat{\mathbf{z}}} \\ \mathbf{B}\end{bmatrix} = \mathbf{I}_n \Rightarrow \begin{bmatrix}\mathbf{S}^{\boldsymbol{\Phi}} & \mathbf{R}^{\boldsymbol{\Phi}}\end{bmatrix} = \begin{bmatrix}\boldsymbol{\Phi}_{\hat{\mathbf{z}}} \\ \mathbf{B}\end{bmatrix}^{-1} \tag{5.55}$$

Thus, the generic derivatives of $\mathbf{S}^{\boldsymbol{\Phi}}$ and $\mathbf{R}^{\boldsymbol{\Phi}}$ can be expressed as:

$$\frac{\partial\begin{bmatrix}\mathbf{S}^{\boldsymbol{\Phi}} & \mathbf{R}^{\boldsymbol{\Phi}}\end{bmatrix}}{\partial\mathbf{x}} = -\begin{bmatrix}\boldsymbol{\Phi}_{\hat{\mathbf{z}}} \\ \mathbf{B}\end{bmatrix}^{-1}\begin{bmatrix}\boldsymbol{\Phi}_{\hat{\mathbf{z}}\mathbf{x}} \\ \mathbf{B}_{\mathbf{x}}\end{bmatrix}\begin{bmatrix}\boldsymbol{\Phi}_{\hat{\mathbf{z}}} \\ \mathbf{B}\end{bmatrix}^{-1} = -\begin{bmatrix}\mathbf{S}^{\boldsymbol{\Phi}} & \mathbf{R}^{\boldsymbol{\Phi}}\end{bmatrix}\begin{bmatrix}\boldsymbol{\Phi}_{\hat{\mathbf{z}}\mathbf{x}} \\ \mathbf{B}_{\mathbf{x}}\end{bmatrix}\begin{bmatrix}\mathbf{S}^{\boldsymbol{\Phi}} & \mathbf{R}^{\boldsymbol{\Phi}}\end{bmatrix} \tag{5.56}$$

The particularization of (5.56) to derivatives with respect to $\mathbf{z}$ and $\boldsymbol{\rho}$, using the rule of differentiation (4.9), yields:

$$\mathbf{S}^{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} = -\left(\mathbf{S}^{\boldsymbol{\Phi}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}} + \mathbf{R}^{\boldsymbol{\Phi}}\mathbf{B}_{\hat{\mathbf{z}}}\right)\mathbf{S}^{\boldsymbol{\Phi}} \tag{5.57}$$

$$\mathbf{R}^{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} = -\left(\mathbf{S}^{\boldsymbol{\Phi}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}} + \mathbf{R}^{\boldsymbol{\Phi}}\mathbf{B}_{\hat{\mathbf{z}}}\right)\mathbf{R}^{\boldsymbol{\Phi}} \tag{5.58}$$

$$\mathbf{S}^{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}} = -\left(\mathbf{S}^{\boldsymbol{\Phi}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}} + \mathbf{R}^{\boldsymbol{\Phi}}\mathbf{B}_{\hat{\boldsymbol{\rho}}}\right)\mathbf{S}^{\boldsymbol{\Phi}} \tag{5.59}$$

$$\mathbf{R}^{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}} = -\left(\mathbf{S}^{\boldsymbol{\Phi}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}} + \mathbf{R}^{\boldsymbol{\Phi}}\mathbf{B}_{\hat{\boldsymbol{\rho}}}\right)\mathbf{R}^{\boldsymbol{\Phi}} \tag{5.60}$$

Observe that if a constant $\mathbf{B}$ matrix is selected, its derivatives will be null and terms $\mathbf{R}^{\mathbf{\Phi}}\mathbf{B}_{\hat{\mathbf{z}}}$ and $\mathbf{R}^{\mathbf{\Phi}}\mathbf{B}_{\hat{\boldsymbol{\rho}}}$ will disappear from (5.57).

The remaining derivatives of masses and forces $\mathbf{M}_{\hat{\mathbf{z}}}$, $\mathbf{K}$, $\mathbf{C}$, $\mathbf{M}_{\hat{\boldsymbol{\rho}}}$ and $\mathbf{Q}_{\hat{\boldsymbol{\rho}}}$ have been already developed and explained in the previous chapter 4 devoted to the sensitivity analysis of open-loop systems.

The sensitivity analysis of the semi-recursive Matrix R formulation has been implemented considering a non-constant $\mathbf{B}$ matrix and the RTdyn0 and RTdyn1 approaches. The results of these sensitivities match the expected ones for all the multibody models tested, being the computational time for this formulation between the sensitivities of the slower natural coordinates formulations and the fastest ALI3-P sensitivities for the experiments considered in this work. Furthermore, the time saving related to the use of a constant $\mathbf{B}$ matrix instead of a non-constant one is below 10% in all the numerical tests executed, which indicates that the inclusion of the variable matrix has an assumable effect in the computational time without impairing the accuracy of the results.

### 5.2.2 Adjoint sensitivity

The adjoint variable method applied to Matrix R formulations was presented and discussed in [107], starting from three different constructions of the equations of motion: a first-order explicit ODE system, a first-order implicit ODE system and a second-order implicit ODE system. The first option delivers the simplest possible expressions for the adjoint equations, avoiding the time derivative of the mass matrix (as in the first-order implicit ODE system) and the time derivative of the damping matrix and the second time derivative of the mass matrix (as in the second-order implicit ODE system). In the current work, the developments presented in [107] for the first-order ODE system will be recalled and combined with the semi-recursive formalism.

The system (5.38) can be transformed into a first-order implicit system by introducing a new set of variables $\dot{\mathbf{z}}^i = \mathbf{v}$, and defining the new vector of first-order states $\mathbf{y} = \begin{bmatrix} \mathbf{z}^{i\mathrm{T}} & \mathbf{v}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$

$$\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \bar{\mathbf{M}} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{z}}^i \\ \dot{\mathbf{v}} \end{bmatrix} = \begin{bmatrix} \mathbf{v} \\ \bar{\mathbf{Q}} \end{bmatrix} \tag{5.61a}$$

$$\hat{\mathbf{M}}(\mathbf{y},\boldsymbol{\rho})\,\dot{\mathbf{y}} = \hat{\mathbf{Q}}(t,\mathbf{y},\boldsymbol{\rho}) \tag{5.61b}$$

Taking the inverse of the leading matrix in (5.61b), the system can be expressed as a first-order explicit one,

$$\dot{\mathbf{y}} = \hat{\mathbf{M}}^{-1}(\mathbf{y},\boldsymbol{\rho})\,\hat{\mathbf{Q}}(t,\mathbf{y},\boldsymbol{\rho}) = \mathbf{f}(t,\mathbf{y},\boldsymbol{\rho}) \tag{5.62}$$

Let us now consider the following Lagrangian:

$$\mathcal{L}(\boldsymbol{\rho}) = \psi - \int_{t_0}^{t_F} \boldsymbol{\mu}^{\mathrm{T}}\left(\dot{\mathbf{y}} - \mathbf{f}(t,\mathbf{y},\boldsymbol{\rho})\right)\mathrm{d}t \tag{5.63}$$

Since $\dot{\mathbf{y}} - \mathbf{f}\left(t, \mathbf{y}, \boldsymbol{\rho}\right) = \mathbf{0}$, the value of the Lagrangian is equal to the value of the objective function, and also its derivatives for any value of $\boldsymbol{\mu}$.

Computing the infinitesimal variations of $\mathcal{L}$,

$$\begin{aligned}\delta\mathcal{L} = \int_{t_0}^{t_F} \left(\mathbf{g}_{\check{\mathbf{y}}}\delta\mathbf{y} + \mathbf{g}_{\check{\dot{\mathbf{y}}}}\delta\dot{\mathbf{y}} + \mathbf{g}_{\check{\boldsymbol{\rho}}}\delta\boldsymbol{\rho}\right)\mathrm{d}t - \int_{t_0}^{t_F} \delta\boldsymbol{\mu}^{\mathrm{T}}\left(\dot{\mathbf{y}} - \mathbf{f}\left(t, \mathbf{y}, \boldsymbol{\rho}\right)\right)\mathrm{d}t \\ - \int_{t_0}^{t_F} \boldsymbol{\mu}^{\mathrm{T}}\left(\delta\dot{\mathbf{y}} - \mathbf{f}_{\check{\mathbf{y}}}\delta\mathbf{y} - \mathbf{f}_{\check{\boldsymbol{\rho}}}\delta\boldsymbol{\rho}\right)\mathrm{d}t\end{aligned} \tag{5.64}$$

in which the notation explained in equation (5.36) has been used to include implicit dependencies on joint-coordinates in the partial derivatives with respect to $\mathbf{y}$, $\dot{\mathbf{y}}$ and $\boldsymbol{\rho}$.

Integrating by parts and rearranging terms:

$$\begin{aligned}\delta\mathcal{L} = \left[\left(\mathbf{g}_{\check{\dot{\mathbf{y}}}} - \boldsymbol{\mu}^{\mathrm{T}}\right)\delta\mathbf{y}\right]_{t_0}^{t_F} + \int_{t_0}^{t_F} \left(\mathbf{g}_{\check{\mathbf{y}}} - \dot{\mathbf{g}}_{\check{\dot{\mathbf{y}}}} + \boldsymbol{\mu}^{\mathrm{T}}\mathbf{f}_{\check{\mathbf{y}}} + \dot{\boldsymbol{\mu}}^{\mathrm{T}}\right)\delta\mathbf{y}\mathrm{d}t \\ + \int_{t_0}^{t_F} \left(\mathbf{g}_{\check{\boldsymbol{\rho}}} + \boldsymbol{\mu}^{\mathrm{T}}\mathbf{f}_{\check{\boldsymbol{\rho}}}\right)\delta\boldsymbol{\rho}\mathrm{d}t\end{aligned} \tag{5.65}$$

The objective of the adjoint approach is to eliminate the need of calculating the derivatives of the states. In this case the objective is to nullify the expression multiplying $\delta\mathbf{y}$, leading to the following adjoint ODE system:

$$\dot{\boldsymbol{\mu}} = -\mathbf{f}_{\check{\mathbf{y}}}{}^{\mathrm{T}}\boldsymbol{\mu} - \mathbf{g}_{\check{\mathbf{y}}}{}^{\mathrm{T}} + \dot{\mathbf{g}}_{\check{\dot{\mathbf{y}}}}{}^{\mathrm{T}} \tag{5.66a}$$

$$\boldsymbol{\mu}^{t_F} = \left[\mathbf{g}_{\check{\dot{\mathbf{y}}}}{}^{\mathrm{T}}\right]^{t_F} \tag{5.66b}$$

$$\tag{5.66c}$$

Nevertheless, these equations involve some problems related to the dependencies on $\dot{\mathbf{y}}$ and the derivative $\dot{\mathbf{g}}_{\check{\dot{\mathbf{y}}}}$, which could need the calculation of the jerks $\dddot{\mathbf{z}}^i$. These problems can be solved through the transformation of the dependencies on $\dot{\mathbf{y}}$ to implicit dependencies on $\mathbf{y}$ and $\boldsymbol{\rho}$ using (5.61b), resulting the final adjoint system:

$$\dot{\boldsymbol{\mu}} = -\mathbf{f}_{\check{\mathbf{y}}}{}^{\mathrm{T}}\left(\boldsymbol{\mu} + \mathbf{g}_{\check{\dot{\mathbf{y}}}}{}^{\mathrm{T}}\right) - \mathbf{g}_{\check{\mathbf{y}}}{}^{\mathrm{T}} \tag{5.67a}$$

$$\boldsymbol{\mu}^{t_F} = \mathbf{0} \tag{5.67b}$$

$$\mathbf{f}_{\check{\mathbf{y}}} = \hat{\mathbf{M}}^{-1}\left(\hat{\mathbf{Q}}_{\check{\mathbf{y}}} - \hat{\mathbf{M}}_{\check{\mathbf{y}}}\mathbf{f}\right) = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\bar{\mathbf{M}}^{-1}\left(\bar{\mathbf{K}} + \bar{\mathbf{M}}_{\mathbf{z}^i}\dot{\mathbf{v}}\right) & -\bar{\mathbf{M}}^{-1}\bar{\mathbf{C}} \end{bmatrix} \tag{5.67c}$$

$$\mathbf{g}_{\check{\mathbf{y}}} = \begin{bmatrix}\mathbf{g}_{\check{\mathbf{z}}^i} & \mathbf{g}_{\check{\dot{\mathbf{z}}}^i}\end{bmatrix} \tag{5.67d}$$

$$\mathbf{g}_{\check{\dot{\mathbf{y}}}} = \begin{bmatrix}\mathbf{0} & \mathbf{g}_{\check{\ddot{\mathbf{z}}}^i}\end{bmatrix} \tag{5.67e}$$

The objective function gradient can be calculated with the remaining terms:

$$\boldsymbol{\psi}'^{\mathrm{T}} = -\left[\mathbf{y}_{\check{\boldsymbol{\rho}}}^{\mathrm{T}}\boldsymbol{\mu}\right]_{t_0} + \int_{t_0}^{t_F} \left(\mathbf{f}_{\check{\boldsymbol{\rho}}}^{\mathrm{T}}\left(\boldsymbol{\mu} + \mathbf{g}_{\check{\dot{\mathbf{y}}}}{}^{\mathrm{T}}\right) + \mathbf{g}_{\check{\boldsymbol{\rho}}}^{\mathrm{T}}\right) \mathrm{d}t \tag{5.68a}$$

$$\mathbf{f}_{\check{\boldsymbol{\rho}}} = \hat{\mathbf{M}}^{-1}\left(\hat{\mathbf{Q}}_{\check{\boldsymbol{\rho}}} - \hat{\mathbf{M}}_{\check{\boldsymbol{\rho}}}\mathbf{f}\right) = \left[\begin{array}{c} \mathbf{0} \\ \bar{\mathbf{M}}^{-1}\left(\bar{\mathbf{Q}}_{\check{\boldsymbol{\rho}}} - \bar{\mathbf{M}}_{\check{\boldsymbol{\rho}}}\ddot{\mathbf{z}}^{i}\right) \end{array}\right] \tag{5.68b}$$

Regarding the simplicity and generality of the adjoint method applied to the semi-recursive Matrix R formulation, it constitutes an interesting approach for the implementation of the sensitivity analysis of the dynamic response of a multibody system. These sensitivities have been implemented in the MBSLIM multibody library and have been tested with excellent results in terms of accuracy. Furthermore, the possibility of using natural coordinates as degrees of freedom allows a direct comparison against the sensitivities of the Matrix R formulation in natural coordinates, already included in MBSLIM before the inception of this work. As a result of this comparison, a significant reduction in the computational cost is achieved with the same behavior in terms of accuracy.

## 5.3 Sensitivity analysis of semi-recursive penalty formulations

As it was commented in section 3.4, the penalty formulation is used exclusively for initialization purposes when ALI3-P formulations are intended to be used to obtain the dynamic response of a multibody system. Despite being circumscribed to the initial time, it has a direct effect into the dynamics as well as into the sensitivity analysis. Both the direct differentiation method and the adjoint variable method require the derivatives of the equations of motion at each time step of the simulation, and since the initial time is part of the simulation, the initial equations have to be differentiated and included into the sensitivity computations.

### 5.3.1 Forward sensitivity

The direct differentiation method was initially applied to the penalty formulation by Pagalday et al. in [86], and has been recently revisited in [2]. It should be reminded that this formulation eliminates the Lagrange multipliers from the dynamic equations and substitutes them by a penalty term involving the constraints vector and its time derivatives. If the objective function is dependent on these Lagrange multipliers, there is the need of approximating them and their sensitivities according to the approximation taken in the dynamics (3.44).

Let us recall the dynamic expressions of the penalty formulation revisited in section 3.4. Now, taking derivatives on (3.42) with respect to a set of parameters $\boldsymbol{\rho} \in \mathbb{R}^p$, the following set of $p$ systems of equations is obtained:

$$\check{\mathbf{M}}\ddot{\mathbf{z}}' = \check{\mathbf{Q}}_{\hat{\boldsymbol{\rho}}} - \check{\mathbf{M}}_{\hat{\boldsymbol{\rho}}}\ddot{\mathbf{z}} - \left(\check{\mathbf{K}} + \check{\mathbf{M}}_{\hat{\mathbf{z}}}^{d}\right)\mathbf{z}' - \check{\mathbf{C}}\dot{\mathbf{z}}' \tag{5.69}$$

with

$$\begin{aligned}\breve{\mathbf{K}} = -\breve{\mathbf{Q}}_{\hat{\mathbf{z}}} = \mathbf{K} + \mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}} \left(\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}\dot{\mathbf{z}} + \dot{\mathbf{\Phi}}_{t} + 2\Omega\xi\dot{\mathbf{\Phi}} + \Omega^2\mathbf{\Phi}\right) \\ +\mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \left(\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}\dot{\mathbf{z}} + \dot{\mathbf{\Phi}}_{t\hat{\mathbf{z}}} + 2\Omega\xi\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}} + \Omega^2\mathbf{\Phi}_{\hat{\mathbf{z}}}\right)\end{aligned} \tag{5.70a}$$

$$\begin{aligned}\breve{\mathbf{C}} = -\breve{\mathbf{Q}}_{\hat{\dot{\mathbf{z}}}} = \mathbf{C} + \mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \left(\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}\hat{\dot{\mathbf{z}}}}\dot{\mathbf{z}} + \dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}} + \dot{\mathbf{\Phi}}_{t\hat{\dot{\mathbf{z}}}} + 2\Omega\xi\dot{\mathbf{\Phi}}_{\hat{\dot{\mathbf{z}}}}\right) \\ = \mathbf{C} + \mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \left(2\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}} + 2\Omega\xi\dot{\mathbf{\Phi}}_{\hat{\dot{\mathbf{z}}}}\right)\end{aligned} \tag{5.70b}$$

$$\begin{aligned}\breve{\mathbf{Q}}_{\hat{\boldsymbol{\rho}}} = \mathbf{Q}_{\hat{\boldsymbol{\rho}}}^{d} - \mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}}\boldsymbol{\alpha} \left(\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}\dot{\mathbf{z}} + \dot{\mathbf{\Phi}}_{t} + 2\Omega\xi\dot{\mathbf{\Phi}} + \Omega^2\mathbf{\Phi}\right) \\ -\mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\alpha} \left(\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}\dot{\mathbf{z}} + \dot{\mathbf{\Phi}}_{t\hat{\mathbf{z}}} + 2\Omega\boldsymbol{\xi}\dot{\mathbf{\Phi}}_{\hat{\boldsymbol{\rho}}} + \Omega^2\mathbf{\Phi}_{\hat{\boldsymbol{\rho}}}\right)\end{aligned} \tag{5.70c}$$

$$\breve{\mathbf{M}}_{\hat{\mathbf{z}}} = \mathbf{M}_{\hat{\mathbf{z}}}^{d} + \mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\alpha}\mathbf{\Phi}_{\hat{\mathbf{z}}} + \mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\alpha}\mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}} \tag{5.70d}$$

$$\breve{\mathbf{M}}_{\hat{\boldsymbol{\rho}}} = \mathbf{M}_{\hat{\boldsymbol{\rho}}}^{d} + \mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}}\boldsymbol{\alpha}\mathbf{\Phi}_{\hat{\mathbf{z}}} + \mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\alpha}\mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}} \tag{5.70e}$$

Herein, $\mathbf{K}$ is the stiffness matrix (see section 4.2.2.1) and $\mathbf{C}$ the damping matrix of the mechanism (see section 4.2.2.2), while the subscripts indicate partial derivatives or the differentiation rule introduced in (4.9).

The absence of Lagrange multipliers in this formulation encompasses a series of additional considerations when the penalty formulation is used in the initialization of an ALI3-P scheme. The main issue is related to the dependency of the objective function considered on the Lagrange multipliers, which could appear if this function uses values of reaction forces or torques corresponding to the constraint equations. Besides, the computation of the reaction forces in joint coordinate models is now put aside since they are not the subject of the present development.

The dependency of the objective function on Lagrange multipliers imply the use of the derivative of the approximated Lagrange multipliers given by equation (3.44):

$$\boldsymbol{\lambda}^{*\prime} = \boldsymbol{\alpha}\left(\ddot{\mathbf{\Phi}}' + 2\Omega\xi\dot{\mathbf{\Phi}}' + \Omega^2\mathbf{\Phi}'\right) \tag{5.71}$$

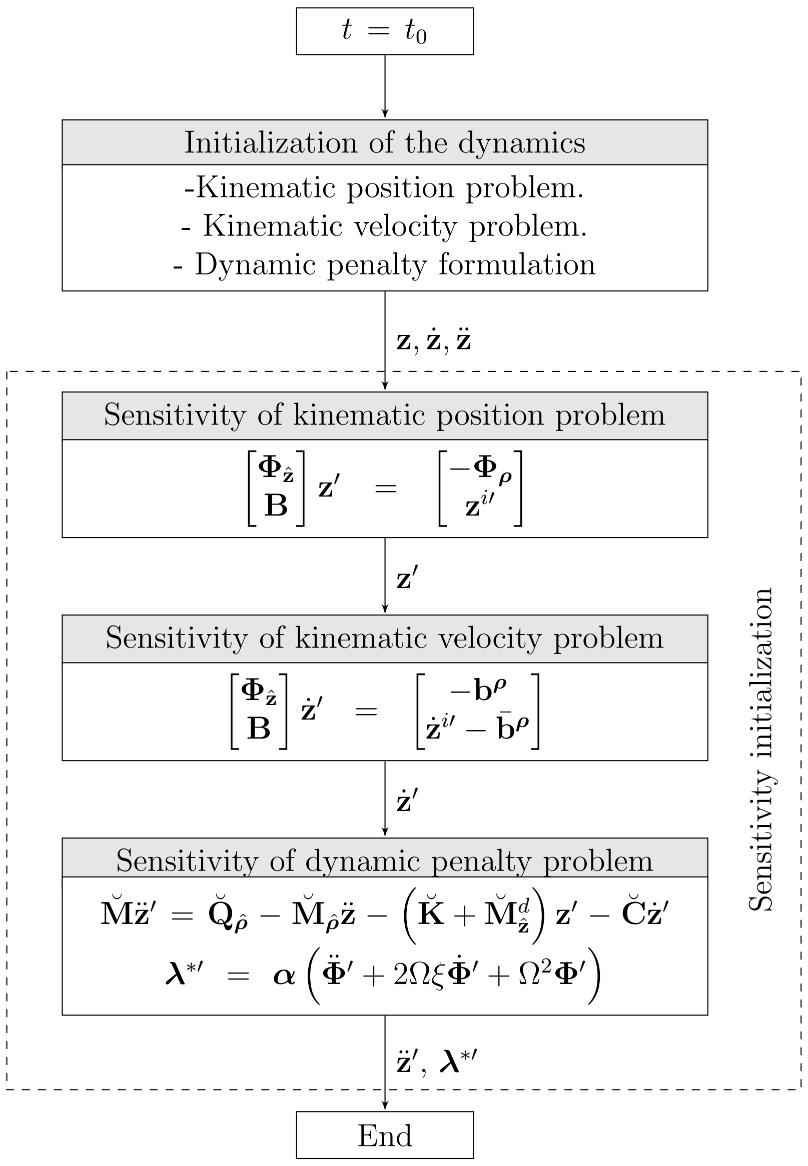
where:

$$\mathbf{\Phi}' = \mathbf{\Phi}_{\hat{\mathbf{z}}}\mathbf{z}' + \mathbf{\Phi}_{\hat{\boldsymbol{\rho}}} \tag{5.72a}$$

$$\dot{\mathbf{\Phi}}' = \dot{\mathbf{\Phi}}_{\hat{\dot{\mathbf{z}}}}\dot{\mathbf{z}}' + \dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}\mathbf{z}' + \dot{\mathbf{\Phi}}_{\hat{\boldsymbol{\rho}}} \tag{5.72b}$$

$$\ddot{\mathbf{\Phi}}' = \ddot{\mathbf{\Phi}}_{\hat{\ddot{\mathbf{z}}}}\ddot{\mathbf{z}}' + \ddot{\mathbf{\Phi}}_{\hat{\dot{\mathbf{z}}}}\dot{\mathbf{z}}' + \ddot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}\mathbf{z}' + \ddot{\mathbf{\Phi}}_{\hat{\boldsymbol{\rho}}} \tag{5.72c}$$

Therefore, since (5.69) is an ODE system, it could be integrated in time to obtain the sensitivities of the states, and then they can be used to compute the sensitivities of the Lagrange multipliers by means of (5.71). In the case of using this formulation to initialize an ALI3-P, the sensitivities of positions and velocities are already known, since they are obtained in the corresponding kinematic sensitivity problems, and therefore $\ddot{\mathbf{z}}'$ can be directly obtained without a numerical integrator. In Figure 5.1, the flow chart of the initialization process of the direct sensitivity of a semi-recursive ALI3-P formulation by means of the sensitivity analysis of the kinematic problems in positions and velocities and the dynamic penalty problem is displayed.



$t = t_0$
Initialization of the dynamics
-Kinematic position problem.
- Kinematic velocity problem.
- Dynamic penalty formulation
$\mathbf{z}, \dot{\mathbf{z}}, \ddot{\mathbf{z}}$
Sensitivity of kinematic position problem
$\begin{bmatrix} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \\ \mathbf{B} \end{bmatrix} \mathbf{z}' = \begin{bmatrix} -\boldsymbol{\Phi}_{\boldsymbol{\rho}} \\ \mathbf{z}^{i\prime} \end{bmatrix}$
$\mathbf{z}'$
Sensitivity of kinematic velocity problem
$\begin{bmatrix} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \\ \mathbf{B} \end{bmatrix} \dot{\mathbf{z}}' = \begin{bmatrix} -\mathbf{b}^{\boldsymbol{\rho}} \\ \dot{\mathbf{z}}^{i\prime} - \bar{\mathbf{b}}^{\boldsymbol{\rho}} \end{bmatrix}$
$\dot{\mathbf{z}}'$
Sensitivity of dynamic penalty problem
$\breve{\mathbf{M}}\ddot{\mathbf{z}}' = \breve{\mathbf{Q}}_{\hat{\boldsymbol{\rho}}} - \breve{\mathbf{M}}_{\hat{\boldsymbol{\rho}}}\ddot{\mathbf{z}} - \left(\breve{\mathbf{K}} + \breve{\mathbf{M}}^{d}_{\hat{\mathbf{z}}}\right)\mathbf{z}' - \breve{\mathbf{C}}\dot{\mathbf{z}}'$
$\boldsymbol{\lambda}^{*\prime} = \boldsymbol{\alpha}\left(\ddot{\boldsymbol{\Phi}}' + 2\Omega\xi\dot{\boldsymbol{\Phi}}' + \Omega^2\boldsymbol{\Phi}'\right)$
$\ddot{\mathbf{z}}', \boldsymbol{\lambda}^{*\prime}$
Sensitivity initialization
End

Figure 5.1: Flowchart of the initialization process for the sensitivity analysis of semi-recursive ALI3-P formulations.

### 5.3.2 Adjoint sensitivity

The particular initialization of the dynamic system in ALI3-P formulations entails a series of considerations in the application of the adjoint variable method to the dynamic equations. On the one hand, there are two possible approaches for the adjoint method, which are the discrete and continuous adjoint variable methods, consisting in the use of the discrete or the continuous equations of motion in the construction of the Lagrangian which originates the set of adjoint equations.

The continuous adjoint variable method applied to the dynamic equations of motion of a multibody system formulated as a second-order DAE system usually conveys a set of conditions at the initial instant of time involving the sensitivities of the states at position and velocity levels[3]. The sensitivities of the position and velocity problems at the initial instant of time can be straightforwardly calculated from the kinematic sensitivity analysis described in section 5.1. However, the effect of the dynamic acceleration problem is minimum in the continuous adjoint variable method, since it will only determine the value of one of the steps of the gradient numerical integration. In brief, for small enough time steps, the inclusion of the dynamic initial acceleration problem in the continuous adjoint variable method can be dismissed, hence it will not be described here. The reader is referred to [2] for a general description of the continuous adjoint variable method applied to a general penalty formulation.

On the contrary, if the sensitivity analysis of an ODE or a DAE system is addressed by means of the discrete adjoint variable method, every equation solved at each time instant has to be considered, including the kinematic position and velocity problems and the dynamic penalty problem. The main reason for these requirements relies on the type of derivatives considered. Since the equations of motion used in this approach contain a numerical integrator already applied to them before differentiation, they cannot be integrated by parts in time (they constitute algebraic equations), thus no instant terms at the initial time arise in this case. The effect of the initialization process in the discrete adjoint variable method has to be considered jointly with the rest of the equations solved in the simulation, hence the discrete adjoint equations of the penalty formulation will be particularized for a semi-recursive ALI3-P simulation in section 5.4.2.2.

## 5.4 Sensitivity analysis of semi-recursive ALI3-P formulations

The formulations based on dependent coordinates such as ALI3-P have larger numbers of variables than the ones based on independent coordinates. However, in general, they usually display better performance in terms of computational time, since no translation from dependent to independent coordinates is needed, as it happens, for instance, in Matrix R formulations.

Let us remind here that the dynamic ALI3-P formulation guarantees a good fulfillment of the constraints in positions but also in velocities and accelerations thanks to orthogonal projections. The results of the dynamics have high accuracy as well as a reduced computational time. Furthermore, the scheme of solution used, allowing the presence of redundant constraints, including the stiffness and damping matrices in the solution, and solving the index-3 DAE directly in positions avoiding the well known drift problem of reduced-index transformations, leads to a robust, accurate and fast solution of the dynamics, properties that can be extended to the sensitivity analysis.

[3]Proof of this can be found in the application of this method to the penalty formulation explored in [2] or to the ALI3-P formulation described in [4].

### 5.4.1 Forward sensitivity

In this section, a generalized method for the forward sensitivity analysis of the semi-recursive ALI3-P formulation is introduced.

Let us consider an objective function expressed in terms of $\mathbf{z}$, $\dot{\mathbf{z}}$, $\ddot{\mathbf{z}}$, $\boldsymbol{\lambda}^*$ and $\boldsymbol{\rho}$, and also in terms of the Lagrange multipliers of the velocity and acceleration projections $\boldsymbol{\sigma}$ and $\boldsymbol{\kappa}$ (in the case of projections solved with an augmented Lagrangian scheme):

$$\boldsymbol{\psi} = \int_{t_0}^{t_F} \mathbf{g}\left(\mathbf{z}, \dot{\mathbf{z}}, \ddot{\mathbf{z}}, \boldsymbol{\lambda}^*, \boldsymbol{\sigma}, \boldsymbol{\kappa}, \boldsymbol{\rho}\right) \mathrm{d}t \tag{5.73}$$

The gradient of the objective function can be expressed by the following equation:

$$\boldsymbol{\psi}' = \nabla \boldsymbol{\psi}^T = \int_{t_0}^{t_F} \left(\mathbf{g}_{\hat{\mathbf{z}}}\mathbf{z}' + \mathbf{g}_{\hat{\dot{\mathbf{z}}}}\dot{\mathbf{z}}' + \mathbf{g}_{\hat{\ddot{\mathbf{z}}}}\ddot{\mathbf{z}}' + \mathbf{g}_{\boldsymbol{\lambda}^*}\boldsymbol{\lambda}^{*\prime} + \mathbf{g}_{\boldsymbol{\sigma}}\boldsymbol{\sigma}' + \mathbf{g}_{\boldsymbol{\kappa}}\boldsymbol{\kappa}' + \mathbf{g}_{\hat{\boldsymbol{\rho}}}\right) \mathrm{d}t \tag{5.74}$$

In equation (5.74) the derivatives of $\mathbf{g}$ are known, and the terms $\mathbf{z}'$, $\dot{\mathbf{z}}'$, $\ddot{\mathbf{z}}'$, $\boldsymbol{\lambda}^{*\prime}$, $\boldsymbol{\sigma}'$ and $\boldsymbol{\kappa}'$ are the unknown matrices, which can be solved by means of: a set $p$ Differential Algebraic Equation systems for the sensitivity analysis of the dynamics; other $p$ systems of equations for the sensitivity of the velocity projections; and other $p$ systems of equations for the sensitivity of the acceleration projections.

Taking derivatives on the augmented Lagrangian index-3 part of the ALI3-P formulation given by equations (3.28) with respect to the set of parameters $\boldsymbol{\rho}$, the following $p$ systems of equations are obtained:

$$\left[\mathbf{M}^d \ddot{\mathbf{z}}^{*\prime\{i\}} + \mathbf{C}\dot{\mathbf{z}}^{*\prime\{i\}} + \bar{\mathbf{K}}\mathbf{z}'^{\{i\}} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \boldsymbol{\lambda}^{*\prime\{i\}}\right] = \bar{\mathbf{Q}}^{\boldsymbol{\rho}} \tag{5.75a}$$

$$\boldsymbol{\lambda}^{*\prime\{i\}} = \boldsymbol{\lambda}^{*\prime\{i-1\}} + \boldsymbol{\alpha}\boldsymbol{\Phi}' \tag{5.75b}$$

with:

$$\bar{\mathbf{K}} = \mathbf{M}_{\hat{\mathbf{z}}}^d \ddot{\mathbf{z}}^* + \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}}\left(\boldsymbol{\lambda}^* + \boldsymbol{\alpha}\boldsymbol{\Phi}\right) + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\alpha}\boldsymbol{\Phi}_{\hat{\mathbf{z}}} + \mathbf{K} \tag{5.76}$$

$$\bar{\mathbf{Q}}^{\boldsymbol{\rho}} = \mathbf{Q}_{\hat{\boldsymbol{\rho}}}^d - \mathbf{M}_{\hat{\boldsymbol{\rho}}}^d \ddot{\mathbf{z}}^* - \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}}\left(\boldsymbol{\lambda}^* + \boldsymbol{\alpha}\boldsymbol{\Phi}\right) - \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\alpha}\boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}} \tag{5.77}$$

$$\boldsymbol{\Phi}' = \boldsymbol{\Phi}_{\hat{\mathbf{z}}}\mathbf{z}' + \boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}} \tag{5.78}$$

in which $\bar{\mathbf{K}}$, $\mathbf{C}$ and $\mathbf{M}^d \in \mathbb{R}^{n\times n}$ are square matrices while $\bar{\mathbf{Q}}^{\boldsymbol{\rho}} \in \mathbb{R}^{n\times p}$ is a matrix with the same dimensions of $\mathbf{z}'^{\{i\}}$. These matrices involve the following tensor-vector products: $\mathbf{M}_{\hat{\mathbf{z}}}^d \ddot{\mathbf{z}}^* = \mathbf{M}_{\hat{\mathbf{z}}}^d \otimes \ddot{\mathbf{z}}^*$, $\boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}}\left(\boldsymbol{\lambda}^* + \boldsymbol{\alpha}\boldsymbol{\Phi}\right) = \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}} \otimes \left(\boldsymbol{\lambda}^* + \boldsymbol{\alpha}\boldsymbol{\Phi}\right)$, $\mathbf{M}_{\hat{\boldsymbol{\rho}}}^d \ddot{\mathbf{z}}^* = \mathbf{M}_{\hat{\boldsymbol{\rho}}}^d \otimes \ddot{\mathbf{z}}^*$ and $\boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}}\left(\boldsymbol{\lambda}^* + \boldsymbol{\alpha}\boldsymbol{\Phi}\right) = \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}} \otimes \left(\boldsymbol{\lambda}^* + \boldsymbol{\alpha}\boldsymbol{\Phi}\right)$.

The constraints errors at velocity and acceleration levels are controlled in semi-recursive ALI3-P formulations by means of velocity and acceleration projections. Considering $\bar{\mathbf{P}} \in \mathbb{R}^{n\times n}$ a symmetric semi-definite positive projection matrix, the sensitivity equations of the iterative velocity projections given by (3.35) take the form:

$$\begin{aligned}\left(\bar{\mathbf{P}} + \varsigma\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\alpha}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right)\dot{\mathbf{z}}'^{\{i\}} = {} & \bar{\mathbf{P}}\dot{\mathbf{z}}^{*\prime} + \bar{\mathbf{P}}'\left(\dot{\mathbf{z}}^* - \dot{\mathbf{z}}\right) - \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}}\left(\boldsymbol{\sigma} + \varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}}\right)\mathbf{z}' \\ & -\boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}}\left(\boldsymbol{\sigma} + \varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}}\right) - \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\left(\boldsymbol{\sigma}'^{\{i\}} + \varsigma\boldsymbol{\alpha}\mathbf{b}^{\boldsymbol{\rho}}\right)\end{aligned} \tag{5.79a}$$

$$\boldsymbol{\sigma}'^{\{i\}} = \boldsymbol{\sigma}'^{\{i-1\}} + \varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}}';\ i > 1 \tag{5.79b}$$

with $\mathbf{b}^{\boldsymbol{\rho}}$ defined in (5.5a), and:

$$\bar{\mathbf{P}}' = \bar{\mathbf{P}}_{\hat{\mathbf{z}}}\mathbf{z}' + \bar{\mathbf{P}}_{\hat{\dot{\mathbf{z}}}^*}\dot{\mathbf{z}}^{*\prime} + \bar{\mathbf{P}}_{\hat{\boldsymbol{\rho}}} \tag{5.80}$$

Analogously, taking derivatives on (3.39) with respect to the parameters of the system, the sensitivity of the acceleration projections is reached:

$$\begin{aligned}\left(\bar{\mathbf{P}} + \varsigma\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\alpha}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right)\ddot{\mathbf{z}}'^{\{i\}} = \bar{\mathbf{P}}\ddot{\mathbf{z}}^{*\prime} + \bar{\mathbf{P}}'\left(\ddot{\mathbf{z}}^* - \ddot{\mathbf{z}}\right) - \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}}\left(\boldsymbol{\kappa} + \varsigma\boldsymbol{\alpha}\ddot{\boldsymbol{\Phi}}\right)\mathbf{z}' \\ -\boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}}\left(\boldsymbol{\kappa} + \varsigma\boldsymbol{\alpha}\ddot{\boldsymbol{\Phi}}\right) - \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\left(\boldsymbol{\kappa}'^{\{i\}} + \varsigma\boldsymbol{\alpha}\mathbf{c}^{\boldsymbol{\rho}}\right)\end{aligned} \tag{5.81a}$$

$$\boldsymbol{\kappa}'^{\{i\}} = \boldsymbol{\kappa}'^{\{i-1\}} + \varsigma\boldsymbol{\alpha}\ddot{\boldsymbol{\Phi}}';\ i > 1 \tag{5.81b}$$

with $\mathbf{c}^{\boldsymbol{\rho}}$ defined in (5.5c).

Equations (5.75) combined with a numerical integrator provide the sensitivities of the relative coordinates in positions $\mathbf{z}'$, the sensitivities with respect to the Lagrange multipliers $\boldsymbol{\lambda}^{*\prime}$ and the sensitivities of the unprojected velocities and accelerations of the states $\dot{\mathbf{z}}^{*\prime}$ and $\ddot{\mathbf{z}}^{*\prime}$. Equation (5.79) uses the unprojected sensitivities of the velocities obtained in (5.75) $\dot{\mathbf{z}}^{*\prime}$ to compute the sensitivities of the projected velocities of the states $\dot{\mathbf{z}}'$. Similarly, equation (5.81) produces the sensitivities of the projected accelerations $\ddot{\mathbf{z}}'$ from the unprojected ones $\ddot{\mathbf{z}}^{*\prime}$.

The sensitivity equations (5.79) and (5.81) have been obtained differentiating the velocity and acceleration projections in the form of an Augmented Lagrangian scheme, but it has been proved empirically that penalty non-iterative schemes of solution (3.37) and (3.40) offer good results as well. In that case, the Lagrange multipliers $\boldsymbol{\sigma}$ and $\boldsymbol{\kappa}$ vanish from the projection sensitivities equations along with their sensitivities $\boldsymbol{\sigma}'$ and $\boldsymbol{\kappa}'$, keeping the remaining terms unchanged.

It is important to remark that the direct differentiation method is based on the forward differentiation of the expressions used to solve the dynamics. Despite that this affirmation seems to be trivial, it becomes of real importance when projections are treated. For instance, if in the computations of the dynamics the unprojected velocities generate an error in the velocity constraints lower than the specified tolerance, the projections do not need to be computed. Accordingly, if the velocity projections are not computed, the sensitivity of the projected velocities has to be omitted too. Otherwise, the final results will be inaccurate.

The ALI3-P formalism combined with a semi-recursive accumulation in joint coordinate models delivers efficient and accurate results for the dynamic simulation of multibody systems. The sensitivity analysis takes advantage of the reduced number of coordinates and constraints generated by joint coordinate models (compared with other coordinate models), leading to fast and accurate sensitivity evaluations, which convert the direct sensitivity analysis of semi-recursive ALI3-P formulations into one of the best options for an efficient and accurate sensitivity analysis.

The sensitivity analysis of closed-loop systems based on the direct differentiation method applied to the semi-recursive ALI3-P formulation has been programmed in the MBSLIM multibody library, including both the RTdyn0 and RTdyn1 approaches and the possibility of iterative or non-iterative projections. The projection matrix implemented is the mass matrix, primarily because both the mass matrix and its derivative are required in the index-3 augmented Lagrangian part of the ALI3-P formulation and in its sensitivity, respectively, hence both terms can be directly reused.

The semi-recursive ALI3-P direct sensitivity formulation presented in this section has usually displayed higher efficiency than the equivalent method in natural coordinates or the semi-recursive Matrix R sensitivity formulation in the numerical experiments in which it has been tested.

### 5.4.2 Adjoint sensitivity

The adjoint sensitivity of any set of ODE or DAE can be computed from two different perspectives, leading to the commonly known as discrete and continuous adjoint methods. The sensitivity of the DAE system originated from the dynamics of a multibody system formulated with an ALI3-P approach can be solved by means of these two approaches, encompassing different advantages and disadvantages related to the terms needed and the generality of the expressions.

On the continuous approach, the equations are considered as continuous in time, which entails a series of continuous differentiation formalisms that can induce complex and high time consuming derivatives when applied to second-order DAE systems. Nevertheless, the resulting expressions are general regardless of the numerical integrator used in the solution of the original ODE or DAE system.

The discrete approach, on the other hand, handles the derivatives of the discretized dynamic expressions with a numerical integrator applied to the states. The adjoint sensitivity system obtained in this approach depends on the numerical integrator selected on the dynamics, which means that the set of equations generated is particular for a family of integrators, and also forces to solve the dynamics and the sensitivities with the same integrator. Despite the particularity, the discrete method usually delivers simpler systems of equations and does not require additional time derivatives of dynamic magnitudes, as it could happen in continuous methods.

#### 5.4.2.1 Continuous approach

The dynamic equations of the semi-recursive ALI3-P formulation constitute a set of continuous-in-time equations which need to be evaluated at successive instants of time to be solved, since it is impossible or very difficult to solve a general DAE system without a discretization. From this perspective, the application of the continuous adjoint variable method (CAVM) seems to be the most logical option regarding the nature of the dynamic equations.

The continuous adjoint sensitivity of the Augmented Lagrangian Index-3 formulation with projections was developed and exhaustively detailed in [4]. In this section, the scheme of solution and expressions presented in this paper are revisited and extended for their application to relative coordinate models. Following the directives explained in [4], a change of variables, $\dot{\mathbf{z}}^* = \mathbf{v}^*$ is applied in order to avoid high order temporal derivatives, thus an additional constraint is added to the adjoint Lagrangian. Moreover, the addition to the Lagrangian of the velocity constraint equations in the final time is imposed to solve the incompatibilities appearing at time $t_F$, $\dot{\boldsymbol{\Phi}}\left(t_F, \mathbf{q}_F, \mathbf{v}_F^*, \boldsymbol{\rho}\right) \approx \mathbf{0}$. Observe that this constraint is evaluated with the unprojected velocities, and therefore it is not exactly equal to zero, but since the constraints in positions, velocities and accelerations in the previous time step are equal to zero thanks to the projections, this error could be considered low enough to neglect it.

Furthermore, the iterations of the Lagrange multipliers related to the Augmented Lagrangian scheme can be avoided using one of the lemmas introduced in the commented paper, which declares: "The augmented Lagrangian scheme of the index-3 DAE TLM is equivalent to the TLM of the augmented Lagrangian Index-3 formulation", [4], with the following approximation of the Lagrange multipliers:

$$\boldsymbol{\lambda} \approx \left(\boldsymbol{\lambda}^* + \boldsymbol{\alpha}\boldsymbol{\Phi}\right) \tag{5.82}$$

Consequently, the augmented Lagrangian index-3 can be substituted by an index-3 TLM, leading to a much direct solution avoiding the iteration of the Lagrange multipliers sensitivities.

Now, let us consider a Lagrangian composed of an objective function minus the set of equations that are fulfilled in the dynamics, and with the considerations of the change of variables and velocity constraint in the final time already commented:

$$\begin{aligned} \mathcal{L} = \boldsymbol{\psi} - \int_{t_0}^{t_F} \boldsymbol{\mu}_2^{\mathrm{T}} \left(\mathbf{M}\dot{\mathbf{v}}^* + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\left(\boldsymbol{\lambda}^* + \boldsymbol{\alpha}\boldsymbol{\Phi}\right) - \mathbf{Q}\right) \mathrm{d}t \\ - \int_{t_0}^{t_F} \boldsymbol{\mu}_1^{\mathrm{T}} \left(\dot{\mathbf{z}}^* - \mathbf{v}^*\right) \mathrm{d}t - \int_{t_0}^{t_F} \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}} \boldsymbol{\Phi} \mathrm{d}t \\ - \int_{t_0}^{t_F} \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left(\left[\bar{\mathbf{P}} + \varsigma \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \boldsymbol{\alpha} \boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right] \dot{\mathbf{z}} - \bar{\mathbf{P}}\mathbf{v}^* + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \boldsymbol{\Phi}_t\right) \mathrm{d}t \\ - \int_{t_0}^{t_F} \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left(\left[\bar{\mathbf{P}} + \varsigma \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \boldsymbol{\alpha} \boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right] \ddot{\mathbf{z}} - \bar{\mathbf{P}}\dot{\mathbf{v}}^* + \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \varsigma \boldsymbol{\alpha} \left(\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\dot{\mathbf{z}} + \dot{\boldsymbol{\Phi}}_t\right)\right) \mathrm{d}t \\ + \boldsymbol{\eta}^{\mathrm{T}} \dot{\boldsymbol{\Phi}}\left(t_F, \mathbf{q}_F, \mathbf{v}_F^*, \boldsymbol{\rho}\right) \end{aligned} \tag{5.83}$$

For the sake of clearness, the adjoint variables $\boldsymbol{\mu}_1$, $\boldsymbol{\mu}_2$, $\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}$, $\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}} \in \mathbb{R}^{n \times o}$ and $\boldsymbol{\eta}$, $\boldsymbol{\mu}_{\boldsymbol{\Phi}} \in \mathbb{R}^{m \times o}$ are identified by the same subindices introduced in [4]. The resulting Lagrangian has 6 matrices of adjoint variables, $\boldsymbol{\mu}_2$ and $\boldsymbol{\mu}_{\boldsymbol{\Phi}}$ corresponding to the index-3 DAE system, $\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}$ to the projections of velocities, $\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}$ to acceleration projections, $\boldsymbol{\mu}_1$ to the change of variable equations and $\boldsymbol{\eta}$ to the velocity constraint for unprojected velocities (this last only evaluated at the final time).

The gradient of the Lagrangian (5.83) with respect to the set of parameters $\boldsymbol{\rho}$ can be formulated as:

$$
\begin{aligned}
\mathcal{L}' = \left[\boldsymbol{\eta}^{\mathrm{T}}\dot{\boldsymbol{\Phi}}^{*}_{\hat{\mathbf{z}}}\mathbf{z}' + \boldsymbol{\eta}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\mathbf{v}^{*\prime} + \boldsymbol{\eta}^{\mathrm{T}}\dot{\boldsymbol{\Phi}}^{*}_{\boldsymbol{\rho}}\right]^{t_F}& \\
+ \int_{t_0}^{t_F} \Big(\mathbf{g}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}_2^{\mathrm{T}}\bar{\mathbf{K}} - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}}\Big(\boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}} - \bar{\mathbf{P}}_{\hat{\mathbf{z}}}\left(\mathbf{v}^{*} - \dot{\mathbf{z}}\right) + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\Big)& \\
-\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\Big(\boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\ddot{\boldsymbol{\Phi}} - \bar{\mathbf{P}}_{\hat{\mathbf{z}}}\left(\dot{\mathbf{v}}^{*} - \ddot{\mathbf{z}}\right) + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\Big)\Big)\,\mathbf{z}'\mathrm{d}t& \\
+ \int_{t_0}^{t_F} \Big(\mathbf{g}_{\hat{\dot{\mathbf{z}}}} - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}}\left(\bar{\mathbf{P}} + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right) - 2\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\Big)\,\dot{\mathbf{z}}'\mathrm{d}t - \int_{t_0}^{t_F}\boldsymbol{\mu}_1^{\mathrm{T}}\dot{\mathbf{z}}^{*\prime}\mathrm{d}t& \\
+ \int_{t_0}^{t_F} \left(\boldsymbol{\mu}_1^{\mathrm{T}} - \boldsymbol{\mu}_2^{\mathrm{T}}\mathbf{C} + \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}}\left(\bar{\mathbf{P}} + \bar{\mathbf{P}}_{\mathbf{v}^{*}}\left(\mathbf{v}^{*} - \dot{\mathbf{z}}\right)\right) + \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\bar{\mathbf{P}}_{\mathbf{v}^{*}}\left(\dot{\mathbf{v}}^{*} - \ddot{\mathbf{z}}\right)\right)\mathbf{v}^{*\prime}\mathrm{d}t& \qquad (5.84)\\
- \int_{t_0}^{t_F} \left(\mathbf{g}_{\hat{\ddot{\mathbf{z}}}} - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\left(\bar{\mathbf{P}} + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right)\right)\ddot{\mathbf{z}}'\mathrm{d}t& \\
- \int_{t_0}^{t_F} \left(\boldsymbol{\mu}_2^{\mathrm{T}}\mathbf{M} - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\bar{\mathbf{P}}\right)\dot{\mathbf{v}}^{*\prime}\mathrm{d}t + \int_{t_0}^{t_F} \left(\mathbf{g}_{\boldsymbol{\lambda}^{*}} - \boldsymbol{\mu}_2^{\mathrm{T}}\boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\right)\boldsymbol{\lambda}^{*\prime}\mathrm{d}t& \\
+ \int_{t_0}^{t_F} \Big(\mathbf{g}_{\hat{\boldsymbol{\rho}}} + \boldsymbol{\mu}_2^{\mathrm{T}}\bar{\mathbf{Q}}^{\boldsymbol{\rho}} - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}} - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}}\Big(\boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}\varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}} - \bar{\mathbf{P}}_{\hat{\boldsymbol{\rho}}}\left(\mathbf{v}^{*} - \dot{\mathbf{z}}\right) + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}}\Big)& \\
-\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\Big(\boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}\varsigma\boldsymbol{\alpha}\ddot{\boldsymbol{\Phi}} - \bar{\mathbf{P}}_{\hat{\boldsymbol{\rho}}}\left(\dot{\mathbf{v}}^{*} - \ddot{\mathbf{z}}\right) + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\ddot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}}\Big)\Big)\,\mathrm{d}t&
\end{aligned}
$$

where the asterisk as superscript means that the term is evaluated with the unprojected velocities or accelerations. Observe that there are 7 unknown sensitivity arrays $\mathbf{z}'$, $\dot{\mathbf{z}}'$, $\mathbf{z}^{*\prime}$, $\ddot{\mathbf{z}}'$, $\mathbf{v}^{*\prime}$, $\dot{\mathbf{v}}^{*\prime}$ and $\boldsymbol{\lambda}^{*\prime}$ which represent $(6n + m) \times p$ sensitivities corresponding to the solution of the direct sensitivity problem. Since the previous expressions are valid for any value of the adjoint variables, they can be selected in order to nullify the term multiplying the forward sensitivity variables. Thanks to this selection, the adjoint variable method dodges the calculation of these variables which can be highly time consuming for a large number of parameters.

Looking into equation (5.84), it can be observed that there are $2n \times p$ forward sensitivities that cannot be neglected, so an integration by parts in time may be applied to the integrals involving $\dot{\mathbf{z}}^{*\prime}$ and $\dot{\mathbf{v}}^{*\prime}$, which constitute the temporal derivatives of $\mathbf{z}^*$ and $\dot{\mathbf{v}}^*$ respectively:

$$\begin{aligned}
\mathcal{L}' = {} & \left[\boldsymbol{\eta}^{\mathrm{T}}\dot{\boldsymbol{\Phi}}^*_{\hat{\mathbf{z}}}\mathbf{z}' + \boldsymbol{\eta}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\mathbf{v}^{*\prime} + \boldsymbol{\eta}^{\mathrm{T}}\dot{\boldsymbol{\Phi}}^*_{\boldsymbol{\rho}}\right]^{t_F} - \left[\boldsymbol{\mu}_1^{\mathrm{T}}\mathbf{z}'\right]_{t_0}^{t_F} - \left[\left(\boldsymbol{\mu}_2^{\mathrm{T}}\mathbf{M} - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\bar{\mathbf{P}}\right)\mathbf{v}^{*\prime}\right]_{t_0}^{t_F} \\
& + \int_{t_0}^{t_F} \left(\mathbf{g}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}_2^{\mathrm{T}}\bar{\mathbf{K}} - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}}\left(\boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}} - \bar{\mathbf{P}}_{\hat{\mathbf{z}}}\left(\mathbf{v}^* - \dot{\mathbf{z}}\right) + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\right)\right. \\
& \qquad \left. - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\left(\boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\ddot{\boldsymbol{\Phi}} - \bar{\mathbf{P}}_{\hat{\mathbf{z}}}\left(\dot{\mathbf{v}}^* - \ddot{\mathbf{z}}\right) + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\right) + \dot{\boldsymbol{\mu}}_1^{\mathrm{T}}\right)\mathbf{z}'\mathrm{d}t \\
& + \int_{t_0}^{t_F} \left(\mathbf{g}_{\dot{\hat{\mathbf{z}}}} - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}}\left(\bar{\mathbf{P}} + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right) - 2\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\right)\dot{\mathbf{z}}'\mathrm{d}t \\
& + \int_{t_0}^{t_F} \left(\boldsymbol{\mu}_1^{\mathrm{T}} - \boldsymbol{\mu}_2^{\mathrm{T}}\mathbf{C} + \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}}\left(\bar{\mathbf{P}} + \bar{\mathbf{P}}_{\mathbf{v}^*}\left(\mathbf{v}^* - \dot{\mathbf{z}}\right)\right) + \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\bar{\mathbf{P}}_{\mathbf{v}^*}\left(\dot{\mathbf{v}}^* - \ddot{\mathbf{z}}\right)\right. \\
& \qquad \left. + \dot{\boldsymbol{\mu}}_2^{\mathrm{T}}\mathbf{M} + \boldsymbol{\mu}_2^{\mathrm{T}}\dot{\mathbf{M}} - \dot{\boldsymbol{\mu}}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\bar{\mathbf{P}} - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\dot{\bar{\mathbf{P}}}\right)\mathbf{v}^{*\prime}\mathrm{d}t \\
& - \int_{t_0}^{t_F} \left(\mathbf{g}_{\ddot{\hat{\mathbf{z}}}} - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\left(\bar{\mathbf{P}} + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right)\right)\ddot{\mathbf{z}}'\mathrm{d}t + \int_{t_0}^{t_F} \left(\mathbf{g}_{\boldsymbol{\lambda}^*} - \boldsymbol{\mu}_2^{\mathrm{T}}\boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\right)\boldsymbol{\lambda}^{*\prime}\mathrm{d}t \\
& + \int_{t_0}^{t_F} \left(\mathbf{g}_{\hat{\boldsymbol{\rho}}} + \boldsymbol{\mu}_2^{\mathrm{T}}\bar{\mathbf{Q}}^{\boldsymbol{\rho}} - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}} - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}}\left(\boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}\varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}} - \bar{\mathbf{P}}_{\hat{\boldsymbol{\rho}}}\left(\mathbf{v}^* - \dot{\mathbf{z}}\right) + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}}\right)\right. \\
& \qquad \left. - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\left(\boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}\varsigma\boldsymbol{\alpha}\ddot{\boldsymbol{\Phi}} - \bar{\mathbf{P}}_{\hat{\boldsymbol{\rho}}}\left(\dot{\mathbf{v}}^* - \ddot{\mathbf{z}}\right) + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\ddot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}}\right)\right)\mathrm{d}t
\end{aligned} \tag{5.85}$$

Once integrated by parts, the unknown direct sensitivities can be neglected making the $o\times(4n+m)$ coefficients multiplying them equal to zero. Therefore, the final system of equations depending exclusively on the adjoint variables is achieved:

$$\dot{\boldsymbol{\mu}}_1 - \bar{\mathbf{K}}^{\mathrm{T}}\boldsymbol{\mu}_2 - \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\boldsymbol{\mu}_{\boldsymbol{\Phi}} - \mathbf{A}^{\mathrm{T}}\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}} - \mathbf{B}^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}} = -\mathbf{g}^{\mathrm{T}}_{\hat{\mathbf{z}}}, \tag{5.86a}$$

$$\mathbf{M}^{\mathrm{T}}\dot{\boldsymbol{\mu}}_2 - \bar{\mathbf{P}}^{\mathrm{T}}\dot{\boldsymbol{\mu}}_{\ddot{\boldsymbol{\Phi}}} - \bar{\mathbf{C}}^{\mathrm{T}}\boldsymbol{\mu}_2 + \boldsymbol{\mu}_1 + \mathbf{E}^{\mathrm{T}}\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}} + \mathbf{F}^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}} = \mathbf{0}, \tag{5.86b}$$

$$\left(\bar{\mathbf{P}} + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right)^{\mathrm{T}}\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}} + 2\varsigma\left(\boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\right)^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}} = \mathbf{g}^{\mathrm{T}}_{\dot{\hat{\mathbf{z}}}}, \tag{5.86c}$$

$$\left(\bar{\mathbf{P}} + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right)^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}} = \mathbf{g}^{\mathrm{T}}_{\ddot{\hat{\mathbf{z}}}}, \tag{5.86d}$$

$$\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\boldsymbol{\mu}_2 = \mathbf{g}^{\mathrm{T}}_{\boldsymbol{\lambda}}. \tag{5.86e}$$

wherein:

$$\mathbf{A} = \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}} - \bar{\mathbf{P}}_{\hat{\mathbf{z}}}\left(\mathbf{v}^* - \dot{\mathbf{z}}\right) + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}, \tag{5.87a}$$

$$\mathbf{B} = \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\ddot{\boldsymbol{\Phi}} - \bar{\mathbf{P}}_{\hat{\mathbf{z}}}\left(\dot{\mathbf{v}}^* - \ddot{\mathbf{z}}\right) + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}, \tag{5.87b}$$

$$\bar{\mathbf{C}} = \mathbf{C} - \dot{\mathbf{M}}, \tag{5.87c}$$

$$\mathbf{E} = \bar{\mathbf{P}} + \bar{\mathbf{P}}_{\mathbf{v}^*}\left(\mathbf{v}^* - \dot{\mathbf{z}}\right), \tag{5.87d}$$

$$\mathbf{F} = -\dot{\bar{\mathbf{P}}} + \bar{\mathbf{P}}_{\mathbf{v}^*}\left(\dot{\mathbf{v}}^* - \ddot{\mathbf{z}}\right), \tag{5.87e}$$

$$\bar{\mathbf{K}} = \mathbf{K} + \mathbf{M}_{\hat{\mathbf{z}}}\dot{\mathbf{v}}^* + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\boldsymbol{\alpha}\boldsymbol{\Phi}_{\hat{\mathbf{z}}} + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}\left(\boldsymbol{\lambda}^* + \boldsymbol{\alpha}\boldsymbol{\Phi}\right). \tag{5.87f}$$

The resulting adjoint system constitutes an index-3 DAE system, where the algebraic equations (5.86c) and (5.86d) related to the projections in velocities and accelerations can be solved separately from the DAE system ((5.86a), (5.86b) and (5.86e)), as it happens with the underlying forward dynamics.

The adjoint system has to fulfill the following set of initialization conditions evaluated at $t_F$:

$$\left[\boldsymbol{\mu}_1\right]^{t_F} = \left[\dot{\boldsymbol{\Phi}}^{*\mathrm{T}}_{\hat{\mathbf{z}}}\boldsymbol{\eta}\right]^{t_F}, \tag{5.88a}$$

$$\left[\mathbf{M}^{\mathrm{T}}\boldsymbol{\mu}_2 - \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\boldsymbol{\eta}\right]^{t_F} = \left[\bar{\mathbf{P}}^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}\right]^{t_F}. \tag{5.88b}$$

The imposition of these initialization conditions in the final time $t_F$ obligates to integrate backward in time equations (5.86). This backward integration implies the storage of Lagrange multipliers, positions and projected and unprojected velocities and accelerations of the states at each time instant, thus once the forward dynamics is complete, the adjoint variables and the gradient of the objective function can be computed from $t_F$ to $t_0$. The terms required to build the adjoint system can be computed during the backwards integration using the stored values of the states and the Lagrange multipliers. A different possibility, not explored in this work, consists in the computation of all the terms required for the adjoint at each time step during the forward dynamics, but this will result in the usage of a big amount of memory whose access for reading and writing could worsen the computational time of the adjoint calculation.

Finally, the properties of the Lagrangian allow to calculate the gradient of the objective function with the remaining terms of (5.85), including solely partial derivatives of dynamic terms with respect to the array of parameters $\boldsymbol{\rho}$:

$$\begin{aligned} \psi' = \left[\boldsymbol{\eta}^{\mathrm{T}}\dot{\boldsymbol{\Phi}}^{*}_{\hat{\boldsymbol{\rho}}}\right]^{t_F} + \left[\boldsymbol{\mu}_1^{\mathrm{T}}\mathbf{z}' - \left(\boldsymbol{\mu}_2^{\mathrm{T}}\mathbf{M} - \boldsymbol{\mu}^{\mathrm{T}}_{\ddot{\boldsymbol{\Phi}}}\bar{\mathbf{P}}\right)\mathbf{v}^{*\prime}\right]_{t_0}^{t_F} \\ + \int_{t_0}^{t_F} \Big(\mathbf{g}_{\hat{\boldsymbol{\rho}}} + \boldsymbol{\mu}_2^{\mathrm{T}}\bar{\mathbf{Q}}^{\boldsymbol{\rho}} - \boldsymbol{\mu}^{\mathrm{T}}_{\boldsymbol{\Phi}}\boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}} - \boldsymbol{\mu}^{\mathrm{T}}_{\dot{\boldsymbol{\Phi}}}\left(\boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}\varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}} - \bar{\mathbf{P}}_{\hat{\boldsymbol{\rho}}}\left(\mathbf{v}^* - \dot{\mathbf{z}}\right) + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}}\right) \\ -\boldsymbol{\mu}^{\mathrm{T}}_{\ddot{\boldsymbol{\Phi}}}\left(\boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}\varsigma\boldsymbol{\alpha}\ddot{\boldsymbol{\Phi}} - \bar{\mathbf{P}}_{\hat{\boldsymbol{\rho}}}\left(\dot{\mathbf{v}}^* - \ddot{\mathbf{z}}\right) + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\ddot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}}\right)\Big)\,\mathrm{d}t \end{aligned} \tag{5.89}$$

The initialization of a DAE system is usually one of the most complex parts of its solution, specially harder the higher its index. In semi-recursive ALI3-P dynamics, for example, if a Newmark integrator is used, the positions, velocities and accelerations of the states are needed for the initialization of the integrator, and they can be easily calculated solving the kinematic problems in positions and velocities and using a reduced-index formulation to solve the initial dynamic accelerations (such as the penalty problem described in section 3.4). Observe that the structure of this initialization is completely different to the calculation of the dynamics at any other time.

Similarly, the solution of the initialization conditions of the adjoint system has a specific scheme of calculation based on the fulfillment of different conditions at the final time, both kinematic and dynamic. In [4], a set of equations and an algorithm for the solution of the initial conditions for an ALI3-P adjoint system were proposed and described in detail. Since the gist and the basis of the initialization have been already presented in the commented paper, here the scheme of solution is directly applied to a semi-recursive accumulation of a joint coordinate model.

At time $t_F$, the adjoint system is composed of equations (5.86) plus the instant conditions at time $t_F$ (5.88):

$$\dot{\boldsymbol{\mu}}_1 - \bar{\mathbf{K}}^{\mathrm{T}}\boldsymbol{\mu}_2 - \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\mu}_{\boldsymbol{\Phi}} - \mathbf{A}^{\mathrm{T}}\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}} - \mathbf{B}^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}} = -\mathbf{g}_{\hat{\mathbf{z}}}^{\mathrm{T}}, \tag{5.90a}$$

$$\mathbf{M}^{\mathrm{T}}\dot{\boldsymbol{\mu}}_2 - \bar{\mathbf{P}}^{\mathrm{T}}\dot{\boldsymbol{\mu}}_{\ddot{\boldsymbol{\Phi}}} - \bar{\mathbf{C}}^{\mathrm{T}}\boldsymbol{\mu}_2 + \boldsymbol{\mu}_1 + \mathbf{E}^{\mathrm{T}}\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}} + \mathbf{F}^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}} = \mathbf{0}, \tag{5.90b}$$

$$\left(\bar{\mathbf{P}} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\varsigma\boldsymbol{\alpha}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right)^{\mathrm{T}}\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}} + 2\varsigma\left(\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\right)^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}} = \mathbf{g}_{\dot{\hat{\mathbf{z}}}}^{\mathrm{T}}, \tag{5.90c}$$

$$\left(\bar{\mathbf{P}} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\varsigma\boldsymbol{\alpha}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right)^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}} = \mathbf{g}_{\ddot{\hat{\mathbf{z}}}}^{\mathrm{T}}, \tag{5.90d}$$

$$\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\boldsymbol{\mu}_2 = \mathbf{g}_{\boldsymbol{\lambda}^*}^{\mathrm{T}}, \tag{5.90e}$$

$$\left[\boldsymbol{\mu}_1\right]^{t_F} = \left[\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}^{*\mathrm{T}}\boldsymbol{\eta}\right]^{t_F}, \tag{5.90f}$$

$$\left[\mathbf{M}^{\mathrm{T}}\boldsymbol{\mu}_2 - \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\eta}\right]^{t_F} = \left[\bar{\mathbf{P}}^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}\right]^{t_F} \tag{5.90g}$$

Comparing the resulting initialization equations to the ones included in the reference work [4], it can be observed that, since the objective function considered in the present work only involves an integral term, no incompatible conditions are present here, like it occurs in the commented work.

The solution of the initial system (5.90) can be achieved with the following algorithm:

1. Calculation of $\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}$ from (5.90d).

2. Evaluation of $\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}$ from (5.90c).

3. Time differentiation of (5.90d) to obtain $\dot{\boldsymbol{\mu}}_{\ddot{\boldsymbol{\Phi}}}$. The time derivative of the adjoint variable related to the acceleration projections is required for solving the initial system of equations and to initialize its time integration. The most direct method to obtain it, consists in taking derivatives in (5.90d) with respect to time, which yields:

   $$\left(\bar{\mathbf{P}} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\varsigma\boldsymbol{\alpha}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right)^{\mathrm{T}}\dot{\boldsymbol{\mu}}_{\ddot{\boldsymbol{\Phi}}} = \dot{\mathbf{g}}_{\ddot{\hat{\mathbf{z}}}}^{\mathrm{T}} - \left(\dot{\bar{\mathbf{P}}} + \dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}^{\mathrm{T}}\varsigma\boldsymbol{\alpha}\boldsymbol{\Phi}_{\hat{\mathbf{z}}} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\right)^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}} \tag{5.91}$$

   Observe that if $\mathbf{g}$ is at least quadratic in accelerations, the term $\dot{\mathbf{g}}_{\ddot{\hat{\mathbf{z}}}}$ can involve jerks $\dddot{\mathbf{z}}$, which could be very difficult to obtain analytically from the semi-recursive ALI3-P equations. The solution given in the reference paper and considered here is the use of a numerical estimation based on the values of the accelerations in the present and previous time steps.

4. Solution of equations (5.90e) and (5.90g) to obtain $\boldsymbol{\mu}_2$ and $\boldsymbol{\eta}$. These two sets of equations have the same structure as a classical index-1 DAE Lagrange system, and accordingly, they have an unique solution if the rank of $\left[\mathbf{M}^{d\mathrm{T}} \;\; \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\right]$ is equal to the number of coordinates and if no redundant constraints are present. However, although the first condition is satisfied in ALI3-P formulations, the use of redundant constraints leads to an infinite number of possible solutions of $\boldsymbol{\eta}$, so the problem is compatible undetermined. One of its possible solutions

consists in the minimum norm solution, which can be obtained by means of the Moore-Penrose generalized inverse. This method uses the singular value decomposition of the leading matrix, and despite the high computational effort involved in these type of calculations, the fact that it is evaluated only once per simulation causes that it does not have a significant impact in the general computation time.

The solution of $\boldsymbol{\mu}_2$ and $\boldsymbol{\eta}$ takes the form:

$$\begin{bmatrix} \boldsymbol{\mu}_2 \\ -\boldsymbol{\eta} \end{bmatrix} = \mathbf{V}_r \boldsymbol{\Sigma}_r^{-1} \mathbf{U}_r^{\mathrm{T}} \begin{bmatrix} \bar{\mathbf{P}}^{\mathrm{T}} \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}} \\ \mathbf{g}_{\boldsymbol{\lambda}}^{\mathrm{T}} \end{bmatrix} \tag{5.92}$$

with,

$$\boldsymbol{\Sigma} = \mathbf{U}^{\mathrm{T}} \begin{bmatrix} \mathbf{M}^{d\mathrm{T}} & \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \\ \boldsymbol{\Phi}_{\hat{\mathbf{z}}} & \mathbf{0} \end{bmatrix} \mathbf{V} \tag{5.93}$$

where $\boldsymbol{\Sigma} \in \mathbb{R}^{(n+m)\times(n+m)}$, $\mathbf{U} \in \mathbb{R}^{(n+m)\times(n+m)}$ and $\mathbf{V} \in \mathbb{R}^{(n+m)\times(n+m)}$ are the resulting matrices of the singular value decomposition (observe that $\boldsymbol{\Sigma}$ is diagonal). In (5.92), matrix $\boldsymbol{\Sigma}_r \in \mathbb{R}^{r\times r}$ is a diagonal matrix containing the non-zero singular values from $\boldsymbol{\Sigma}$, while $\mathbf{U}_r \in \mathbb{R}^{(n+m)\times r}$ and $\mathbf{V}_r \in \mathbb{R}^{(n+m)\times r}$ contain, respectively, the right and left singular vectors associated to them.

5. Assessment of $\boldsymbol{\mu}_1$ from (5.90f).

6. Calculation of $\dot{\boldsymbol{\mu}}_1$ from (5.90a) assuming $\boldsymbol{\mu}_{\boldsymbol{\Phi}} = \mathbf{0}$.

7. Computation of $\dot{\boldsymbol{\mu}}_2$. If the mass matrix has full rank, $\dot{\boldsymbol{\mu}}_2$ can be directly obtained from (5.90b), but the MBSLIM implementation involves semi-definite positive mass matrices which not always have an inverse. This issue is circumvented in [4] by means of the approximation of this variable by means of the value of the equivalent index-1 DAE adjoint system. In order to obtain an scheme of solution equivalent to the ALI3-P system where $\boldsymbol{\mu}_2$ has an analog meaning, the conditions and adjoint variables are transformed, as it is described in the appendix D. The following assumption is done:

$$\left[\dot{\boldsymbol{\mu}}_2\right]^{t_F} \approx \left[\dot{\boldsymbol{\mu}}_2^{I1}\right]^{t_F} \tag{5.94}$$

where the variable obtained from the index-1 adjoint system is identified with the superscript $I1$.

As it can be deduced from the previous algorithm, the main challenge in the solution of (5.90) is related to the assumption that redundant constraints could be present.

The proposed algorithm for the solution of the adjoint equations for any time $t_i < t_F$ is devised here:

1. Determination of $\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}$ from (5.90d).

2. Determination of $\dot{\boldsymbol{\mu}}_{\ddot{\boldsymbol{\Phi}}}$ my means of a numerical integrator. In the present development, the backward implicit Newmark integrator is used, with the following expressions:

$$\dot{\boldsymbol{\mu}}_*^n = -\frac{\gamma}{\beta h}\boldsymbol{\mu}_*^n + \hat{\dot{\boldsymbol{\mu}}}_*^{n+1}; \qquad \hat{\dot{\boldsymbol{\mu}}}_*^{n+1} = \frac{\gamma}{\beta h}\boldsymbol{\mu}_*^{n+1} - \left(\frac{\gamma}{\beta} - 1\right)\dot{\boldsymbol{\mu}}_*^{n+1} \tag{5.95}$$

Note that only one integration in time is needed, since there are no adjoint variables with double time derivatives due to the change of variables included in the Lagrangian.

3. Computation of $\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}$ from (5.90c).

4. Calculation of $\boldsymbol{\mu}_2$, $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_{\boldsymbol{\Phi}}$ using (5.90a), (5.90b) and (5.90e). First, the variables $\dot{\boldsymbol{\mu}}_2$ and $\dot{\boldsymbol{\mu}}_1$ have to be integrated backwards in time. The application of the backwards integrator to (5.90a), (5.90b) and (5.90e) yields:

$$-\frac{\gamma}{\beta h}\boldsymbol{\mu}_1 + \hat{\dot{\boldsymbol{\mu}}}_1 - \bar{\mathbf{K}}^{\mathrm{T}}\boldsymbol{\mu}_2 - \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\mu}_{\boldsymbol{\Phi}} - \mathbf{A}^{\mathrm{T}}\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}} - \mathbf{B}^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}} = -\mathbf{g}_{\hat{\mathbf{z}}}^{\mathrm{T}}, \tag{5.96a}$$

$$\mathbf{M}^{\mathrm{T}}\left(-\frac{\gamma}{\beta h}\boldsymbol{\mu}_2 + \hat{\dot{\boldsymbol{\mu}}}_2\right) - \bar{\mathbf{P}}^{\mathrm{T}}\dot{\boldsymbol{\mu}}_{\ddot{\boldsymbol{\Phi}}} - \bar{\mathbf{C}}^{\mathrm{T}}\boldsymbol{\mu}_2 + \boldsymbol{\mu}_1 + \mathbf{E}^{\mathrm{T}}\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}} + \mathbf{F}^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}} = \mathbf{0}, \tag{5.96b}$$

$$\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\boldsymbol{\mu}_2 = \mathbf{g}_{\boldsymbol{\lambda}^*}^{\mathrm{T}}. \tag{5.96c}$$

For the sake of clearness, the superscripts $n$ and $n+1$ have been omitted from the integrator terms, being the term in positions evaluated at the time step $n$, and the correction part $\hat{\dot{\boldsymbol{\mu}}}_*$ calculated with the terms of the previously computed instant of time, which is $n+1$.

Rearranging the three equations into a matrix form:

$$\begin{bmatrix} -\frac{\gamma}{\beta h}\mathbf{I} & -\bar{\mathbf{K}}^{\mathrm{T}} & -\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \\ \mathbf{I} & -\frac{\gamma}{\beta h}\mathbf{M}^{\mathrm{T}} - \bar{\mathbf{C}}^{\mathrm{T}} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Phi}_{\hat{\mathbf{z}}} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \\ \boldsymbol{\mu}_{\boldsymbol{\Phi}} \end{bmatrix} = \begin{bmatrix} \mathbf{r}_1 \\ \mathbf{r}_2 \\ \mathbf{g}_{\boldsymbol{\lambda}^*}^{\mathrm{T}} \end{bmatrix} \tag{5.97}$$

with

$$\mathbf{r}_1 = -\mathbf{g}_{\hat{\mathbf{z}}}^{\mathrm{T}} + \mathbf{A}^{\mathrm{T}}\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}} + \mathbf{B}^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}} - \hat{\dot{\boldsymbol{\mu}}}_1 \tag{5.98a}$$

$$\mathbf{r}_2 = \bar{\mathbf{P}}^{\mathrm{T}}\dot{\boldsymbol{\mu}}_{\ddot{\boldsymbol{\Phi}}} - \mathbf{E}^{\mathrm{T}}\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}} - \mathbf{F}^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}} - \mathbf{M}^{\mathrm{T}}\hat{\dot{\boldsymbol{\mu}}}_2 \tag{5.98b}$$

According to [4], a scaling of this equation is recommended to improve its ill conditioning. Following these guidelines, the first equation is scaled by a factor of $\beta h^2$; the second equation is scaled by a factor of $\gamma h$; the first adjoint variable is substituted by the scaled one $\bar{\boldsymbol{\mu}}_1 = \gamma h \boldsymbol{\mu}_1$; finally, the third adjoint variable is substituted by $\bar{\boldsymbol{\mu}}_{\boldsymbol{\Phi}} = \beta h^2 \boldsymbol{\mu}_{\boldsymbol{\Phi}}$. The resulting scaled system of equations becomes:

$$\begin{bmatrix} -\mathbf{I} & -\beta h^2\bar{\mathbf{K}}^{\mathrm{T}} & -\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \\ \mathbf{I} & -\frac{\gamma^2}{\beta}\mathbf{M}^{\mathrm{T}} - \gamma h\bar{\mathbf{C}}^{\mathrm{T}} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Phi}_{\hat{\mathbf{z}}} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{\mu}}_1 \\ \boldsymbol{\mu}_2 \\ \bar{\boldsymbol{\mu}}_{\boldsymbol{\Phi}} \end{bmatrix} = \begin{bmatrix} \beta h^2\mathbf{r}_1 \\ \gamma h\mathbf{r}_2 \\ \mathbf{g}_{\boldsymbol{\lambda}^*}^{\mathrm{T}} \end{bmatrix} \tag{5.99}$$

If the Newmark integrator used is the trapezoidal rule, with $\beta = \dfrac{1}{4}$ and $\gamma = \dfrac{1}{2}$, the optimal scaled expression presented in [4] is reached.

The possible presence of redundant constraints entails infinite solutions of the previous system. Hence, there are various methods to obtain different valid solutions, such as the Moore-Penrose generalized inverse by means of singular value decomposition, introduced previously in this section. This method is absolutely valid, but it involves a high computational effort, which is specially relevant when this computation has to be executed at each time step.

Other possibility, presented in [4], consists in the solution of the system by means of a transformation of (5.99) into an augmented Lagrangian scheme. This approach offers a robust scheme of solution, accepting redundant constraints, with low computational effort. The first step to achieve an augmented Lagrangian scheme is to add the first and second equations of (5.99):

$$\begin{bmatrix} -\dfrac{\gamma^2}{\beta}\mathbf{M}^{\mathrm{T}} - \gamma h \bar{\mathbf{C}}^{\mathrm{T}} - \beta h^2 \bar{\mathbf{K}}^{\mathrm{T}} & -\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \\ \boldsymbol{\Phi}_{\hat{\mathbf{z}}} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\mu}_2 \\ \bar{\boldsymbol{\mu}}_{\boldsymbol{\Phi}} \end{bmatrix} = \begin{bmatrix} \gamma h \mathbf{r}_2 + \beta h^2 \mathbf{r}_1 \\ \mathbf{g}_{\boldsymbol{\lambda}^*}^{\mathrm{T}} \end{bmatrix} \tag{5.100}$$

Observe that this system is analog to the classical Index-1 Lagrange formulation, so its transformation into an augmented Lagrangian scheme is almost straightforward.

$$\left(\frac{\gamma^2}{\beta}\mathbf{M}^d + \gamma h \bar{\mathbf{C}} + \beta h^2 \bar{\mathbf{K}}\right)^{\mathrm{T}} \boldsymbol{\mu}_2 + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \left(\bar{\boldsymbol{\mu}}_{\boldsymbol{\Phi}}^{*(i)} - \boldsymbol{\alpha}_a \left(\mathbf{g}_{\boldsymbol{\lambda}} - \boldsymbol{\Phi}_{\hat{\mathbf{z}}}\boldsymbol{\mu}\right)\right) = -\gamma h \mathbf{r}_2 - \beta h^2 \mathbf{r}_1 \tag{5.101a}$$

$$\bar{\boldsymbol{\mu}}_{\boldsymbol{\Phi}}^{*(i)} = \bar{\boldsymbol{\mu}}_{\boldsymbol{\Phi}}^{*(i-1)} - \boldsymbol{\alpha}_a \left(\mathbf{g}_{\boldsymbol{\lambda}} - \boldsymbol{\Phi}_{\hat{\mathbf{z}}}\boldsymbol{\mu}\right) \tag{5.101b}$$

where $\boldsymbol{\alpha}_a \in \mathbb{R}^{m\times m}$ is a diagonal matrix with penalty factors, opening the possibility of different factors per constraint equation, as it has been implemented in the dynamics.

The determination of the values of the adjoint variables makes possible to compute the gradient of the objective function without the evaluation of the sensitivities of the states, using the remaining terms described in (5.89). Despite the complexity of the initialization problem and the conditions that have to fulfill the expression of the objective function at time $t_F$, the solution of the system can be executed with low computational effort, involving two linear equations and one augmented Lagrangian system. It has been proved that this method, and particularly this algorithm, offers good results in terms of accuracy along with high efficiency when a high number of parameters is used.

In this section, the method described in [4] has been successfully extended to semi-recursive formulations based on joint coordinates, and it has been tested and implemented in the MBSLIM multibody library as a general sensitivity formulation.

During the study and implementation of this method, a different and simpler approach has been developed as an alternative in order to avoid complex initialization problems. As a result, the discrete approach has been achieved.

#### 5.4.2.2 Discrete approach

The discrete approach is based on the use of the discrete expressions of the dynamics for building the adjoint variable Lagrangian. Basically, this approach represents the adjoint of the discretization, which means that the adjoint is generated with the discrete equations of motion and the discrete variables obtained in the dynamics. The approach has been successfully applied to multibody models in [96] with a Runge-Kutta integrator and in [97,128] with Hilber-Hughes-Taylor integrator, among others.

In general, this approach is less cumbersome than the continuous method, specially with DAE systems, but it has as main drawback the generation of particular expressions involving the numerical integrator applied to the dynamics.

First of all, the discrete nature of the discrete AVM implies that any integral will be substituted by its discrete form in terms of sums. For simplicity, the integral function is discretized by means of the trapezoidal rule, with the form:

$$\int_{t_0}^{t_F} \mathbf{X} \mathrm{d}t = \frac{h}{2}\left(\mathbf{X}_0 + \mathbf{X}_n\right) + h\sum_{i=1}^{n-1} \mathbf{X}_i \tag{5.102}$$

where $h$ is the step of time of the discretization, $n = \frac{t_F - t_0}{h}$ the number of steps and $\mathbf{X}_i$ the value of $\mathbf{X}$ at the time $ih + t_0$.

Accordingly, the objective function:

$$\boldsymbol{\psi} = \int_{t_0}^{t_F} \mathbf{g} \mathrm{d}t \tag{5.103}$$

can be discretized and transformed into:

$$\boldsymbol{\psi} = \frac{h}{2}\left(\mathbf{g}_0 + \mathbf{g}_n\right) + h\sum_{i=1}^{n-1} \mathbf{g}_i \tag{5.104}$$

In this approach, the index-3 DAE system is better suited to build the adjoint system than the augmented Lagrangian index-3 DAE as long as it does not require an iteration for the Lagrange multipliers. The lemma 4.3 presented in [4] established the basis to interchange these two formulations within a sensitivity analysis, using the following upgrade of the Lagrange multipliers

$$\boldsymbol{\lambda} \approx \left(\boldsymbol{\lambda}^* + \boldsymbol{\alpha}\boldsymbol{\Phi}\right) \tag{5.105}$$

where $\boldsymbol{\lambda}$ are the Lagrange multipliers of the index-3 DAE and $\boldsymbol{\lambda}^*$ are the Lagrange multipliers of the augmented Lagrangian index-3.

The semi-recursive index-3 DAE tangent linear model takes the form:

$$\mathbf{M}^{d}\ddot{\mathbf{z}}^{*\prime} + \mathbf{C}\dot{\mathbf{z}}^{*\prime} + \left(\mathbf{M}^{d}_{\hat{\mathbf{z}}}\ddot{\mathbf{z}}^{*} + \mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}\boldsymbol{\lambda} + \mathbf{K}\right)\mathbf{z}' + \mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\boldsymbol{\lambda}' = \mathbf{Q}^{d}_{\hat{\boldsymbol{\rho}}} - \mathbf{M}^{d}_{\hat{\boldsymbol{\rho}}}\ddot{\mathbf{z}}^{*} - \mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}\boldsymbol{\lambda} \tag{5.106a}$$

$$\mathbf{\Phi}_{\hat{\mathbf{z}}}\mathbf{z}' = -\mathbf{\Phi}_{\hat{\boldsymbol{\rho}}} \tag{5.106b}$$

and with the substitution of the Lagrange multipliers by the ones of the augmented Lagrangian scheme, it becomes:

$$\begin{aligned}\mathbf{M}^{d}\ddot{\mathbf{z}}^{*\prime} + \mathbf{C}\dot{\mathbf{z}}^{*\prime} + \left(\mathbf{M}^{d}_{\hat{\mathbf{z}}}\ddot{\mathbf{z}}^{*} + \mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\boldsymbol{\alpha}\mathbf{\Phi}_{\hat{\mathbf{z}}} + \mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}\left(\boldsymbol{\lambda}^{*} + \boldsymbol{\alpha}\mathbf{\Phi}\right) + \mathbf{K}\right)\mathbf{z}' + \mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\boldsymbol{\lambda}^{*\prime} = \\ \mathbf{Q}^{d}_{\hat{\boldsymbol{\rho}}} - \mathbf{M}^{d}_{\hat{\boldsymbol{\rho}}}\ddot{\mathbf{z}}^{*} - \mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}\left(\boldsymbol{\lambda}^{*} + \boldsymbol{\alpha}\mathbf{\Phi}\right) - \mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\boldsymbol{\alpha}\mathbf{\Phi}_{\hat{\boldsymbol{\rho}}}\end{aligned} \tag{5.107a}$$

$$\mathbf{\Phi}_{\hat{\mathbf{z}}}\mathbf{z}' = -\mathbf{\Phi}_{\hat{\boldsymbol{\rho}}} \tag{5.107b}$$

The discrete approach used to solve the dynamics requires to handle the derivatives of the equations with the numerical integrator already applied to them. In the present development, the Newmark's integrator is selected due to its simplicity and good behavior, with the sensitivities of positions of the states as the main variables of the system. The application of the integrator to the index-3 DAE TLM yields:

$$\begin{aligned}\left(\mathbf{M}^{d}_{\hat{\mathbf{z}}}\ddot{\mathbf{z}}^{*} + \mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\boldsymbol{\alpha}\mathbf{\Phi}_{\hat{\mathbf{z}}} + \mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}\left(\boldsymbol{\lambda}^{*} + \boldsymbol{\alpha}\mathbf{\Phi}\right) + \mathbf{K} + \frac{1}{\beta h}\mathbf{M}^{d} + \frac{\gamma}{\beta h}\mathbf{C}\right)\mathbf{z}^{*\prime} + \mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\boldsymbol{\lambda}^{*\prime} = \\ \mathbf{Q}^{d}_{\hat{\boldsymbol{\rho}}} - \mathbf{M}^{d}_{\hat{\boldsymbol{\rho}}}\ddot{\mathbf{z}}^{*} - \mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}\left(\boldsymbol{\lambda}^{*} + \boldsymbol{\alpha}\mathbf{\Phi}\right) - \mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\boldsymbol{\alpha}\mathbf{\Phi}_{\hat{\boldsymbol{\rho}}} - \mathbf{M}^{d}\hat{\ddot{\mathbf{z}}}' - \mathbf{C}\hat{\dot{\mathbf{z}}}'\end{aligned} \tag{5.108a}$$

$$\mathbf{\Phi}_{\hat{\mathbf{z}}}\mathbf{z}' = -\mathbf{\Phi}_{\hat{\boldsymbol{\rho}}} \tag{5.108b}$$

with the following Newmark's integrator equations used for the sensitivities of the states:

$$\dot{\mathbf{z}}^{*\prime}_{i} = \frac{\gamma}{\beta h}\mathbf{z}'_{i} + \hat{\dot{\mathbf{z}}}'_{i-1}; \qquad \hat{\dot{\mathbf{z}}}'_{i-1} = \left\{-\frac{\gamma}{\beta h}\mathbf{z}'_{i-1} - \left(\frac{\gamma}{\beta} - 1\right)\dot{\mathbf{z}}'_{i-1} - \left(\frac{\gamma}{2\beta} - 1\right)h\ddot{\mathbf{z}}'_{i-1}\right\} \tag{5.109a}$$

$$\ddot{\mathbf{z}}^{*\prime}_{i} = \frac{1}{\beta h^{2}}\mathbf{z}'_{i} + \hat{\ddot{\mathbf{z}}}'_{i-1}; \qquad \hat{\ddot{\mathbf{z}}}'_{i-1} = \left\{-\frac{1}{\beta h^{2}}\mathbf{z}_{i-1} - \frac{1}{\beta h}\dot{\mathbf{z}}_{i-1} - \left(\frac{1}{2\beta} - 1\right)\ddot{\mathbf{z}}_{i-1}\right\} \tag{5.109b}$$

The application of the Newmark integrator to the sensitivities of the non-iterative velocity projections yields:

$$\begin{aligned}\left(\bar{\mathbf{P}} + \varsigma\mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\boldsymbol{\alpha}\mathbf{\Phi}_{\hat{\mathbf{z}}}\right)\dot{\mathbf{z}}'^{\{i\}} = \bar{\mathbf{P}}\left(\frac{\gamma}{\beta h}\mathbf{z}' + \hat{\dot{\mathbf{z}}}'\right) + \bar{\mathbf{P}}'\left(\dot{\mathbf{z}}^{*} - \dot{\mathbf{z}}\right) - \mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\dot{\mathbf{\Phi}}\mathbf{z}' \\ -\mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}\varsigma\boldsymbol{\alpha}\dot{\mathbf{\Phi}} - \mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\mathbf{b}^{\boldsymbol{\rho}}\end{aligned} \tag{5.110}$$

with

$$\bar{\mathbf{P}}' = \bar{\mathbf{P}}_{\hat{\mathbf{z}}}\mathbf{z}' + \bar{\mathbf{P}}_{\hat{\dot{\mathbf{z}}}^{*}}\left(\frac{\gamma}{\beta h}\mathbf{z}' + \hat{\dot{\mathbf{z}}}'\right) + \bar{\mathbf{P}}_{\hat{\boldsymbol{\rho}}} \tag{5.111}$$

Similarly, the discrete sensitivities of the acceleration projections become:

$$\begin{aligned}\left(\bar{\mathbf{P}} + \varsigma\mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\boldsymbol{\alpha}\mathbf{\Phi}_{\hat{\mathbf{z}}}\right)\ddot{\mathbf{z}}'^{\{i\}} = \bar{\mathbf{P}}\left(\frac{1}{\beta h^{2}}\mathbf{z}' + \hat{\ddot{\mathbf{z}}}'\right) + \bar{\mathbf{P}}'\left(\ddot{\mathbf{z}}^{*} - \ddot{\mathbf{z}}\right) - \mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\ddot{\mathbf{\Phi}}\mathbf{z}' \\ -\mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}\varsigma\boldsymbol{\alpha}\ddot{\mathbf{\Phi}} - \mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\mathbf{c}^{\boldsymbol{\rho}}\end{aligned} \tag{5.112}$$

Observe that, since differentiation and discretization are decoupled processes for this numerical integrator and sensitivity expressions, their order can be interchanged, which means that the discretization of the continuous sensitivities of the EoM is equivalent to the differentiation of the discrete EoM. Upon that base the discrete adjoint variable method will be built.

The first step in the generation of the adjoint equations is the composition of a Lagrangian preserving the same value of the objective function but including a set of new adjoint variables and the constraint conditions that the system has to satisfy. Before looking at the details on how to build it, let us consider the following discrete adjoint Lagrangian:

$$\mathcal{L} = \frac{h}{2}\left(\mathcal{L}_0 + \mathcal{L}_n\right) + h\sum_{i=1}^{n-1}\mathcal{L}_i \tag{5.113}$$

where, $\mathcal{L}_i$ is the value of $\mathcal{L}$ at time $ih + t_0$ and $\mathcal{L}_0$, $\mathcal{L}_n$ are the initial and final values of $\mathcal{L}$ at $t_0$ and $t_F$, respectively.

Analogously, the gradient of the Lagrangian involving a discrete integral can be computed as:

$$\mathcal{L}' = \frac{h}{2}\left(\mathcal{L}'_0 + \mathcal{L}'_n\right) + h\sum_{i=1}^{n-1}\mathcal{L}'_i \tag{5.114}$$

Once the Lagrangian (5.113) and its gradient (5.114) are expressed in terms of discrete magnitudes, the equations of the Lagrangian for each instant of time can be introduced. Let us define the Lagrangian at an instant of time $t_i \in (t_0, t_F]$ such as:

$$\begin{aligned}\mathcal{L}_i = \mathbf{g}_i - \boldsymbol{\mu}^{\mathrm{T}}\left(\mathbf{M}^d\ddot{\mathbf{z}}^* + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\left(\boldsymbol{\lambda}^* + \boldsymbol{\alpha\Phi}\right) - \mathbf{Q}\right) - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\boldsymbol{\Phi}\\ -\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}}\left(\left[\bar{\mathbf{P}} + \varsigma\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\alpha}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right]\dot{\mathbf{z}} - \bar{\mathbf{P}}\dot{\mathbf{z}}^* + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\varsigma\boldsymbol{\alpha}\boldsymbol{\Phi}_t\right) -\\ -\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\left(\left[\bar{\mathbf{P}} + \varsigma\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\alpha}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right]\ddot{\mathbf{z}} - \bar{\mathbf{P}}\ddot{\mathbf{z}}^* + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}\varsigma\boldsymbol{\alpha}\left(\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\dot{\mathbf{z}} + \dot{\boldsymbol{\Phi}}_t\right)\right)\end{aligned} \tag{5.115}$$

in which:

- $\boldsymbol{\mu} \in \mathbb{R}^{n \times o}$ is the set of adjoint variables associated to the first $n$ equations of the index-3 DAE system.

- $\boldsymbol{\mu}_{\boldsymbol{\Phi}} \in \mathbb{R}^{m \times o}$ are the adjoint variables associated to the last $m$ equations of the index-3 DAE system.

- $\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}} \in \mathbb{R}^{n \times o}$ is the set of adjoint variables related to the velocity projections.

- $\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}} \in \mathbb{R}^{n \times o}$ is the set of adjoint variables corresponding to acceleration projections.

The gradient of this instant Lagrangian, considering the discrete derivatives introduced in (5.108), (5.110) and (5.112), has the following expression:

$$
\begin{aligned}
\mathcal{L}_i' = \Bigg\{ & \mathbf{g}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}^{\mathrm{T}} \left( \frac{1}{\beta h^2} \mathbf{M}^d + \frac{\gamma}{\beta h} \bar{\mathbf{C}} + \bar{\mathbf{K}} \right) + \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \\
& + \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \bar{\mathbf{P}} \frac{\gamma}{\beta h} + \left( \bar{\mathbf{P}}_{\hat{\mathbf{z}}} + \frac{\gamma}{\beta h} \bar{\mathbf{P}}_{\hat{\dot{\mathbf{z}}}^*} \right) (\dot{\mathbf{z}}^* - \dot{\mathbf{z}}) - \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \dot{\boldsymbol{\Phi}} - \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} ) \right) \\
& + \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \bar{\mathbf{P}} \frac{1}{\beta h^2} + \left( \bar{\mathbf{P}}_{\hat{\mathbf{z}}} + \frac{\gamma}{\beta h} \bar{\mathbf{P}}_{\hat{\dot{\mathbf{z}}}^*} \right) (\ddot{\mathbf{z}}^* - \ddot{\mathbf{z}}) - \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \ddot{\boldsymbol{\Phi}} - \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} \right) \Bigg\} \mathbf{z}' \\
& + \left\{ \mathbf{g}_{\hat{\dot{\mathbf{z}}}} - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \bar{\mathbf{P}} + \varsigma \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \boldsymbol{\alpha} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \right) - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \left( \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}} \dot{\mathbf{z}} + \dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}t} \right) \right) \right\} \dot{\mathbf{z}}' \\
& + \left\{ \mathbf{g}_{\hat{\ddot{\mathbf{z}}}} - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \bar{\mathbf{P}} + \varsigma \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \boldsymbol{\alpha} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \right) \right\} \ddot{\mathbf{z}}' \\
& - \left\{ \left( \boldsymbol{\mu}^{\mathrm{T}} \mathbf{C} - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}} \bar{\mathbf{P}} \right) \right\} \hat{\dot{\mathbf{z}}}' \\
& - \left\{ \boldsymbol{\mu}^{\mathrm{T}} \mathbf{M}^d - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \bar{\mathbf{P}} + \bar{\mathbf{P}}_{\hat{\dot{\mathbf{z}}}^*} (\dot{\mathbf{z}}^* - \dot{\mathbf{z}}) \right) - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \bar{\mathbf{P}}_{\hat{\dot{\mathbf{z}}}^*} (\ddot{\mathbf{z}}^* - \ddot{\mathbf{z}}) \right\} \hat{\ddot{\mathbf{z}}}' \\
& + \left\{ \mathbf{g}_{\boldsymbol{\lambda}^*} - \boldsymbol{\mu}^{\mathrm{T}} \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \right\} \boldsymbol{\lambda}^{*\prime} \\
& + \Big\{ \Big( \mathbf{g}_{\hat{\boldsymbol{\rho}}} + \boldsymbol{\mu}^{\mathrm{T}} \left( \mathbf{Q}_{\hat{\boldsymbol{\rho}}}^d - \mathbf{M}_{\hat{\boldsymbol{\rho}}}^d \ddot{\mathbf{z}}^* - \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}} (\boldsymbol{\lambda}^* + \boldsymbol{\alpha} \boldsymbol{\Phi}) - \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \boldsymbol{\alpha} \boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}} \right) - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}} \boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}} \\
& - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \dot{\boldsymbol{\Phi}} - \bar{\mathbf{P}}_{\hat{\boldsymbol{\rho}}} (\dot{\mathbf{z}}^* - \dot{\mathbf{z}}) + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \dot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}} \right) \\
& - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \ddot{\boldsymbol{\Phi}} - \bar{\mathbf{P}}_{\hat{\boldsymbol{\rho}}} (\ddot{\mathbf{z}}^* - \ddot{\mathbf{z}}) + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \ddot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}} \right) \Big) \Big\}
\end{aligned}
\tag{5.116}
$$

wherein all the magnitudes within the previous equations are evaluated at time $t_i$.

The instant Lagrangian sensitivities can be substituted into (5.114), conforming an unique expression with the adjoint variables, the sensitivities of the states and the sensitivities of the Lagrange multipliers from every time step as unknowns. Since the values of the Lagrangian and objective function gradients are identical for any value of the adjoint variables, these variables can be selected such as they nullify the terms multiplying the unknown sensitivities of the states and the Lagrange multipliers, hence avoiding their calculation.

Returning again to equation (5.116), it can be seen that $\hat{\dot{\mathbf{z}}}'$ and $\hat{\ddot{\mathbf{z}}}'$ could be transformed into expressions dependent on the sensitivities of the previous step of time, using (5.109). Therefore, the instant Lagrangian for any time $t_i$ becomes:

$$
\begin{aligned}
\mathcal{L}_i' = \Bigg\{ & \mathbf{g}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}^{\mathrm{T}} \left( \frac{1}{\beta h^2} \mathbf{M}^d + \frac{\gamma}{\beta h} \bar{\mathbf{C}} + \bar{\mathbf{K}} \right) + \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \\
& + \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \bar{\mathbf{P}} \frac{\gamma}{\beta h} + \left( \bar{\mathbf{P}}_{\hat{\mathbf{z}}} + \frac{\gamma}{\beta h} \bar{\mathbf{P}}_{\hat{\dot{\mathbf{z}}}^*} \right) \left( \dot{\mathbf{z}}^* - \dot{\mathbf{z}} \right) - \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \dot{\boldsymbol{\Phi}} - \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} \right) \\
& + \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \bar{\mathbf{P}} \frac{1}{\beta h^2} + \left( \bar{\mathbf{P}}_{\hat{\mathbf{z}}} + \frac{\gamma}{\beta h} \bar{\mathbf{P}}_{\hat{\dot{\mathbf{z}}}^*} \right) \left( \ddot{\mathbf{z}}^* - \ddot{\mathbf{z}} \right) - \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \ddot{\boldsymbol{\Phi}} - \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} \right) \Bigg\} \mathbf{z}' \\
& + \left\{ \mathbf{g}_{\hat{\dot{\mathbf{z}}}} - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \bar{\mathbf{P}} + \varsigma \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \boldsymbol{\alpha} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \right) - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \left( \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}} \dot{\mathbf{z}} + \dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}t} \right) \right) \right\} \dot{\mathbf{z}}' \\
& + \left\{ \mathbf{g}_{\hat{\ddot{\mathbf{z}}}} - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \bar{\mathbf{P}} + \varsigma \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \boldsymbol{\alpha} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \right) \right\} \ddot{\mathbf{z}}' \\
& - \left\{ \left( \boldsymbol{\mu}^{\mathrm{T}} \mathbf{C} - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}} \bar{\mathbf{P}} \right) \right\} \left\{ - \frac{\gamma}{\beta h} \mathbf{z}_{i-1}' - \left( \frac{\gamma}{\beta} - 1 \right) \dot{\mathbf{z}}_{i-1}' - \left( \frac{\gamma}{2\beta} - 1 \right) h \ddot{\mathbf{z}}_{i-1}' \right\} \\
& - \left\{ \boldsymbol{\mu}^{\mathrm{T}} \mathbf{M}^d - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \bar{\mathbf{P}} + \bar{\mathbf{P}}_{\hat{\dot{\mathbf{z}}}^*} \left( \dot{\mathbf{z}}^* - \dot{\mathbf{z}} \right) \right) - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \bar{\mathbf{P}}_{\hat{\dot{\mathbf{z}}}^*} \left( \ddot{\mathbf{z}}^* - \ddot{\mathbf{z}} \right) \right\} \Bigg\{ - \frac{1}{\beta h^2} \mathbf{z}_{i-1}' \\
& - \frac{1}{\beta h} \dot{\mathbf{z}}_{i-1}' - \left( \frac{1}{2\beta} - 1 \right) \ddot{\mathbf{z}}_{i-1}' \Bigg\} \\
& + \left\{ \mathbf{g}_{\boldsymbol{\lambda}^*} - \boldsymbol{\mu}^{\mathrm{T}} \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \right\} \boldsymbol{\lambda}^{*\prime} \\
& + \Big\{ \left( \mathbf{g}_{\hat{\boldsymbol{\rho}}} + \boldsymbol{\mu}^{\mathrm{T}} \left( \mathbf{Q}_{\hat{\boldsymbol{\rho}}}^d - \mathbf{M}_{\hat{\boldsymbol{\rho}}}^d \ddot{\mathbf{z}}^* - \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}} \left( \boldsymbol{\lambda}^* + \boldsymbol{\alpha} \boldsymbol{\Phi} \right) - \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \boldsymbol{\alpha} \boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}} \right) - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}} \boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}} \\
& - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \dot{\boldsymbol{\Phi}} - \bar{\mathbf{P}}_{\hat{\boldsymbol{\rho}}} \left( \dot{\mathbf{z}}^* - \dot{\mathbf{z}} \right) + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \dot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}} \right) \\
& - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \ddot{\boldsymbol{\Phi}} - \bar{\mathbf{P}}_{\hat{\boldsymbol{\rho}}} \left( \ddot{\mathbf{z}}^* - \ddot{\mathbf{z}} \right) + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \ddot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}} \right) \right) \Big\}
\end{aligned}
\tag{5.117}
$$

Observe that the sensitivity of each instant Lagrangian involves the sensitivities of the states in the time step $t_i$ and in the previous one $t_{i-1}$. The integrator terms induce "couplings" between subsequent time steps and imply that the instant state sensitivities $\mathbf{z}_{i-1}'$, $\dot{\mathbf{z}}_{i-1}'$ and $\ddot{\mathbf{z}}_{i-1}'$ affect both the instant Lagrangians $\mathcal{L}_{i-1}$ and $\mathcal{L}_i$. Consequently, the cancellation of terms multiplying the instant state sensitivities (required for the generation of the adjoint equations) cannot be addressed for each instant Lagrangian alone, but for both $\mathcal{L}_{i-1}$ and $\mathcal{L}_i$ together in accordance with the scheme of integration (5.114).

Let us now consider two subsequent steps of time $t_i$ and $t_{i+1}$, none of them being the initial or final time of the dynamic simulation. Both expressions appear in (5.114) multiplied by the same coefficient $h$, so it is avoided here for the sake of clarity.

$$
\begin{aligned}
\mathcal{L}'_i + \mathcal{L}'_{i+1} = \Bigg\{ & \mathbf{g}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}^{\mathrm{T}} \left( \frac{1}{\beta h^2} \mathrm{M}^d + \frac{\gamma}{\beta h} \bar{\mathbf{C}} + \bar{\mathbf{K}} \right) + \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \\
& + \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}} \Bigg( \bar{\mathbf{P}} \frac{\gamma}{\beta h} + \left( \bar{\mathbf{P}}_{\hat{\mathbf{z}}} + \frac{\gamma}{\beta h} \bar{\mathbf{P}}_{\dot{\hat{\mathbf{z}}}^*} \right) (\dot{\mathbf{z}}^* - \dot{\mathbf{z}}) - \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \dot{\boldsymbol{\Phi}} - \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}) \Bigg) \\
& + \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \Bigg( \bar{\mathbf{P}} \frac{1}{\beta h^2} + \left( \bar{\mathbf{P}}_{\hat{\mathbf{z}}} + \frac{\gamma}{\beta h} \bar{\mathbf{P}}_{\dot{\hat{\mathbf{z}}}^*} \right) (\ddot{\mathbf{z}}^* - \ddot{\mathbf{z}}) - \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \ddot{\boldsymbol{\Phi}} - \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} \Bigg) \Bigg\}_i \mathbf{z}'_i \\
& + \Bigg\{ \mathbf{g}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}^{\mathrm{T}} \left( \frac{1}{\beta h^2} \mathrm{M}^d + \frac{\gamma}{\beta h} \bar{\mathbf{C}} + \bar{\mathbf{K}} \right) + \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \\
& + \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}} \Bigg( \bar{\mathbf{P}} \frac{\gamma}{\beta h} + \left( \bar{\mathbf{P}}_{\hat{\mathbf{z}}} + \frac{\gamma}{\beta h} \bar{\mathbf{P}}_{\dot{\hat{\mathbf{z}}}^*} \right) (\dot{\mathbf{z}}^* - \dot{\mathbf{z}}) - \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \dot{\boldsymbol{\Phi}} - \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}) \Bigg) \\
& + \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \Bigg( \bar{\mathbf{P}} \frac{1}{\beta h^2} + \left( \bar{\mathbf{P}}_{\hat{\mathbf{z}}} + \frac{\gamma}{\beta h} \bar{\mathbf{P}}_{\dot{\hat{\mathbf{z}}}^*} \right) (\ddot{\mathbf{z}}^* - \ddot{\mathbf{z}}) - \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \ddot{\boldsymbol{\Phi}} - \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} \Bigg) \Bigg\}_{i+1} \mathbf{z}'_{i+1} \\
& + \Big\{ \mathbf{g}_{\dot{\hat{\mathbf{z}}}} - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \bar{\mathbf{P}} + \varsigma \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \boldsymbol{\alpha} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \right) - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \left( \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}} \dot{\mathbf{z}} + \dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}t} \right) \right) \Big\}_i \dot{\mathbf{z}}'_i \\
& + \Big\{ \mathbf{g}_{\dot{\hat{\mathbf{z}}}} - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \bar{\mathbf{P}} + \varsigma \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \boldsymbol{\alpha} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \right) - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \left( \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}} \dot{\mathbf{z}} + \dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}t} \right) \right) \Big\}_{i+1} \dot{\mathbf{z}}'_{i+1} \\
& - \left\{ \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \bar{\mathbf{P}} + \varsigma \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \boldsymbol{\alpha} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \right) \right\}_i \ddot{\mathbf{z}}'_i - \left\{ \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \bar{\mathbf{P}} + \varsigma \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \boldsymbol{\alpha} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \right) \right\}_{i+1} \ddot{\mathbf{z}}'_{i+1} \\
& - \left\{ \left( \boldsymbol{\mu}^{\mathrm{T}} \mathbf{C} - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}} \bar{\mathbf{P}} \right) \right\}_i \left\{ -\frac{\gamma}{\beta h} \mathbf{z}'_{i-1} - \left( \frac{\gamma}{\beta} - 1 \right) \dot{\mathbf{z}}'_{i-1} - \left( \frac{\gamma}{2\beta} - 1 \right) h \ddot{\mathbf{z}}'_{i-1} \right\} \\
& - \left\{ \left( \boldsymbol{\mu}^{\mathrm{T}} \mathbf{C} - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}} \bar{\mathbf{P}} \right) \right\}_{i+1} \left\{ -\frac{\gamma}{\beta h} \mathbf{z}'_i - \left( \frac{\gamma}{\beta} - 1 \right) \dot{\mathbf{z}}'_i - \left( \frac{\gamma}{2\beta} - 1 \right) h \ddot{\mathbf{z}}'_i \right\} \\
& - \left\{ \boldsymbol{\mu}^{\mathrm{T}} \mathrm{M}^d - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \bar{\mathbf{P}} + \bar{\mathbf{P}}_{\dot{\hat{\mathbf{z}}}^*} (\dot{\mathbf{z}}^* - \dot{\mathbf{z}}) \right) - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \bar{\mathbf{P}}_{\dot{\hat{\mathbf{z}}}^*} (\ddot{\mathbf{z}}^* - \ddot{\mathbf{z}}) \right\}_i \Bigg\{ -\frac{1}{\beta h^2} \mathbf{z}'_{i-1} \\
& \qquad - \frac{1}{\beta h} \dot{\mathbf{z}}'_{i-1} - \left( \frac{1}{2\beta} - 1 \right) \ddot{\mathbf{z}}'_{i-1} \Bigg\} \\
& - \left\{ \boldsymbol{\mu}^{\mathrm{T}} \mathrm{M}^d - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \bar{\mathbf{P}} + \bar{\mathbf{P}}_{\dot{\hat{\mathbf{z}}}^*} (\dot{\mathbf{z}}^* - \dot{\mathbf{z}}) \right) - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \bar{\mathbf{P}}_{\dot{\hat{\mathbf{z}}}^*} (\ddot{\mathbf{z}}^* - \ddot{\mathbf{z}}) \right\}_{i+1} \Bigg\{ -\frac{1}{\beta h^2} \mathbf{z}'_i \\
& \qquad - \frac{1}{\beta h} \dot{\mathbf{z}}'_i - \left( \frac{1}{2\beta} - 1 \right) \ddot{\mathbf{z}}'_i \Bigg\} \\
& + \left\{ \mathbf{g}_{\boldsymbol{\lambda}^*} - \boldsymbol{\mu}^{\mathrm{T}} \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \right\}_i \boldsymbol{\lambda}^{*\prime}_i + \left\{ \mathbf{g}_{\boldsymbol{\lambda}^*} - \boldsymbol{\mu}^{\mathrm{T}} \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \right\}_{i+1} \boldsymbol{\lambda}^{*\prime}_{i+1} \\
& + \Big\{ \Big( \mathbf{g}_{\hat{\boldsymbol{\rho}}} + \boldsymbol{\mu}^{\mathrm{T}} \left( \mathrm{Q}_{\hat{\boldsymbol{\rho}}}^d - \mathrm{M}_{\hat{\boldsymbol{\rho}}}^d \ddot{\mathbf{z}}^* - \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}} (\boldsymbol{\lambda}^* + \boldsymbol{\alpha} \boldsymbol{\Phi}) - \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \boldsymbol{\alpha} \boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}} \right) - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}} \boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}} \\
& \qquad - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \dot{\boldsymbol{\Phi}} - \bar{\mathbf{P}}_{\hat{\boldsymbol{\rho}}} (\dot{\mathbf{z}}^* - \dot{\mathbf{z}}) + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \dot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}} \right) \\
& \qquad - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \ddot{\boldsymbol{\Phi}} - \bar{\mathbf{P}}_{\hat{\boldsymbol{\rho}}} (\ddot{\mathbf{z}}^* - \ddot{\mathbf{z}}) + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \ddot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}} \right) \Big) \Big\}_i \\
& + \Big\{ \Big( \mathbf{g}_{\hat{\boldsymbol{\rho}}} + \boldsymbol{\mu}^{\mathrm{T}} \left( \mathrm{Q}_{\hat{\boldsymbol{\rho}}}^d - \mathrm{M}_{\hat{\boldsymbol{\rho}}}^d \ddot{\mathbf{z}}^* - \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}} (\boldsymbol{\lambda}^* + \boldsymbol{\alpha} \boldsymbol{\Phi}) - \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \boldsymbol{\alpha} \boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}} \right) - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}} \boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}} \\
& \qquad - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \dot{\boldsymbol{\Phi}} - \bar{\mathbf{P}}_{\hat{\boldsymbol{\rho}}} (\dot{\mathbf{z}}^* - \dot{\mathbf{z}}) + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \dot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}} \right) \\
& \qquad - \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \left( \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \ddot{\boldsymbol{\Phi}} - \bar{\mathbf{P}}_{\hat{\boldsymbol{\rho}}} (\ddot{\mathbf{z}}^* - \ddot{\mathbf{z}}) + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \varsigma \boldsymbol{\alpha} \ddot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}} \right) \Big) \Big\}_{i+1}
\end{aligned}
\tag{5.118}
$$

where the subscripts $i$ and $i+1$ indicate that the magnitudes are calculated in time instants $t_i$ and $t_{i+1}$ respectively.

If the evaluation of $\mathbf{z}_i'$, $\dot{\mathbf{z}}_i'$, $\ddot{\mathbf{z}}_i'$ and $\boldsymbol{\lambda}_i'$ is intended to be avoided, all the terms of (5.118) multiplying them have to be nullified. Accordingly, the following equations are obtained:

$$\left\{\left(\frac{1}{\beta h^2}\mathbf{M}^d+\frac{\gamma}{\beta h}\bar{\mathbf{C}}+\bar{\mathbf{K}}\right)^{\mathrm{T}}\boldsymbol{\mu}+\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\mu}_{\boldsymbol{\Phi}}-\mathbf{A}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}}\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}-\mathbf{A}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}\right\}_{\{i\}}=\mathbf{g}_{\hat{\mathbf{z}}\{i\}}^{\mathrm{T}} + \left\{\frac{\gamma}{\beta h}\mathbf{G}^{\mathrm{T}}+\frac{1}{\beta h^2}\mathbf{H}^{\mathrm{T}}\right\}_{\{i+1\}} \tag{5.119a}$$

$$\left\{\left(\bar{\mathbf{P}}+\varsigma\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\alpha}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right)^{\mathrm{T}}\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}+\left(\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\varsigma\boldsymbol{\alpha}\left(2\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\right)\right)^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}\right\}_{\{i\}}=\mathbf{g}_{\dot{\hat{\mathbf{z}}}\{i\}}^{\mathrm{T}} - \left\{\left(1-\frac{\gamma}{\beta}\right)\mathbf{G}^{\mathrm{T}}-\frac{1}{\beta h}\mathbf{H}^{\mathrm{T}}\right\}_{\{i+1\}} \tag{5.119b}$$

$$\left\{\left(\bar{\mathbf{P}}+\varsigma\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\alpha}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right)^{\mathrm{T}}\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}\right\}_{\{i\}}=\mathbf{g}_{\ddot{\hat{\mathbf{z}}}\{i\}}^{\mathrm{T}}-\left\{\left(1-\frac{\gamma}{2\beta}\right)h\mathbf{G}^{\mathrm{T}}+\left(1-\frac{1}{2\beta}\right)\mathbf{H}^{\mathrm{T}}\right\}_{\{i+1\}} \tag{5.119c}$$

$$\left\{\boldsymbol{\mu}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\right\}_{\{i\}}=\mathbf{g}_{\boldsymbol{\lambda}^*\{i\}} \tag{5.119d}$$

where:

$$\mathbf{G}=\boldsymbol{\mu}^{\mathrm{T}}\mathbf{C}-\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}}\left(\bar{\mathbf{P}}+\bar{\mathbf{P}}_{\dot{\hat{\mathbf{z}}}^*}(\dot{\mathbf{z}}^*-\dot{\mathbf{z}})\right)-\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\bar{\mathbf{P}}_{\dot{\hat{\mathbf{z}}}^*}\left(\ddot{\mathbf{z}}^*-\ddot{\mathbf{z}}\right) \tag{5.120a}$$

$$\mathbf{H}=\boldsymbol{\mu}^{\mathrm{T}}\mathbf{M}^d-\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}\bar{\mathbf{P}} \tag{5.120b}$$

$$\mathbf{A}_{\dot{\boldsymbol{\Phi}}}=\bar{\mathbf{P}}\frac{\gamma}{\beta h}+\left(\bar{\mathbf{P}}_{\hat{\mathbf{z}}}+\frac{\gamma}{\beta h}\bar{\mathbf{P}}_{\dot{\hat{\mathbf{z}}}^*}\right)(\dot{\mathbf{z}}^*-\dot{\mathbf{z}})-\boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}}\varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}}-\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} \tag{5.120c}$$

$$\mathbf{A}_{\ddot{\boldsymbol{\Phi}}}=\bar{\mathbf{P}}\frac{1}{\beta h^2}+\left(\bar{\mathbf{P}}_{\hat{\mathbf{z}}}+\frac{\gamma}{\beta h}\bar{\mathbf{P}}_{\dot{\hat{\mathbf{z}}}^*}\right)(\ddot{\mathbf{z}}^*-\ddot{\mathbf{z}})-\boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}}\varsigma\boldsymbol{\alpha}\ddot{\boldsymbol{\Phi}}-\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\varsigma\boldsymbol{\alpha}\ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} \tag{5.120d}$$

Equations (5.119) for any $t_i \in (t_0, t_F)$ also represent the adjoint problem at time $t_F$, but according to (5.114), since there is no term evaluated at time $t_F + h$, $\mathbf{G}$ and $\mathbf{H}$ become null for this particular case. Behold that time $t_F$ neither requires a particular set of equations, nor the addition of new adjoint variables as it happens in the continuous AVM [4].

Adjoint equations (5.119) can be solved sequentially: firstly, $\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}$ can be determined from (5.119c); secondly, the value of $\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}$ can be substituted in (5.119b), so $\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}$ can be obtained; thirdly and last, (5.119a) and (5.119d) can be combined and solved jointly, giving as result the values of $\boldsymbol{\mu}$ and $\boldsymbol{\mu}_{\boldsymbol{\Phi}}$. The solution of this third system of equations is equivalent to (5.100) solved in the continuous approach, and suffers the same problems of indetermination when redundant constraints are part of the multibody model. Hence, the same techniques presented for (5.100) are equally valid in the discrete approach. In fact, the solution of (5.119a) and (5.119d) implemented in the MBSLIM multibody library is based on an augmented Lagrangian scheme with (5.119a) scaled by a factor of $\beta h^2$ in order to improve its numerical conditioning. The resulting expressions have the form:

$$\left(\mathbf{M}^d + \gamma h \bar{\mathbf{C}} + \beta h^2 \bar{\mathbf{K}}\right)^{\mathrm{T}} \boldsymbol{\mu} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \left(\boldsymbol{\mu}_{\boldsymbol{\Phi}}^{*(i)} - \boldsymbol{\alpha}_a \left(\mathbf{g}_{\boldsymbol{\lambda}} - \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \boldsymbol{\mu}\right)\right) = \mathbf{r}_1 \tag{5.121a}$$

$$\boldsymbol{\mu}_{\boldsymbol{\Phi}}^{*(i)} = \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{*(i-1)} - \boldsymbol{\alpha}_a \left(\mathbf{g}_{\boldsymbol{\lambda}} - \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \boldsymbol{\mu}\right) \tag{5.121b}$$

with

$$\mathbf{r}_1 = \left\{\mathbf{g}_{\hat{\mathbf{z}}}^{\mathrm{T}} - \mathbf{A}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}} \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}} - \mathbf{A}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}} \boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}\right\}_{\{i\}} + \left\{\frac{\gamma}{\beta h} \mathbf{G}^{\mathrm{T}} + \frac{1}{\beta h^2} \mathbf{H}^{\mathrm{T}}\right\}_{\{i+1\}} \tag{5.122}$$

In the discrete adjoint system, the different expressions used to solve the initial position, velocity and acceleration problems have to be considered in the Lagrangian in order to incorporate their effect on the sensitivities. Consequently, the adjoint of the initial position and velocity problem, given by (5.1.2), have to be computed at the first time instant of the simulation, but also the adjoint of the initial dynamic acceleration problem have to be addressed. In this work, the initial acceleration problem is solved by means of the semi-recursive penalty approach with a fixed point scheme in accelerations described in section 3.4, and whose sensitivity analysis was outlined in section 5.3. Here, the discrete adjoint equations needed for the initial instant of time during the discrete adjoint sensitivity analysis of a semi-recursive ALI3-P formulation is briefly described.

Equations (3.15), (3.19a) and (3.42) allow the computation of the instant Lagrangian at the initial time $t_0$ as:

$$\mathcal{L}_0 = \mathbf{g}_0 - \boldsymbol{\mu}_{\boldsymbol{\Phi}0}^{\mathrm{T}} \left(\begin{bmatrix} \boldsymbol{\Phi} \\ \mathbf{z}^i - \mathbf{d} \end{bmatrix}\right) - \boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}0}^{\mathrm{T}} \left(\begin{bmatrix} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \\ \mathbf{B} \end{bmatrix} \dot{\mathbf{z}} + \begin{bmatrix} \boldsymbol{\Phi}_t \\ -\dot{\mathbf{z}}^i \end{bmatrix}\right) - \boldsymbol{\mu}_0^{\mathrm{T}} \left(\breve{\mathbf{M}} \ddot{\mathbf{z}} - \breve{\mathbf{Q}}\right) \tag{5.123}$$

Now, applying the sensitivity expressions (5.4a), (5.4b) and (5.69) of the initial problems, the gradient of the instant Lagrangian at the initial time step is:

$$\begin{aligned} \mathcal{L}_0' = \mathbf{g}_{\hat{\mathbf{z}}} \mathbf{z}' + \mathbf{g}_{\hat{\dot{\mathbf{z}}}} \dot{\mathbf{z}}' + \mathbf{g}_{\hat{\ddot{\mathbf{z}}}} \ddot{\mathbf{z}}' + \mathbf{g}_{\hat{\rho}} - \boldsymbol{\mu}_{\boldsymbol{\Phi}0}^{\mathrm{T}} \left(\begin{bmatrix} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \\ \mathbf{B} \end{bmatrix} \mathbf{z}' - \begin{bmatrix} -\boldsymbol{\Phi}_{\hat{\rho}} \\ \mathbf{d}' \end{bmatrix}\right) \\ -\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}0}^{\mathrm{T}} \left(\begin{bmatrix} \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \\ \mathbf{B} \end{bmatrix} \dot{\mathbf{z}}' + \begin{bmatrix} \dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} \\ \dot{\mathbf{B}} \end{bmatrix} \mathbf{z}' + \begin{bmatrix} \dot{\boldsymbol{\Phi}}_{\hat{\rho}} \\ \mathbf{B}_{\hat{\rho}} \dot{\mathbf{z}} - \dot{\mathbf{z}}^{i\prime} \end{bmatrix}\right) \\ -\boldsymbol{\mu}_0^{\mathrm{T}} \left(\breve{\mathbf{M}} \ddot{\mathbf{z}}' + \breve{\mathbf{C}} \dot{\mathbf{z}}' + \left(\breve{\mathbf{M}}_{\hat{\mathbf{z}}} \ddot{\mathbf{z}} + \breve{\mathbf{K}}\right) \mathbf{z}' - \breve{\mathbf{Q}}_{\hat{\rho}} + \breve{\mathbf{M}}_{\hat{\rho}} \ddot{\mathbf{z}}\right) \end{aligned} \tag{5.124}$$

In the two previous equations, $\boldsymbol{\mu}_{\boldsymbol{\Phi}0} \in \mathbb{R}^{(m+d)\times o}$ is the adjoint variable related to the initial position equations, $\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}0} \in \mathbb{R}^{(m+d)\times o}$ is the variable related to the fulfillment of the initial velocity problem and $\boldsymbol{\mu}_0 \in \mathbb{R}^{n\times o}$ is related to the satisfaction of the dynamic penalty problem.

In (5.124), there are not unprojected velocities or accelerations, neither Lagrange multipliers, but there are 3 sets of unknown sensitivities: $\mathbf{z}'$, $\dot{\mathbf{z}}'$ and $\ddot{\mathbf{z}}'$. Furthermore, the equations used in the adjoint are all linear, so its solution does not require a numerical integration. Nullifying the terms multiplying the sensitivities of the states, considering also the expression of the complete Lagrangian gradient (5.114), the following system of adjoint equations at the initial time $t_0$ is obtained:

$$\left\{\boldsymbol{\mu}_0^{\mathrm{T}}\left(\breve{\mathbf{M}}_{\hat{\mathbf{z}}}\ddot{\mathbf{z}}+\breve{\mathbf{K}}\right)+\boldsymbol{\mu}_{\boldsymbol{\Phi}0}^{\mathrm{T}}\left(\begin{bmatrix}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\\ \mathbf{B}\end{bmatrix}\right)+\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}}\left(\begin{bmatrix}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\\ \dot{\mathbf{B}}\end{bmatrix}\right)\right\}_{\{0\}}=\mathbf{g}_{\hat{\mathbf{z}}\{0\}}+\left\{\frac{2\gamma}{\beta h}\mathbf{G}+\frac{2}{\beta h^2}\mathbf{H}\right\}_{\{1\}} \tag{5.125a}$$

$$\left\{\boldsymbol{\mu}_0^{\mathrm{T}}\left(\breve{\mathbf{C}}\right)+\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}0}^{\mathrm{T}}\left(\begin{bmatrix}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\\ \mathbf{B}\end{bmatrix}\right)\right\}_{\{0\}}=\mathbf{g}_{\hat{\dot{\mathbf{z}}}\{0\}}-2\left\{\left(1-\frac{\gamma}{\beta}\right)\mathbf{G}+\left(-\frac{1}{\beta h}\right)\mathbf{H}\right\}_{\{1\}} \tag{5.125b}$$

$$\left\{\boldsymbol{\mu}_0^{\mathrm{T}}\left(\breve{\mathbf{M}}\right)\right\}_{\{0\}}=\mathbf{g}_{\hat{\ddot{\mathbf{z}}}\{0\}}-2\left\{\left(1-\frac{\gamma}{2\beta}\right)h\mathbf{G}+\left(1-\frac{1}{2\beta}\right)\mathbf{H}\right\}_{\{1\}} \tag{5.125c}$$

Observe that the right hand sides of equations (5.125a), (5.125b) and (5.125c) are identical to (5.119a), (5.119b) and (5.119c), respectively, except for a coefficient 2 multiplying terms $\mathbf{G}$ and $\mathbf{H}$ due to the trapezoidal integration scheme used (5.114). This coefficient arises during the addition of equations and the cancellation of the terms including the sensitivities of the states at $t_0$.

The system of equations (5.125) can be easily solved sequentially: firstly, $\boldsymbol{\mu}_0$ can be determined from (5.125c) (this system is compatible determined, hence it has an unique solution); secondly, $\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}0}$ can be obtained from (5.125b). Thirdly, $\boldsymbol{\mu}_{\boldsymbol{\Phi}0}$ can be calculated from (5.125a).

It is important to note that the use of redundant constraints entails the existence of infinite valid solutions for $\boldsymbol{\mu}_{\boldsymbol{\Phi}0}$ and $\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}0}$, being one of them the minimum norm solution of the system. Since this system is solved only once per simulation, there is no need to use efficient techniques to solve it. This minimum norm solution is considered and implemented in MBSLIM for the solution of both $\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}0}$ and $\boldsymbol{\mu}_{\boldsymbol{\Phi}0}$.

Once solved the systems of adjoint equations corresponding to each time step, the gradient of the objective function can be computed with the remaining terms independently of the sensitivities of the states. The instant gradient for $t_i \in (t_0, t_F]$ can be computed as:

$$\begin{aligned}\boldsymbol{\psi}_i'=\mathbf{g}_{\hat{\boldsymbol{\rho}}}+\boldsymbol{\mu}^{\mathrm{T}}\bar{\mathbf{Q}}^{\boldsymbol{\rho}}-\boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}}-\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}^{\mathrm{T}}&\left(\boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}}\varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}}-\bar{\mathbf{P}}_{\hat{\boldsymbol{\rho}}}\left(\dot{\mathbf{z}}^*-\dot{\mathbf{z}}\right)+\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\varsigma\boldsymbol{\alpha}\dot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}}\right)\\ -\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}^{\mathrm{T}}&\left(\boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}}\varsigma\boldsymbol{\alpha}\ddot{\boldsymbol{\Phi}}-\bar{\mathbf{P}}_{\hat{\boldsymbol{\rho}}}\left(\ddot{\mathbf{z}}^*-\ddot{\mathbf{z}}\right)+\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\varsigma\boldsymbol{\alpha}\ddot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}}\right)\end{aligned} \tag{5.126}$$

At $t_0$, the instant gradient of the objective function is:

$$\boldsymbol{\psi}_0'=\mathcal{L}_0'=\mathbf{g}_{\hat{\boldsymbol{\rho}}}-\boldsymbol{\mu}_{\boldsymbol{\Phi}0}^{\mathrm{T}}\left(\begin{bmatrix}\boldsymbol{\Phi}_{\boldsymbol{\rho}}\\ -\mathbf{z}^{i\prime}\end{bmatrix}\right)-\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}0}^{\mathrm{T}}\left(\begin{bmatrix}\dot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}}\\ \mathbf{B}_{\hat{\boldsymbol{\rho}}}\dot{\mathbf{z}}-\dot{\mathbf{z}}^{i\prime}\end{bmatrix}\right)-\boldsymbol{\mu}_0^{\mathrm{T}}\left(\breve{\mathbf{M}}_{\hat{\boldsymbol{\rho}}}\ddot{\mathbf{z}}-\breve{\mathbf{Q}}_{\hat{\boldsymbol{\rho}}}\right) \tag{5.127}$$

Applying the numerical integration of the integral (5.114), the gradient of the objective function can be obtained as:

$$\boldsymbol{\psi}'=\frac{h}{2}\left(\boldsymbol{\psi}_0'+\boldsymbol{\psi}_n'\right)+h\sum_{i=1}^{n-1}\boldsymbol{\psi}_i' \tag{5.128}$$

The semi-recursive ALI3-P discrete adjoint sensitivity formulation can be generally computed by means of the following algorithm:

1. Solution of the dynamic problem and storage of the values of the states, the Lagrange multipliers and the projected and unprojected velocities and accelerations at each time step.
2. Initiation of the backwards evaluation of the adjoint variable sensitivity at $t_F$ with $\mathbf{G} = \mathbf{0}$ and $\mathbf{H} = \mathbf{0}$.
3. Determination of $\boldsymbol{\mu}_{\ddot{\boldsymbol{\Phi}}}$ from (5.119c).
4. Evaluation of $\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}}$ from (5.119b).
5. Solution of $\boldsymbol{\mu}$ and $\boldsymbol{\mu}_{\boldsymbol{\Phi}}$ form (5.119a) and (5.119d).
6. Computation of the instant objective function gradient (5.126) and integration in time by means of (5.128).
7. Calculation of terms $\mathbf{G}$ and $\mathbf{H}$ through (5.120) at the current time step.
8. Decrease of the time instant ($i = i - 1$), and repetition of the stages from 2 to 6 until $t_0$ is reached.
9. At $t_0$, computation of $\boldsymbol{\mu}_0$ from (5.125c).
10. Calculation of $\boldsymbol{\mu}_{\dot{\boldsymbol{\Phi}}0}$ from (5.125b), using a minimum norm solver.
11. Determination of $\boldsymbol{\mu}_{\boldsymbol{\Phi}0}$ from (5.125a), using a minimum norm solver.
12. Computation of the instant objective function gradient (5.127) and integration in time by means of (5.128).

Behold that most of the obstacles found in the continuous adjoint variable method applied to the semi-recursive ALI3-P formulation, such as time derivatives of mass and projection matrices, the application of a variable change, the addition of adjoint variables at $t_F$ or the complex initialization process (see section 5.4.2.1) are avoided with this method. There are no additional derivatives apart from the ones present on the forward sensitivity formulation and the initialization process at $t_F$ is reduced to use the same expressions valid for any instant of time $t_i \in (t_0, t_F]$ but with $\mathbf{G}$ and $\mathbf{H}$ equal to zero.

## 5.5 Constraint derivatives

The derivatives of the constraints vector are ubiquitous in the kinematic or dynamic analysis of any constrained multibody system. The sensitivity analysis of a constrained dynamic or kinematic problem involves the derivatives of each magnitude evaluated in the original problem, which encompasses an explosion of constraint vector derivatives. Among other magnitudes, the sensitivity analysis require first and second derivatives of the constraints vector, as it has been evidenced in the expressions of the semi-recursive ALI3-P and Matrix R sensitivity formulations.

Constraints can be expressed in terms of positions, velocities and accelerations of Cartesian coordinates or joint coordinates. For some constraint equations, such as loop-closure constraints, the definition in terms of Cartesian coordinates is more direct, and analytical derivatives of these expressions with respect to Cartesian coordinates are forthright to obtain. In this sense, the following set of dependencies will be considered in the differentiation of the constraints vector:

$$\mathbf{\Phi} = \mathbf{\Phi}\left(\mathbf{q}\left(\mathbf{z},\boldsymbol{\rho}\right),\mathbf{z},\boldsymbol{\rho}\right) \tag{5.129a}$$

$$\dot{\mathbf{\Phi}} = \dot{\mathbf{\Phi}}\left(\dot{\mathbf{q}}\left(\dot{\mathbf{z}},\mathbf{z},\boldsymbol{\rho}\right),\mathbf{q}\left(\mathbf{z},\boldsymbol{\rho}\right),\dot{\mathbf{z}},\mathbf{z},\boldsymbol{\rho}\right) \tag{5.129b}$$

$$\ddot{\mathbf{\Phi}} = \ddot{\mathbf{\Phi}}\left(\ddot{\mathbf{q}}\left(\ddot{\mathbf{z}},\dot{\mathbf{z}},\mathbf{z},\boldsymbol{\rho}\right),\dot{\mathbf{q}}\left(\dot{\mathbf{z}},\mathbf{z},\boldsymbol{\rho}\right),\mathbf{q}\left(\mathbf{z},\boldsymbol{\rho}\right),\ddot{\mathbf{z}},\dot{\mathbf{z}},\mathbf{z},\boldsymbol{\rho}\right) \tag{5.129c}$$

being $\mathbf{q}$ , $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ Cartesian positions, velocities and accelerations, respectively.

The transformation from coordinate derivatives with respect to Cartesian coordinates to joint-coordinate derivatives can be attained using the differentiation chain rule in combination with the expressions presented in section 4.5.

In the following sections, the analytical expressions of the derivatives of the constraints vector needed for the solution of the sensitivity analysis of the dynamics (and kinematics) of multibody systems are introduced.

Moreover, it is important to remark that, hereinafter, the operator $\otimes_2$ of tensor-vector or tensor-matrix products on mode 2 will be omitted in order to produce a more compact notation. If tensor-vector or tensor-matrix products are on mode 3, the operator $\otimes_3$ will be explicitly used.

## 5.5.1 Evaluation of $\mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}$

The Jacobian matrix of the constraints vector is explicitly present in every of the formulations used to solve constrained multibody dynamics. This matrix involves arithmetic operations of positions of points, vectors, angles and distances, as well as other parameters. In general, the constraints Jacobian matrix is not constant, thus the second derivative $\mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}$ exists, and has the following expression:

$$\begin{gathered}\mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}} = \left(\mathbf{\Phi}_{\hat{\mathbf{z}}}\right)_{\hat{\mathbf{z}}} = \left(\mathbf{\Phi}_{\mathbf{q}}\mathbf{q}_{\mathbf{z}} + \mathbf{\Phi}_{\mathbf{z}}\right)_{\hat{\mathbf{z}}} = \left(\mathbf{\Phi}_{\mathbf{q}}\right)_{\hat{\mathbf{z}}}\mathbf{q}_{\mathbf{z}} + \mathbf{\Phi}_{\mathbf{q}}\left(\mathbf{q}_{\mathbf{z}}\right)_{\hat{\mathbf{z}}} + \left(\mathbf{\Phi}_{\mathbf{z}}\right)_{\hat{\mathbf{z}}} = \\ \left(\mathbf{\Phi}_{\mathbf{qq}}\mathbf{q}_{\mathbf{z}} + \mathbf{\Phi}_{\mathbf{qz}}\right)\mathbf{q}_{\mathbf{z}} + \mathbf{\Phi}_{\mathbf{q}}\mathbf{q}_{\mathbf{zz}} + \mathbf{\Phi}_{\mathbf{zq}}\mathbf{q}_{\mathbf{z}} + \mathbf{\Phi}_{\mathbf{zz}} = \left(\mathbf{\Phi}_{\mathbf{qq}}\mathbf{q}_{\mathbf{z}}\right)\mathbf{q}_{\mathbf{z}} + \mathbf{\Phi}_{\mathbf{q}}\mathbf{q}_{\mathbf{zz}} + 2\mathbf{\Phi}_{\mathbf{zq}}\mathbf{q}_{\mathbf{z}} + \mathbf{\Phi}_{\mathbf{zz}}\end{gathered} \tag{5.130}$$

In general, the presence of a constraint involving relative coordinates $\mathbf{z}$ and Cartesian coordinates of points or vectors $\mathbf{q}$ is not very usual, so the term $\mathbf{\Phi}_{\mathbf{zq}}$ will be commonly zero. Observe that three new derivatives of the constraint vector appear in (5.130), involving the second partial derivative of the constraints with respect to natural coordinates $\mathbf{\Phi}_{\mathbf{qq}} \in \mathbb{R}^{m\times n_q\times n_q}$, the second partial derivative with respect to relative coordinates $\mathbf{\Phi}_{\mathbf{zz}} \in \mathbb{R}^{m\times n\times n}$ and the crossed derivatives with respect to the relative and Cartesian coordinates and $\mathbf{\Phi_{zq}} \in \mathbb{R}^{m \times n \times n_q}$ (being $n_q$ the number of natural coordinates affecting the constraints vector).

In (5.130), terms can be divided in two groups: the derivatives involving analytical expressions dependent on the type of constraints and terms involving the topology of the mechanism. The derivatives $\mathbf{\Phi_{qq}}$, $\mathbf{\Phi_{zq}}$ and $\mathbf{\Phi_{zz}}$ gather explicit dependencies on $\mathbf{q}$ and $\mathbf{z}$, so they can be directly evaluated differentiating the particular equations of each type of constraint. On the other hand, the terms $\mathbf{q_z}$ and $\mathbf{q_{zz}}$ described in sections 4.5.1 and 4.5.4 respectively, represent the variation of positions of points and vectors with respect to the relative coordinates, and they are directly related to the sequence and type of joints of the model.

The partial derivative of the Jacobian matrix with respect to any of the magnitudes contained in the relative coordinates vector are gathered in $\mathbf{\Phi_{zz}}$.

One important note with respect to equation (5.130) is that different modes of tensor products appear, as it can be observed in the following expression. This product has the commutative property, so 2 expressions are valid:

$$(\mathbf{\Phi_{qq}} \mathbf{q_z})\, \mathbf{q_z} = \mathbf{\Phi_{qq}} \otimes_2 \mathbf{q_z} \otimes_3 \mathbf{q_z} = \mathbf{\Phi_{qq}} \otimes_3 \mathbf{q_z} \otimes_2 \mathbf{q_z} \tag{5.131}$$

where the sub-index of the cross specifies the mode of the reduced tensor product. The same properties can be applied to a tensor-vector product.

## 5.5.2 Evaluation of $\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}$

The tensor $\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}$ is the term that gathers more different terms and a long concatenation of sums of products, thus it requires an important computational effort. It can be demonstrated that this tensor can be calculated through a direct differentiation of the expression of $\mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}$ given by 5.130 with respect to time, regarding that $\left(\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}\right)_{\hat{\mathbf{z}}} = \dfrac{\mathrm{d}\mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}}{\mathrm{d}t}$:

$$\begin{aligned}\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}} = \frac{\mathrm{d}}{\mathrm{d}t}\left((\mathbf{\Phi_{qq}}\mathbf{q_z})\,\mathbf{q_z} + \mathbf{\Phi_q}\mathbf{q_{zz}} + 2\mathbf{\Phi_{zq}}\mathbf{q_z} + \mathbf{\Phi_{zz}}\right) = \\ \left(\dot{\mathbf{\Phi}}_{\mathbf{qq}}\mathbf{q_z}\right)\mathbf{q_z} + (\mathbf{\Phi_{qq}}\dot{\mathbf{q}}_{\mathbf{z}})\,\mathbf{q_z} + (\mathbf{\Phi_{qq}}\mathbf{q_z})\,\dot{\mathbf{q}}_{\mathbf{z}} + \dot{\mathbf{\Phi}}_{\mathbf{q}}\mathbf{q_{zz}} + \mathbf{\Phi_q}\dot{\mathbf{q}}_{\mathbf{zz}} + 2\dot{\mathbf{\Phi}}_{\mathbf{zq}}\mathbf{q_z} + 2\mathbf{\Phi_{zq}}\dot{\mathbf{q}}_{\mathbf{z}} + \dot{\mathbf{\Phi}}_{\mathbf{zz}}\end{aligned} \tag{5.132}$$

The tensors $\dot{\mathbf{\Phi}}_{\mathbf{qq}}$, $\dot{\mathbf{\Phi}}_{\mathbf{zq}}$ and $\dot{\mathbf{\Phi}}_{\mathbf{zz}}$ can be easily calculated for a particular constraint equation, but $\dot{\mathbf{q}}_{\mathbf{zz}}$ is a general term related to the topology of the mechanism, and its calculation has been explained in section 4.5.6. The remaining terms of equation (5.132) have been introduced in previous sections.

Observe that (5.132) involves the sum of eight different tensor products, and requires eleven different terms to be computed, seven dependent on each individual constraint equation ($\dot{\mathbf{\Phi}}_{\mathbf{qq}}$, $\dot{\mathbf{\Phi}}_{\mathbf{zq}}$ and $\dot{\mathbf{\Phi}}_{\mathbf{zz}}$, $\mathbf{\Phi_{qq}}$, $\mathbf{\Phi_{zq}}$ and $\dot{\mathbf{\Phi}}_{\mathbf{q}}$, and $\mathbf{\Phi_q}$), and four dependent on the topology of the mechanism ($\dot{\mathbf{q}}_{\mathbf{zz}}$, $\mathbf{q_{zz}}$, $\dot{\mathbf{q}}_{\mathbf{z}}$ and $\mathbf{q_z}$). If this term is compared with the equivalent one in natural coordinates, which solely requires the term $\dot{\mathbf{\Phi}}_{\mathbf{qq}}$, it can be deduced that relative coordinate models find a handicap when the sensitivities of constrained systems are addressed.

Despite the large expression, four considerations must be pointed out: first, seven of the eleven terms present in the computation of $\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}$ are also needed by other terms of the multibody sensitivity analysis, so they could be reused; second, relative coordinate models usually generate a low number of constraints, so the order of the products is relative low, specially if it is compared with the size of the equivalent term of natural coordinates $\dot{\boldsymbol{\Phi}}_{\mathbf{qq}}$; third, the constraints of multibody models, such as loop-closure constraints, have very simple expressions which make differentiation really simple, and in some cases several derivatives will become even null; and fourth, the crossed dependencies on $\mathbf{q}$ and $\mathbf{z}$, though possible, are rare, so these terms are usually not required.

Furthermore, the reuse of terms and the application of sparse tensor algebra allow an efficient evaluation of this derivative.

### 5.5.3 Evaluation of $\boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}}$

Since any constant or coefficient of a constraint can be selected as a parameter, the derivative of the constraints vector with respect to it can have several forms depending on the type of the constraint and the type of the parameter. Thus, the particular equations of each type of constraint have to be differentiated with respect to the parameter selected.

For example, for a point coincidence constraint used as loop-closure constraint, its derivative with respect to a set of parameters would be:

$$\boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}} = \left(\mathbf{r}_j - \mathbf{r}_k\right)_{\hat{\boldsymbol{\rho}}} = \frac{\partial \mathbf{r}_j}{\partial \boldsymbol{\rho}} - \frac{\partial \mathbf{r}_k}{\partial \boldsymbol{\rho}} \tag{5.133}$$

If the position of any of the points depends on a parameter, the derivative will exist. In the MBSLIM implementation, global positions of points are determined by the local coordinates of different points in the multibody model. In that case, if the parameters selected are local coordinates of points, expressions of section 4.5.7 can be used to compute $\boldsymbol{\Phi}_{\hat{\boldsymbol{\rho}}}$.

### 5.5.4 Evaluation of $\boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}$

The computation of second derivatives in relative coordinate models involves, as it was patent in previous sections, a concatenation of sums of tensor and matrix products. The same circumstance is present in the assessment of $\boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}$:

$$\boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}} = \left(\boldsymbol{\Phi}_{\mathbf{q}}\mathbf{q}_{\mathbf{z}} + \boldsymbol{\Phi}_{\mathbf{z}}\right) = \boldsymbol{\Phi}_{\mathbf{q}\hat{\boldsymbol{\rho}}}\mathbf{q}_{\mathbf{z}} + \boldsymbol{\Phi}_{\mathbf{q}}\mathbf{q}_{\mathbf{z}\boldsymbol{\rho}} + \boldsymbol{\Phi}_{\mathbf{z}\hat{\boldsymbol{\rho}}} \tag{5.134}$$

Tensors $\boldsymbol{\Phi}_{\mathbf{q}\hat{\boldsymbol{\rho}}} \in \mathbb{R}^{m \times n_q \times p}$ and $\boldsymbol{\Phi}_{\mathbf{z}\hat{\boldsymbol{\rho}}} \in \mathbb{R}^{m \times n \times p}$ are obtained from the particular expressions of each type of constraint, while $\mathbf{q}_{\mathbf{z}\boldsymbol{\rho}} \in \mathbb{R}^{n_q \times n \times p}$ is related to the topology of the mechanism.

If the parameters of the system are related to the local geometry of a body, this is, they affect local coordinates, there would be a double explicit dependency in the derivatives of the constraint vector (with $\mathbf{q}$ and $\boldsymbol{\rho}$). In this case:

$$\mathbf{\Phi}_{\mathbf{q}\hat{\boldsymbol{\rho}}} = \mathbf{\Phi}_{\mathbf{qq}}\mathbf{q}_{\boldsymbol{\rho}} + \mathbf{\Phi}_{\mathbf{q}\boldsymbol{\rho}} \tag{5.135a}$$

$$\mathbf{\Phi}_{\mathbf{z}\hat{\boldsymbol{\rho}}} = \mathbf{\Phi}_{\mathbf{zq}}\mathbf{q}_{\boldsymbol{\rho}} + \mathbf{\Phi}_{\mathbf{z}\boldsymbol{\rho}} \tag{5.135b}$$

The derivative $\mathbf{q}_{\boldsymbol{\rho}}$ is a term related to the topology of the system and can be computed by means of the expressions developed in section 4.5.7.

### 5.5.5 Evaluation of $\dot{\mathbf{\Phi}}_{\hat{\boldsymbol{\rho}}}$

This derivative can be obtained applying the rule of differentiation (4.9) to the expression of the time derivative of the constraints vector (3.9):

$$\left(\dot{\mathbf{\Phi}}\right)_{\hat{\boldsymbol{\rho}}} = \left(\mathbf{\Phi}_{\hat{\mathbf{z}}}\dot{\mathbf{z}} + \mathbf{\Phi}_t\right)_{\hat{\boldsymbol{\rho}}} = \mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}\dot{\mathbf{z}} + \mathbf{\Phi}_{t\hat{\boldsymbol{\rho}}} \tag{5.136}$$

where $\mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}} \in \mathbb{R}^{m\times n\times p}$ is a tensor described in section 5.5.4, and $\mathbf{\Phi}_{t\hat{\boldsymbol{\rho}}} \in \mathbb{R}^{m\times p}$ is a matrix obtained from the particular expressions of each type of constraint.

Observe that $\mathbf{\Phi}_{t\hat{\boldsymbol{\rho}}}$ can be decomposed into:

$$\mathbf{\Phi}_{t\hat{\boldsymbol{\rho}}} = \mathbf{\Phi}_{t\mathbf{q}}\mathbf{q}_{\boldsymbol{\rho}} + \mathbf{\Phi}_{t\boldsymbol{\rho}} \tag{5.137}$$

with $\mathbf{q}_{\boldsymbol{\rho}}$ computed using the equations introduced in section 4.5.7.

### 5.5.6 Evaluation of $\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}$

The time derivative of the constraints vector can be affected by the different types of parameters determining the dynamic or kinematic behavior of a multibody system, and therefore, their derivatives determine a sensitivity analysis. The term $\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}} \in \mathbb{R}^{m\times n\times p}$ is a tensor result of the differentiation of $\dot{\mathbf{\Phi}}$ with respect to the relative coordinates and with respect with the set of parameters using the rule of differentiation (4.9). It can be expressed in terms of partial derivatives as:

$$\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}} = \left(\dot{\mathbf{\Phi}}_{\mathbf{q}}\mathbf{q}_{\mathbf{z}} + \mathbf{\Phi}_{\mathbf{q}}\dot{\mathbf{q}}_{\mathbf{z}} + \dot{\mathbf{\Phi}}_{\mathbf{z}}\right)_{\hat{\boldsymbol{\rho}}} = \dot{\mathbf{\Phi}}_{\mathbf{q}\hat{\boldsymbol{\rho}}}\mathbf{q}_{\mathbf{z}} + \mathbf{\Phi}_{\mathbf{q}\hat{\boldsymbol{\rho}}}\dot{\mathbf{q}}_{\mathbf{z}} + \dot{\mathbf{\Phi}}_{\mathbf{q}}\mathbf{q}_{\mathbf{z}\boldsymbol{\rho}} + \mathbf{\Phi}_{\mathbf{q}}\dot{\mathbf{q}}_{\mathbf{z}\boldsymbol{\rho}} + \dot{\mathbf{\Phi}}_{\mathbf{z}\hat{\boldsymbol{\rho}}} \tag{5.138}$$

It can be proved that:

$$\left(\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}\right)_{\hat{\boldsymbol{\rho}}} = \frac{\mathrm{d}\mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}}{\mathrm{d}t} \tag{5.139}$$

Accordingly, this derivative can be obtained through these two different procedures, yielding both of them equivalent expressions

Once again, the terms of (5.138) with the subscript $\hat{\boldsymbol{\rho}}$ can be further decomposed into:

$$\dot{\mathbf{\Phi}}_{\mathbf{q}\hat{\boldsymbol{\rho}}} = \dot{\mathbf{\Phi}}_{\mathbf{qq}}\mathbf{q}_{\boldsymbol{\rho}} + \mathbf{\Phi}_{\mathbf{qq}}\dot{\mathbf{q}}_{\boldsymbol{\rho}} + \dot{\mathbf{\Phi}}_{\mathbf{q}\boldsymbol{\rho}} \tag{5.140a}$$

$$\dot{\mathbf{\Phi}}_{\mathbf{z}\hat{\boldsymbol{\rho}}} = \dot{\mathbf{\Phi}}_{\mathbf{zq}}\mathbf{q}_{\boldsymbol{\rho}} + \mathbf{\Phi}_{\mathbf{zq}}\dot{\mathbf{q}}_{\boldsymbol{\rho}} + \dot{\mathbf{\Phi}}_{\mathbf{z}\boldsymbol{\rho}} \tag{5.140b}$$

with $\mathbf{q}_{\boldsymbol{\rho}}$ and $\dot{\mathbf{q}}_{\boldsymbol{\rho}}$ given by the equations introduced in sections 4.5.7 and 4.5.8.

Observe that most of the terms vanish for simple constraint equations, with the consequent reduction in the computational effort.

### 5.5.7 Evaluation of $\ddot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}}$

The derivative $\ddot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}}$ is explicitly required by the term $\mathbf{c}^{\boldsymbol{\rho}}$ in (5.5c), which is used in the sensitivity equations of kinematic and dynamic problems. This term can be obtained differentiating (5.136) with respect to time:

$$\left(\ddot{\boldsymbol{\Phi}}\right)_{\hat{\boldsymbol{\rho}}} = \frac{\mathrm{d}}{\mathrm{d}t}\left(\boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}\dot{\mathbf{z}} + \boldsymbol{\Phi}_{t\hat{\boldsymbol{\rho}}}\right) = \dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}\dot{\mathbf{z}} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}\ddot{\mathbf{z}} + \dot{\boldsymbol{\Phi}}_{t\hat{\boldsymbol{\rho}}} \tag{5.141}$$

where:

$$\dot{\boldsymbol{\Phi}}_{t\hat{\boldsymbol{\rho}}} = \dot{\boldsymbol{\Phi}}_{t\mathbf{q}}\mathbf{q}_{\boldsymbol{\rho}} + \boldsymbol{\Phi}_{t\mathbf{q}}\dot{\mathbf{q}}_{\boldsymbol{\rho}} + \dot{\boldsymbol{\Phi}}_{t\boldsymbol{\rho}} \tag{5.142}$$

with terms $\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}$ and $\boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}$ described in sections 5.5.6 and 5.5.4 respectively.

The explosion of constraint derivatives creates some difficulties for the implementation of constrained joint coordinate models, especially regarding the detection of what intermediate derivatives or terms can be eliminated from the calculation (in the case they are null). This happens, for instance, in the case of the normalization constraint of the Euler parameters (see section 3.5.1), which only depends on the relative coordinates vector, and all the other derivatives vanish and does not need to be computed. A general code could execute this dependency detection in an early stage of the simulation during the creation of the model, so during the sensitivity analysis process only the required derivatives and intermediate products are calculated. This is the philosophy followed in the implementation of these derivatives in the MBSLIM multibody library.

# Chapter 6

# Implementation of the proposed methods in MBSLIM

One of the biggest efforts during the development of the present thesis has been the implementation of the kinematics, dynamics and sensitivities described in previous chapters into a general multibody library, namely MBSLIM [5]. MBSLIM is a multipurpose multibody systems library for rigid-body simulations[1] originally based on natural coordinates and programmed in Fortran 2018, whose main capabilities are:

- Kinematics of multibody models in positions, velocities and accelerations, supporting mixed coordinates and a broad variety of constraints, including rheonomic and non-holonomic.
- Dynamics of multibody models, including different types of formulations and a wide variety of forces and constraints.
- Sensitivity analysis of kinematics.
- Optimization applied to the kinematics of a multibody system (optimum synthesis problem).
- Sensitivity analysis of the dynamics of multibody systems with different formulations.
- Optimization of the dynamic performance of a multibody model, including optimal control and design optimization.

The possibility of implementing the new relative coordinates formulations into an existing library has its advantages and drawbacks. On the one hand, all the types of forces and constraints, the definition of the models, the structure of resolution of equations and, definitely, all the capabilities already implemented can be reused without important changes. On the other hand, a full study of the library is needed to identify the structures in order to integrate the new formulations and algorithms while preserving the uniformity in the code and keeping the existing code running.

[1]MBSLIM is currently being extended to flexible multibody systems.

The main rules followed during the implementation were :

1. The definition of the models by the user shall be equivalent to the original one in order to make possible the execution of existing models with topological formulations.
2. The relative coordinate models shall be automatically generated from the information given by the user.
3. Natural coordinates models shall coexist with relative coordinate ones, in order to reuse some problems already solved with natural coordinates.

Despite that the two first conditions hinder the beginning of the implementation, the third one opens the possibility of a separation inside the code, dividing all in natural and relative structures and simplifying the immersion in the code. Furthermore, the chance of solving problems in natural coordinates allows to start the dynamics with initial positions and velocities calculated with the natural coordinates kinematic or static equilibrium problems.

## 6.1 Kinematics formulations

The first and perhaps the more challenging part of the implementation appeared during the creation of relative coordinate models. If the user gave the information of the model in the form of forces, masses, constraints and types of joints relating the motion of all the bodies with each other, the creation of the topological model would be straightforward. However, the models are described in MBSLIM primarily in terms of points and vectors, which can be shared or not between bodies and related by different constraints. The generation of a relative coordinate model from this information can be significantly difficult due to the fact that a joint can be defined differently in terms of points and vectors.

The process of building a topological model can be divided in three phases. Firstly, relations between each possible couple of bodies have to be evaluated in terms of constraints and shared points and vectors in order to determine if the relative motion between these bodies can be described with a kinematic joint. Secondly, the kinematic chains have to be detected, identifying the sequence of precedence of bodies and joints. And thirdly and last, loops in the kinematic chains have to be located and opened by the elimination of one of the joints of each loop and by the addition of a loop-closure constraint.

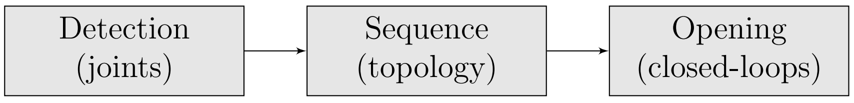


Figure 6.1: Stages of the generation of a topological model.

The opening of closed loops is programmed as an automatic operation, with a recursive algorithm which detects the closure of a chain and an automatic selection of the joint to be eliminated and the addition of the corresponding constraints. Although any type of joint can be eliminated, the priority is to clear the spherical joints, and then the rest of the types, starting from the ones with the higher number of variables. The reason of the elimination of spherical joints relies on the fact that this type of joint has 4 coordinates and 1 constraint equation related to the normalization of the Euler parameters. If it is eliminated, the number of coordinates is reduced in 4 variables, only 3 constraint equations of coincidence of points have to be added and the normalization constraint disappears.

Paying attention to the kinematics, one of the main problems is in the selection of the degrees of freedom. In MBSLIM, the DoF are selected and specified by the user among all the angles and distances defined in the model, but also among coordinates of points and vectors. Since there is a condition of minimum impact of the relative coordinate model creation in the supplied user information, the degrees of freedom must be conserved. Due to that, among other reasons, the kinematics of natural coordinates models are used to solve the initial position and velocity problems, and then a conversion from natural to relative coordinates is applied. Besides, in semi-recursive Matrix R, for example, it is necessary to solve the kinematics at each time step, and here the degrees of freedom play an essential role since they constitute the independent coordinates which are the main variables of the formulation. For this reason, the use of natural coordinates as degrees of freedom is formulated and programmed in MBSLIM for the kinematics of topological models and different dynamic formulations. Despite the slight increase in complexity, this allows a higher integration of the topological kinematics and a more direct comparison with natural coordinates models.

The two specific versions of accumulation, RTdyn0 and RTdyn1, have been included in the MBSLIM multibody library with the purpose of comparing their performance, despite entailing a duplication in part of the code. However, the reference point selected is configured to affect exclusively elemental terms, allowing the reuse of the same assembly structures and solution of the system. Furthermore, the derivatives of the constraints vector as well as the derivatives of points and vectors with respect to relative coordinates are computed with a set of expressions which are independent of the reference point selected, with the consequent reduction in the code.

The types of joints listed in chapter 2 are all implemented in MBSLIM, including all the terms and derivatives introduced in the present work. The use of modern Fortran object oriented programming features simplifies the implementation, making the addition of new types of joints more direct.

## 6.2 Forward dynamics formulations

The application of forces and masses to the kinematic structure of relative coordinate models and its study compose the dynamic analysis of the system. The selection of the reference points has here a bigger impact than in the kinematics, since the expressions of the elemental mass matrices and generalized forces vector of each body are different for the two specific accumulations being studied. On order to simplify the implementation, these differences are considered exclusively at an elemental body level, using a general structure of assembly and solution for the two accumulations.

The solution of the dynamics of relative coordinate models can be categorized as unconstrained open-loop systems and constrained models. The first group can be solved using a semi-recursive accumulation scheme as well as a fully-recursive approach. Although the two methods are basically the same (being the fully-recursive approach the triangular system of the semi-recursive expressions), their equations and their solution are completely different, involving even different terms. This is the reason for implementing two different separated codes in MBSLIM, one for each method.

Constrained systems are solved using a semi-recursive accumulation and two different formulations, one based on dependent coordinates (ALI3-P) and another on independent coordinates (Matrix R). The structure of solution of the formulations ALI3-P and Matrix R was implemented in MBSLIM previously to the onset of the present work, and it has been reused in the solution of relative coordinate models. However, there are different conditions and considerations that have to be applied solely to relative coordinates, such as the computation of the mass matrix at each time step, or the presence of a non-constant **B** matrix in relative coordinate models, which encompasses the generation of specific code in relative coordinates with its own routines of solution.

The application of constraints to fully-recursive schemes has been also studied, but its implementation is still under development, and only particular codes have been generated in the multibody library MBLSIM.

In order to obtain the low computational times presented in the numerical experiments of chapter 7, a substantial list of simplifications have been developed, being the bulk of them presented in previous chapters. However, there is a simplification that has not being previously presented, which is related to the computation of the residual of the equations of motion for a given system without considering the constraints. This simplification is described in the following subsection.

### 6.2.1 Simplified evaluation of $\mathbf{Q}^d - \mathbf{M}^d\ddot{\mathbf{z}}$

Despite the fact that a general constrained fully-recursive formulation has not being finally implemented, the study of the fully-recursive accumulations for open-loop systems has produced several improvements into the semi-recursive approach, both in constrained and unconstrained systems.

The scheme of solution usually employed to solve the dynamics from the semi-recursive perspective for the formulations studied in the present thesis is based on the Newton-Raphson method, where a linearization of the system is used to find a solution with an error lower than a given tolerance. This method requires, on the one hand, a tangent matrix of the system, and on the other, the evaluation of the residual.

In relative coordinates, the bigger effort in terms of computational time is on the calculation and assembly of the tangent matrix, involving derivatives of constraints, forces and masses. The residual, however, can be evaluated very efficiently by means of the fully-recursive scheme.

The fully-recursive expressions are obtained from the same equations of the semi-recursive approach and using the same principle of virtual power. Since there are no different assumptions taken in one or other accumulation, the final expressions have to be the same. In fact, as it has been commented before, the fully-recursive equations can be obtained by triangularization of the semi-recursive expressions. Although the two formulations are based on the same equations, the fully-recursive one is much more efficient in the computation of the residual.

Let us consider the equations of motion of an open-loop system. They can be expressed in terms of the accelerations of the relative coordinates $\ddot{\mathbf{z}}$ or depending on the linear and angular accelerations of the reference point coordinates $\dot{\mathbf{V}}$:

$$\begin{gathered}\mathbf{Q}^d - \mathbf{M}^d\ddot{\mathbf{z}} = \mathbf{R}^{v\mathrm{T}}\left(\mathbf{Q}^v - \mathbf{M}^v\dot{\mathbf{R}}^v\dot{\mathbf{z}}\right) - \left(\mathbf{R}^{v\mathrm{T}}\mathbf{M}^v\mathbf{R}^v\right)\ddot{\mathbf{z}} = \\ \mathbf{R}^{v\mathrm{T}}\left(\mathbf{Q}^v - \mathbf{M}^v\left(\dot{\mathbf{R}}^v\dot{\mathbf{z}} + \mathbf{R}^v\ddot{\mathbf{z}}\right)\right) = \mathbf{R}^{v\mathrm{T}}\left(\mathbf{Q}^v - \mathbf{M}^v\dot{\mathbf{V}}\right) = \mathbf{0}\end{gathered} \tag{6.1}$$

Using the expression dependent on the reference points, a more efficient residual evaluation is achieved.

The recursive terms $\mathbf{b}_i$ and $\dot{\mathbf{b}}_i$, used as the elemental terms for the composition of the mass matrix, generalized forces vector and even for the derivatives of the constraints vector, are expressed by positions and velocities of points and vectors (as well as other terms present on the spherical and floating joints). Thus, a forward recursive computation of the positions and velocities of the points and vectors of the mechanism is needed after any update of the relative coordinates vector. During this recursive computation, the reference point vectors in velocities ($\mathbf{V}$) and accelerations ($\dot{\mathbf{V}}$) can be calculated too with a minimum computational effort. Let us recall here the kinematic recursive relations (2.121):

$$\mathbf{V}_i = \mathbf{B}_i^v\mathbf{V}_{i-1} + \mathbf{b}_i^v\dot{\mathbf{z}}_i \tag{6.2a}$$

$$\dot{\mathbf{V}}_i = \mathbf{B}_i^v\dot{\mathbf{V}}_{i-1} + \mathbf{b}_i^v\ddot{\mathbf{z}}_i + \mathbf{d}_i^v \tag{6.2b}$$

The terms $\mathbf{B}_i^v$, $\mathbf{b}_i^v$ and $\mathbf{d}_i^v$ have to be calculated for each joint and each iteration, so the $\mathbf{V}$ and $\dot{\mathbf{V}}$ can be obtained with only a few more operations.

Recalling the dynamic assembly procedure, the expression $\mathbf{Q}^v - \mathbf{M}^v\dot{\mathbf{V}}$ can be directly computed for each joint:

$$\mathbf{Q}^v - \mathbf{M}^v\dot{\mathbf{V}} = \begin{bmatrix}\mathbf{Q}_1^v - \mathbf{M}_1^v\dot{\mathbf{V}}_1 \\ \mathbf{Q}_2^v - \mathbf{M}_2^v\dot{\mathbf{V}}_2 \\ \dots \\ \mathbf{Q}_{n_j}^v - \mathbf{M}_{n_j}^v\dot{\mathbf{V}}_{n_j}\end{bmatrix} \tag{6.3}$$

being $n_j$ the number of kinematic joints (which is equal to the number of bodies of the model).

At this point, the array $\mathbf{Q}^v - \mathbf{M}^v\dot{\mathbf{V}}$ is achieved with $\dot{\mathbf{V}}$ and the elemental mass matrix and elemental generalized forces vector of each body. Now, the accumulation of masses and forces must be done, which in a matrix notation is attained multiplying it by the transpose of $\mathbf{R}^v$, presented in (6.1). Nevertheless, this matrix has a particular structure which allows a recursive accumulation from one body to the previous one within the kinematic chain, starting by the ends of the mechanism until the base is reached. This accumulation is the basis of the fully-recursive accumulation presented in section 2.4.2, which can be summarized into:

$$\mathbf{R}_i^{\mathrm{T}}\left(\mathbf{Q} - \mathbf{M}\dot{\mathbf{V}}\right) = \mathbf{b}_i{}^{\mathrm{T}}\mathbf{H}_i \tag{6.4}$$

$$\mathbf{H}_i = \left(\mathbf{Q}_i - \mathbf{M}_i\dot{\mathbf{V}}_i\right) + \mathbf{B}_i^{v\mathrm{T}}\mathbf{H}_{i+1} \tag{6.5}$$

where $\mathbf{H}_i \in \mathbb{R}^6$ is a new term including the accumulation of the forces and torques from one body to its preceding one.

It should be indicated that this method suffers the same problem commented for the fully-recursive approach: if a direct assembly of forces over the relative coordinates is used, the previous equation have to be corrected:

$$\mathbf{R}_i^{\mathrm{T}}\left(\mathbf{Q} - \mathbf{M}\dot{\mathbf{V}}\right) = \mathbf{b}_i{}^{\mathrm{T}}\mathbf{H}_i + \mathbf{Q}_i^{\mathbf{z}} \tag{6.6}$$

$$\mathbf{H}_i = \left(\mathbf{Q}_i - \mathbf{M}_i\dot{\mathbf{V}}_i\right) + \mathbf{B}_i^{v\mathrm{T}}\mathbf{H}_{i+1} \tag{6.7}$$

being $\mathbf{Q}_i^{\mathbf{z}}$ the values of the forces and\or torques directly applied on joint $i$.

### 6.2.2 Approximate tangent matrix

As it has been previously mentioned in this document, the dynamics of some of the formulations studied is solved by means of an iterative Newton-Raphson scheme. This method is based on the decomposition of the system into a Taylor's series, where only the first derivative is used and the higher order derivatives are dismissed. Thus, the method approximates the solution of the equations of motion by means of a linearization, and since it is an iterative process, it allows also to use an approximate tangent matrix in the solution of the new set of linearized equations.

To begin with, let us recall the augmented Lagrangian part of a semi-recursive ALI3-P formulation, introduced in (3.28a):

$$\mathbf{M}^d\ddot{\mathbf{z}} + \mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\left(\boldsymbol{\lambda}^* + \boldsymbol{\alpha}\mathbf{\Phi}\right) - \mathbf{Q}^d = \mathbf{0} \tag{6.8}$$

The tangent matrix of the aforementioned system of equations, considering $\mathbf{z}$ as the main variables, can be obtained as:

$$\mathbf{T} = \mathbf{M}_{\hat{\mathbf{z}}}^d\ddot{\mathbf{z}} + \mathbf{M}^d\frac{\partial\ddot{\mathbf{z}}}{\partial\mathbf{z}} + \mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}}\left(\boldsymbol{\lambda}^* + \boldsymbol{\alpha}\mathbf{\Phi}\right) + \mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\left(\boldsymbol{\lambda}^* + \boldsymbol{\alpha}\mathbf{\Phi}_{\hat{\mathbf{z}}}\right) + \mathbf{K} + \mathbf{C}\frac{\partial\dot{\mathbf{z}}}{\partial\mathbf{z}} \tag{6.9}$$

If a Newmark's family numerical integrator, described by (4.16) and (4.18), is applied to the previous equation, expressing velocities and acceleration of the states as function of the positions of the current time step and a correction term dependent on the states of the previous step of time, the following tangent matrix is reached:

$$\mathbf{T} = \mathbf{M}^d_{\hat{\mathbf{z}}}\ddot{\mathbf{z}} + \mathbf{M}^d\frac{1}{\beta h^2} + \mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}\left(\boldsymbol{\lambda}^* + \boldsymbol{\alpha}\mathbf{\Phi}\right) + \mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\left(\boldsymbol{\lambda}^* + \boldsymbol{\alpha}\mathbf{\Phi}_{\hat{\mathbf{z}}}\right) + \mathbf{K} + \mathbf{C}\frac{\gamma}{\beta h} \tag{6.10}$$

Regarding the new factors introduced with the numerical integrator, it can be deduced that one of the most relevant terms of this tangent matrix is the mass matrix, which is multiplied by $\frac{1}{\beta h^2}$, followed by the damping matrix, scaled by $\frac{\gamma}{\beta h}$. Moreover, there is a penalty matrix $\boldsymbol{\alpha}$ as well that increases the relevance of the terms it is penalizing. However, there is a big difference among $\mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}\left(\boldsymbol{\lambda}^* + \boldsymbol{\alpha}\mathbf{\Phi}\right)$ and $\mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\left(\boldsymbol{\alpha}\mathbf{\Phi}_{\hat{\mathbf{z}}}\right)$, since $\mathbf{\Phi}$ usually have very low values, while $\mathbf{\Phi}_{\hat{\mathbf{z}}}$ is steadier and with values independent of the convergence of $\mathbf{\Phi}$. Furthermore, the stiffness matrix $\mathbf{K}$ has a very direct impact in this matrix when high stiff forces are involved in the simulation, becoming as important as the mass matrix in some simulations.

In the present development, different approximate tangent matrices have been tested, evaluating the computational effort consumed during their calculation and the number of iterations required to achieve convergence. The results of this study yielded that the term $\mathbf{M}^d_{\hat{\mathbf{z}}}\ddot{\mathbf{z}}$ has a minimum impact in the number of iterations whereas its computation is really expensive, and accordingly, it has been eliminated from the tangent matrix of the Newton-Raphson solution. The second consequence of the study is that the term $\mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}\left(\boldsymbol{\lambda}^* + \boldsymbol{\alpha}\mathbf{\Phi}\right)$ does not contribute to the convergence of the problem, and in some simulations, it could also worsen the number of iterations, which indicates that the exact tangent matrix does not always imply a faster convergence. Therefore, this term has been also extracted from the approximated tangent matrix, which can be now computed as:

$$\mathbf{T}^{ap} = \frac{1}{\beta h^2}\mathbf{M}^d + \mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\boldsymbol{\alpha}\mathbf{\Phi}_{\hat{\mathbf{z}}} + \mathbf{K} + \frac{\gamma}{\beta h}\mathbf{C} \tag{6.11}$$

Now, some of the most time demanding terms have been eliminated from the expression of the tangent matrix, with the subsequent reduction in computational time, making this formulation more competitive with respect to other formulations. However, immersing in the study of the assembly of the matrices composing the tangent matrix, a group of new simplifications can be incorporated. Observe that $\mathbf{T}^{ap}$ is now composed of one term of masses ($\frac{1}{\beta h^2}\mathbf{M}^d$), one term of constraints ($\mathbf{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\boldsymbol{\alpha}\mathbf{\Phi}_{\hat{\mathbf{z}}}$) and two terms of forces ($\mathbf{K}$ and $\frac{\gamma}{\beta h}\mathbf{C}$).

In section 4.2.2, the exact derivatives of the generalized forces vector with respect to the relative coordinates in positions and velocities have been presented for an arbitrary selection of reference points, yielding the exact stiffness and damping matrices. Regarding its composition, it could be seen that there are terms required in the computation of both matrices as well as some repetitions in the assemblies and other calculations. These redundancies can be avoided by computing both terms jointly, generating a new matrix denoted here as $\mathbf{KC}$.

Moreover, there are some terms of the computation of these matrices that have a low impact in the convergence of the solution, such as inertial terms or the derivatives of the kinematic terms used to assemble the generalized forces vector. Considering all the issues commented, the new matrix **KC** grouping the derivatives of forces can be expressed as:

$$\mathbf{KC} = \mathbf{K}^{ap} + \frac{\gamma}{\beta h}\mathbf{C}^{ap} \tag{6.12}$$

and therefore, the approximated tangent matrix could be obtained as:

$$\mathbf{T}^{ap} = \frac{1}{\beta h^2}\mathbf{M}^d + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\alpha}\boldsymbol{\Phi}_{\hat{\mathbf{z}}} + \mathbf{KC} \tag{6.13}$$

Let us recall the expression of the calculation of the generalized forces vector of each body introduced in (4.32):

$$\mathbf{Q}_i^v = \sum_{j=1}^{n_f^i} \mathbf{Q}_{i,j}^{v\ (e)} + \mathbf{Q}_i^{v(I)} \tag{6.14a}$$

$$\mathbf{Q}_{i,j}^{v\ (e)} = \begin{bmatrix} \mathbf{f}_j \\ \mathbf{n}_j^G + \left(\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_i\right)\mathbf{f}_j \end{bmatrix} \tag{6.14b}$$

$$\mathbf{Q}_i^{v(I)} = \begin{bmatrix} -m_i\tilde{\boldsymbol{\omega}}_i\left(\tilde{\boldsymbol{\omega}}_i\left(\mathbf{r}_G^i - \mathbf{r}_i\right)\right) \\ -\tilde{\boldsymbol{\omega}}_i\mathbf{J}_i^G\boldsymbol{\omega}_i - \left(\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_i\right)\left(m_i\tilde{\boldsymbol{\omega}}_i\left(\tilde{\boldsymbol{\omega}}_i\left(\mathbf{r}_G^i - \mathbf{r}_i\right)\right)\right) \end{bmatrix} \tag{6.14c}$$

being $n_f^i$ the number of external forces applied to the body $i$. Analogously, the approximated derivative of the generalized forces vector can be obtained as:

$$\mathbf{KC}_i^v = \sum_{j=1}^{n_f^i} \mathbf{KC}_{i,j}^{v\ (e)} + \mathbf{KC}_i^{v(I)} \tag{6.15a}$$

$$\mathbf{KC}_{i,j}^{v\ (e)} \begin{bmatrix} \left(\left(\mathbf{K}_e\right)_j\mathbf{q}_{\mathbf{z}} + \left(\mathbf{C}_e\right)_j\left(\dot{\mathbf{q}}_{\mathbf{z}} + \frac{\gamma}{\beta h}\mathbf{q}_{\dot{\mathbf{z}}}\right)\right) \\ \left(\tilde{\mathbf{r}}_j - \tilde{\mathbf{r}}_i\right)\left(\left(\mathbf{K}_e\right)_j\mathbf{q}_{\mathbf{z}} + \left(\mathbf{C}_e\right)_j\left(\dot{\mathbf{q}}_{\mathbf{z}} + \frac{\gamma}{\beta h}\mathbf{q}_{\dot{\mathbf{z}}}\right)\right) \end{bmatrix} \tag{6.15b}$$

$$\mathbf{KC}_i^{v(I)} = \mathbf{0} \tag{6.15c}$$

which can be regarded as an approximation of equations (4.40) and (4.49) added together. Observe also that the term related to the inertial forces is dismissed.

Once calculated the elemental approximate derivative of the generalized forces vector for each body, they have to be assembled following the same scheme of the dynamics, already presented in (4.33a) and (4.33b), but here the term corresponding to derivatives of masses and kinematic accumulative terms is disregarded:

$$\mathbf{KC}_i = \mathbf{b}_i^{v\mathrm{T}}\mathbf{KC}_i^{v\Sigma}, \tag{6.16a}$$

$$\mathbf{KC}_i^{v\Sigma} = \mathbf{KC}_i^{\ v} + \sum_{s=1}^{n_s^i}\left(\mathbf{B}_s^{v\mathrm{T}}\mathbf{KC}_s^{v\Sigma}\right) \tag{6.16b}$$

in which $n_s^i$ is the number of children of body $i$. In this case, (6.16) can be regarded as an approximation of equations (4.34a), (4.34b), (4.51a) and (4.51b) in which the least relevant terms have been neglected. This new approximated term gathering the derivatives of the generalized forces vector has given very good results in terms of convergence of the augmented Lagrangian part of the dynamics of the semi-recursive ALI3-P formulation with a substantially lower computational effort than using the exact stiffness and damping matrices. It should be remarked that the computations presented in this section are valid for Newton-Raphson iterations in the dynamics, yet the exact stiffness and damping matrices provided by equations (4.40), (4.34a), (4.34b), (4.49), (4.51a) and (4.51b) are essential for the execution of any sensitivity analysis and therefore the exact matrices have been implemented in every sensitivity analysis formulation.

## 6.3 Sensitivity analysis.

The formulations used to obtain the dynamic response of a multibody system are 4: fully-recursive approach for open-loop systems, semi-recursive accumulation for open-loop systems, semi-recursive ALI3-P for constrained systems and semi-recursive Matrix R for constrained systems. Besides, each one of the formulations listed have 2 different expressions depending on the reference point selected, one for RTdyn0 and other for RTdyn1.

The computation of the sensitivity analysis of the previous formulations have been developed using a purely analytical scheme, this is, considering all the analytical derivatives of each elemental term, constraint, mass and force, and assembling it considering the analytical derivatives of the dynamic expressions.

The variety of formulations, accumulations, types of joints, constraints and forces that could be present in a multibody model makes a general implementation of a sensitivity analysis a goal difficult to achieve. The work is hardened with the consideration of 2 different schemes of differentiation: the direct differentiation method and the adjoint variable method.

The implementation process of the analytical computation of the sensitivity analysis has been a sequential process involving different stages, all of them validated by means of a comparison with numerical differentiation. This process is outlined in Figure 6.3.

The great variety of terms, expressions, reference points and formulations does not necessarily imply an increase in the complexity of the code, but it increases the probability of including errors. In order to reduce this probability, each one of the simulation results has been tested against numerical differentiation (as it was commented) and with other multibody formulations in natural coordinates whose accuracy and reliability have been previously proved. Furthermore, numerical and symbolic differentiation (in some cases) have been also used in the detection of internal errors in different terms of each formulation, and they have worked as excellent tools for the detection and solution of anomalies in the code.

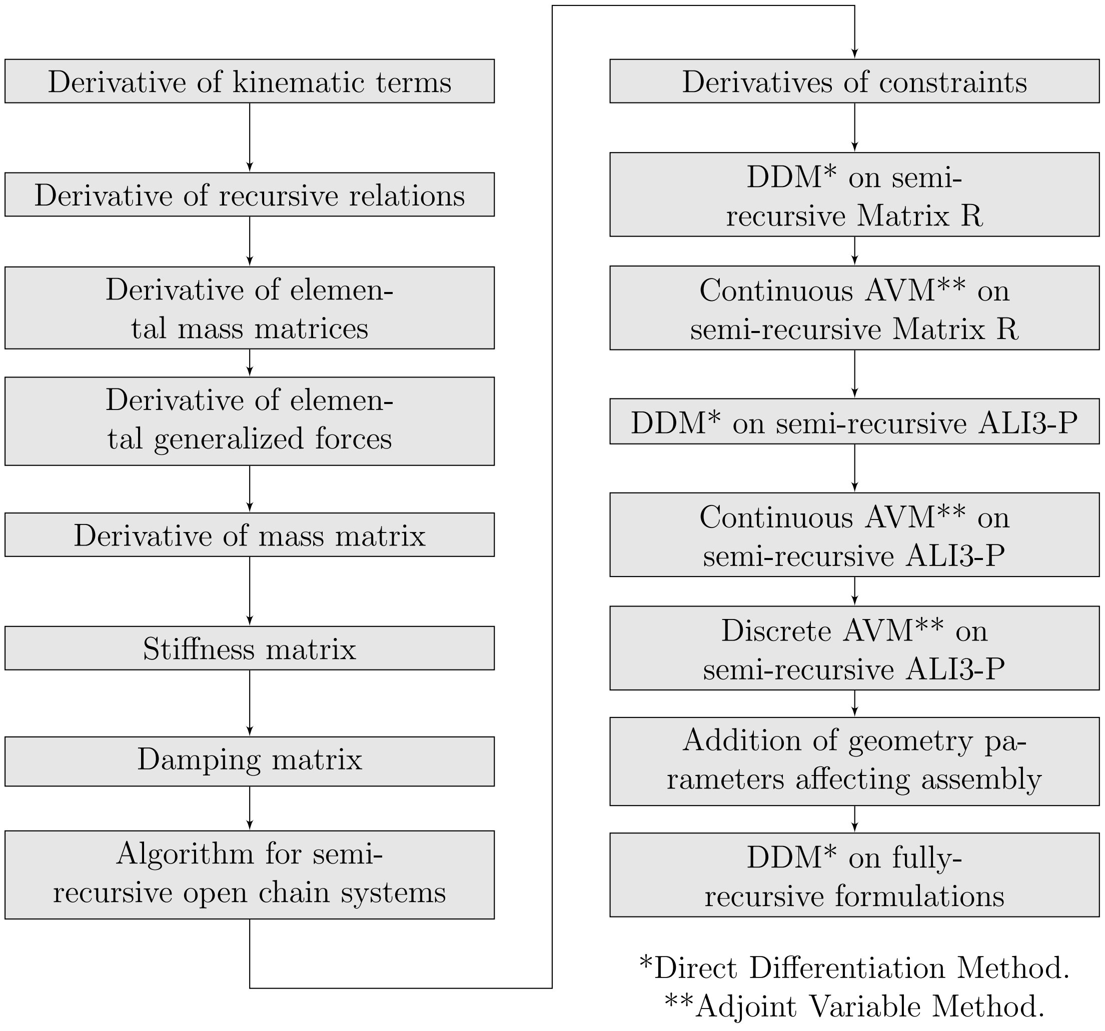

Derivative of kinematic terms
Derivative of recursive relations
Derivative of elemental mass matrices
Derivative of elemental generalized forces
Derivative of mass matrix
Stiffness matrix
Damping matrix
Algorithm for semi-recursive open chain systems
Derivatives of constraints
DDM* on semi-recursive Matrix R
Continuous AVM** on semi-recursive Matrix R
DDM* on semi-recursive ALI3-P
Continuous AVM** on semi-recursive ALI3-P
Discrete AVM** on semi-recursive ALI3-P
Addition of geometry parameters affecting assembly
DDM* on fully-recursive formulations
*Direct Differentiation Method.
**Adjoint Variable Method.

Figure 6.2: Stages of the implementation of the sensitivity analysis described in this document.

Once reached an appropriate accuracy level in the results of the sensitivity analysis of topological models, the simplification process begins. As it has been evidenced in chapters 4 and 5, the sensitivity analysis of relative coordinate models requires a significant amount of intermediate derivatives and products related to the assembly of the system matrices and the accumulation of terms. It even requires the derivatives of terms that are constant with other models, as it occurs with the mass matrix in natural coordinates.

Our first sensitivity implementations of topological formulations have been especially slow, but the use of topology to simplify the computation of derivatives in addition to the consideration of the symmetry of matrices and tensors had brought about an important increase in efficiency, reaching lower computational times than natural coordinates formulations in the numerical examples considered.

Moreover, one particular group of computations has contributed to decrease even more the computational expense: the use of symmetric and non-symmetric sparse tensors. The inclusion of sparsity in a tensor contributes to simplify its products with vectors or matrices, so it can be reused in different parts of the problem without a significant increase in the computational time. Being discarded the use of dense tensors because of its inefficiency, the other possibility, which would be to compute directly the product of a tensor by a vector, could be very efficient if the tensor appears only once per iteration, but if it appears more than once multiplying several terms, there would be a repetition in calculations and assemblies that would impair the performance of the sensitivity analysis.

Further simplifications have also been applied to the differentiation of the dynamic expressions such as the recursive computation of the residual partial derivative with respect to the parameters of the system, explained in the following subsection.

### 6.3.1 Simplified calculation of $\mathbf{Q}^d_{\hat{\boldsymbol{\rho}}} - \mathbf{M}^d_{\hat{\boldsymbol{\rho}}}\ddot{\mathbf{z}}$

The simplifications introduced in section 6.2.1 for the computation of the residual in the dynamics can be extended to obtain a simple scheme of differentiation of different terms in the framework of a joint coordinate model with respect of any parameter. These simplifications are specially helpful when local coordinates of points and vectors defining one joint are used as parameters, so a propagation of the sensitivities is needed.

The scheme presented involves tho phases: first, a differentiation of the kinematic expressions of the open-loop system, from the base of the mechanism to the ends; and second, the differentiation of the elemental masses and forces and their accumulation from the ends of the mechanism to the base.

Let us begin with the differentiation of the kinematic recursive expressions in accelerations (2.121):

$$\left(\dot{\mathbf{V}}_i\right)_{\hat{\boldsymbol{\rho}}} = \left(\mathbf{B}^v_i\right)_{\hat{\boldsymbol{\rho}}}\dot{\mathbf{V}}_{i-1} + \mathbf{B}^v_i\left(\dot{\mathbf{V}}_{i-1}\right)_{\hat{\boldsymbol{\rho}}} + \left(\mathbf{b}^v_i\right)_{\hat{\boldsymbol{\rho}}}\ddot{\mathbf{z}}_i + \left(\mathbf{d}^v_i\right)_{\hat{\boldsymbol{\rho}}} \tag{6.17}$$

The derivatives with respect to $\boldsymbol{\rho}$ of the accelerations of the reference point coordinates can be calculated, therefore, with a recursive accumulation starting from the base body to the tips of the kinematic tree. The derivatives $\left(\mathbf{B}^v_i\right)_{\hat{\boldsymbol{\rho}}}$, $\left(\mathbf{b}^v_i\right)_{\hat{\boldsymbol{\rho}}}$ and $\left(\mathbf{d}^v_i\right)_{\hat{\boldsymbol{\rho}}}$ have analytical expressions that can be obtained depending on the type of parameter selected.

It should be pointed out that the unique parameters that affect the kinematics of open-loop systems are those related to geometrical considerations, such as lengths or local coordinates of points and vectors. In constrained systems, the kinematics can also be affected by other different parameters depending on the kind of constraints defining the model.

Back to the dynamics, derivatives also have to be taken on the accumulation of forces and masses. In this case, the simplified accumulation (6.18) is directly differentiated:

$$\left(\mathbf{R}_i^{\mathrm{T}}\left(\mathbf{Q}-\mathbf{M}\dot{\mathbf{V}}\right)\right)_{\hat{\boldsymbol{\rho}}}=\left(\mathbf{b}_i^{\mathrm{T}}\right)_{\hat{\boldsymbol{\rho}}}\mathbf{H}_i+\mathbf{b}_i^{\mathrm{T}}\left(\mathbf{H}_i\right)_{\hat{\boldsymbol{\rho}}}+\left(\mathbf{Q}^{\mathbf{z}}\right)_{\hat{\boldsymbol{\rho}}} \tag{6.18}$$

$$\left(\mathbf{H}_i\right)_{\hat{\boldsymbol{\rho}}}=\left[\left(\mathbf{Q}_i\right)_{\hat{\boldsymbol{\rho}}}-\left(\mathbf{M}_i\right)_{\hat{\boldsymbol{\rho}}}\dot{\mathbf{V}}_i-\mathbf{M}_i\left(\dot{\mathbf{V}}_i\right)_{\hat{\boldsymbol{\rho}}}\right]+\left(\mathbf{B}_i^{v\mathrm{T}}\right)_{\hat{\boldsymbol{\rho}}}\mathbf{H}_{i+1}+\mathbf{B}_i^{v\mathrm{T}}\left(\mathbf{H}_{i+1}\right)_{\hat{\boldsymbol{\rho}}} \tag{6.19}$$

Observe that these expressions use only once the derivative of each term, which means that the elemental mass matrix and elemental generalized forces vector for each body have to be differentiated only once per iteration. This expression generates the easiest and faster computation of the term $\mathbf{Q}^d_{\hat{\boldsymbol{\rho}}}-\mathbf{M}^d_{\hat{\boldsymbol{\rho}}}\ddot{\mathbf{z}}$, which can be used in any of the semi-recursive formulations presented, both for open-loop and closed-loop systems.

## 6.4 Design optimization and optimal control of multibody systems

The purpose of the present thesis was originally the development of a set of fast and accurate sensitivity analysis computations which could be used in the design optimization or optimal control problems. With the advance of the work, the research lines had been zeroing in on the extension and generalization of relative coordinate formulations and sensitivity computations, putting aside the investigation into optimization applications. However, a study of these optimization problems has been performed, particularly in what concerns to their definition, implementation and practical considerations.

Optimization problems can be classified considering its nature in five categories, according to [129]: geometrical, kinematic, dynamical, multidisciplinary and topology optimization. In the present work, all the efforts have been focused in dynamic optimization, but some kinematic optimization problems related to the synthesis of different mechanisms have been also worked on.

Dynamical optimization problems can be classified as well in accordance with their purpose and the type of parameters selected, leading to optimal control and optimal design problems. Optimal control is focused on the optimization of a series of forces, torques or constraints actuating on the system over time, in order to achieve an expected performance. On the other hand, the aim of design optimization is in the determination of the values of a set of design parameters which optimizes an objective function. The difference here is patent: optimal control is referred to *external* parameters, this is, it does not affect the mechanism, but the environment conditions, whereas the optimal design considers *internal* parameters, referred to the structure, geometry, inertia or masses of the mechanism, and which directly affect the mechanism but not the environment. In MBSLIM, both optimization problems have been implemented considering the same scheme, since both use the same definition of objective function and gradient.

During the development of the present work, three approaches have been used in order to link third-party optimization algorithms with the multibody models solved with MBSLIM:

1. **Modified L-BFGS-B:** the origin of this algorithm can be dated in 1989, when Liu and Nocedal presented in [130] a new method called L-BFGS, based on a limited memory quasi-Newton method for large scale optimizations. In 1997, Zhu et al. [131] extended the algorithm so as to support simple boundary constraints in the optimization, leading to the so called L-BFGS-B algorithm. The latest revision of this code has been made by Morales and Nocedal [132] in 2011, in which the previous code was modified in order to achieve faster convergence and to eliminate the addition of errors of some compilers.

   The algorithm is programmed in Fortran 77, and therefore, its integration in a multibody library written in Fortran 2018 as MBSLIM is almost straightforward. One of the greatest advantages of this code is that it can be executed using directly the same multibody library in which the dynamics and sensitivity analysis are computed, without the need of resorting to external libraries. This simplifies as well the debugging process inherent to any program implementation.

2. **Optimization Toolbox of Matlab:** Matlab offers a wide range of possibilities related to the optimization of a function, involving constrained or unconstrained optimizations, scalar or vector objective functions, etc. In the present work, the following features of the Optimization Toolbox of Matlab have been used:

   - $fminunc$: it is a routine used for unconstrained optimization problems involving scalar objective functions.
   - $fmincon$: it supports the minimization of scalar functions subjected to a set of constraints of any type, i.e. boundary conditions, linear and nonlinear constraints. The definition of objective functions in MBSLIM as arrays allows to incorporate there any nonlinear constraint, so the analytical computation of their gradient can be directly executed without requiring additional code.
   - *multistart*: one of the biggest problems of any optimization, and especially in gradient-based optimization, is that any optimization algorithm seeks a local optimum, but not a global one. The local minimum obtained strongly depends on the initial values of the set of parameters. In order to minimize the effect of these values, which are usually approximately or arbitrarily selected, Maltlab offers the *multistart* routine, which allows to initiate the same optimization with different sets of initial values of the parameters.
   - *minimax*: multi-objective optimization can be focused from different perspectives depending on what is its desired result. One possibility could be to use a set of weight factors indicating the degree of minimization sought for each component of the objective function. An analog result can be obtained by composing a scalar objective function as the weighted addition of the components of the objective function array, which allows to use scalar optimization routines such as $fminunc$ or $fmincon$. Other possibility,

contemplated by the *minimax* routine, consists in minimize the maximum value of the components of the objective function array. This technique is also known as worst-case optimization.

The user code, containing the multibody model, maneuver and objective function, has been linked to the optimization routines of Matlab by means of .MEX files. In this sense, the debugging and detection of errors become more entangled. On the other hand, the possibility of having all the mathematical tools of Matlab and all its optimization routines are worth it.

3. **Python:** the line of investigation in Python is more recent, and only a few optimization tests have been executed in this language. Python is a programming language whose relevance in many fields of knowledge has exploded in the last few years, partially thanks to its broad variety of libraries. In what refers to the particular implementation, the user code in Fortran is connected with the code in Python by means of an additional file programmed in C++, with an equivalent role to the .MEX file in Matlab interfaces. As it has been commented, only a few tests have been executed in Python, being the optimization algorithm the modified L-BFGS-B presented above, with which the results for both implementations (Fortran and Python) of this optimizer can be tested. The expectations with the Fortran-Python interface is to widen the range of optimization algorithms by means of diverse Python libraries and to take advantage of all the features of Python.

## 6.5 Software integration

One of the objectives of the implementation of the new code related to relative coordinate models in the multibody library MBSLIM has been to integrate it with all the calculations of the preexisting natural coordinates models in a way that both codes do not interfere but share routines, schemes or algorithms. The goal has been achieved at different levels:

- **Unique user definition:** there is an unique definition of the model by the user, from which the natural and relative coordinate models are generated. All the user information is translated and included in those models, being both completely equivalent. At the moment, the natural coordinate model is created unconditionally, while relative coordinate models are created only if the formulation selected is based on these coordinates.

- **Shared routines:** the great variety of constraints and forces defined in terms of natural coordinates can be used in relative coordinate models without significant modifications. This entails a great advantage with respect to new codes, since all the features already programmed, tested and optimized in natural coordinates can be then used in joint coordinates, and accordingly, high complex models with complex forces or constraints can be then executed without redefining their expressions. Among all the constraints supported, there are dot products, angle definitions, distance definitions, cross products, guidance constraints, coincidence of points or vectors or linear combination constraints. In what refers to forces, there can be found spring forces, contact forces, different models of tire forces, other frictional forces and other user-defined actuation forces. All of these are fully supported by natural and relative coordinate models.

- **Reuse of natural coordinates problems:** the integration of natural and relative coordinate models is such that different problems solved in a set of coordinates can be reused on the other model. For instance, the initial position and velocity problem in natural coordinates is reused in relative coordinate models, being required an additional layer of calculation of the relative coordinates in positions and velocities from the values of positions and velocities of the points and vectors of the model. This layer is needed as long as the values of the vector of initial dependent relative coordinates are not specified by the user. The translation can be executed before the solution of the initial position and velocity problems, and then these problems can be solved in relative coordinates. The other possibility is solving the problem in natural coordinates and then translating the converged variables to joint coordinates. Both approaches have been implemented in MBSLIM. Other problem which can be reused from natural coordinates is the static equilibrium problem (also solved in relative coordinates). In general, the coexistence of both models would allow even to solve different parts of the dynamic or sensitivity problem with different models, even though this path has not been explored yet.

- **Numerical integration:** other part of the code that is completely shared between any of the multibody models supported by MBSLIM is related to the numerical integration of the equations of motion. The same routines of integration, such as a Newmark's family integrator, HHT integrator(Hilbert-Hughes-Taylor) or generalized-$\alpha$ integrator, are used by all the models and formulations.

There is still an incomplete line of development in MBSLIM related to the integration of natural and relative coordinate models. In future modifications, it is expected to pack all the structures and terms of every model into a bigger structure which allows to reuse the same routines of solution of constrained problems, such as the ALI3-P or Matrix R schemes. The differences of accumulations, non-constant mass matrices and other particular computations make this process unreachable by the moment, but this task is intended to be accomplished in the near future.

# Chapter 7

# Numerical experiments

The theoretical developments presented in previous chapters are tested here with different multibody systems, starting from the simplest ones such as a five-bar mechanism, to the more realistic industrial systems such as a vehicle. In the present chapter, dynamic formulations of multibody systems and their sensitivity analysis are evaluated in terms of accuracy and computational time. The kinematics and its sensitivities are not considered separately in this chapter, but they are regarded as part of the dynamic analysis, appearing implicitly in problems such as the initial position and velocity problem, or during the evaluation of semi-recursive Matrix R formulations.

Numerical experiments seek to shed light on the excellent behavior of the relative coordinates formulations described in this work compared with the equivalent formulations in natural coordinates. All the simulations and analyses have been executed with an Intel Core i7-8700 CPU at 3.20GHz in a single thread, which means that no parallelization have been considered. The computation in a single thread offers a more direct comparison between the computational effort required to solve relative and natural coordinates models, since the computational time is directly related to the number of instructions executed (as well to the accesses to memory). All the code has been generated in Fortran using the latest features allowed by the standard Fortran 2018, with the Fortran Intel Parallel Studio XE 2018 as compiler on a Windows 10 operating system.

The multibody models and maneuvers used to test the dynamic and sensitivity formulations proposed are listed in increasing complexity. Each numerical experiment is structured in four parts: first, the description of the model is presented along with the properties of the maneuver, the sensitivity analysis objective function and the set of parameters; second, the results of the dynamic simulation are displayed by means of tables and graphics, indicating computational times and the evolution of a set of variables over time in order to compare results of all the formulations tested; third, the results of the sensitivity analyses are displayed and compared among different relative and natural coordinates formulations, using direct and adjoint approaches; and fourth, the results of an optimization problem involving the sensitivity analysis of the dynamics are presented (only for the buggy vehicle and bicycle).

## 7.1 Five-bar mechanism

The first numerical experiment consists in the simulation of a five-bar linkage. The mechanism considered has been traditionally used as benchmark problem to test sensitivity results, including penalty methods [109], Matrix R methods (also referred as Maggi's formulation) [107,133], classical index-1 and index-3 Lagrange formulations [2] and even the analytical sensitivity analysis of ALI3-P formulations in its direct [3] and adjoint versions [4].

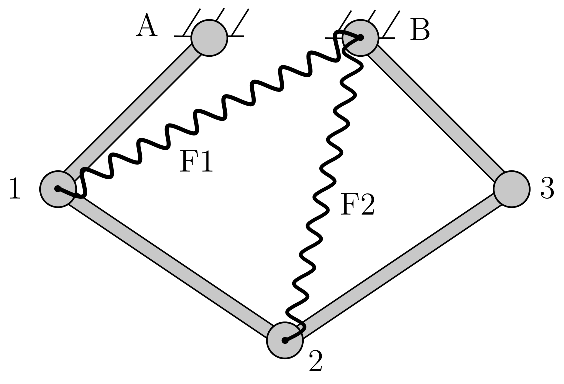


Figure 7.1: Five-bar mechanism.

### 7.1.1 Multibody model

The five-bar linkage is a mechanism composed of 4 movable bars linked to a fifth one fixed to the ground. The bars are linked by means of five revolute joints with parallel axes conforming a single closed chain. Looking at Figure 7.1, the revolute joints are located at points A, 1, 2, 3 and B. The lengths of the bars are:

$$L_{A1} = \sqrt{2}\,m \tag{7.1}$$
$$L_{12} = \frac{\sqrt{13}}{2}\,m \tag{7.2}$$
$$L_{23} = \frac{\sqrt{13}}{2}\,m \tag{7.3}$$
$$L_{3B} = \sqrt{2}\,m \tag{7.4}$$
$$L_{AB} = 1\,m \tag{7.5}$$

Points A and B are fixed to the ground, being their position:

$$\mathbf{r}_A = \begin{bmatrix}0 & 0 & 0\end{bmatrix}^{\mathrm{T}} \tag{7.6}$$
$$\mathbf{r}_B = \begin{bmatrix}1 & 0 & 0\end{bmatrix}^{\mathrm{T}} \tag{7.7}$$

The rest of the points are initially placed at:

$$\mathbf{r}_1 = \begin{bmatrix}-1 & -1 & 0\end{bmatrix}^{\mathrm{T}} \tag{7.8}$$

$$\mathbf{r}_2 = \begin{bmatrix}0.5 & -2 & 0\end{bmatrix}^{\mathrm{T}} \tag{7.9}$$

$$\mathbf{r}_3 = \begin{bmatrix}2 & -1 & 0\end{bmatrix}^{\mathrm{T}} \tag{7.10}$$

The mass of the bodies is assumed to be uniformly distributed with respect to inertia considerations. The mass of each bar is:

$$m_{A1} = 1\,kg \tag{7.11}$$

$$m_{12} = 1.5\,kg \tag{7.12}$$

$$m_{23} = 1.5\,kg \tag{7.13}$$

$$m_{3B} = 1\,kg \tag{7.14}$$

The CoM is placed in the geometrical center of each bar, matching the middle point of the line linking the tips of each bar where the revolte joints are placed. The forces acting over the mechanism are gravitational forces along with two spring-damper forces, one between points B and 1, and the other one between points B and 2. The stiffness and damping coefficients as well as the natural length characterizing the spring-damper forces are described in Table 7.1.

| Force | Stiffness $(N/m)$ | Damping $(Ns/m)$ | Natural length $(m)$ |
|---|---|---|---|
| Spring 1 | 100 | 0 | $\sqrt{5}$ |
| Spring 2 | 100 | 0 | $\sqrt{17}/2$ |

Table 7.1: Coefficients of spring-damper forces acting on the five-bar linkage

The two degrees of freedom considered for the simulation are the angles of the joints located at points A and 1. These angles are defined counterclockwise, considering as angle “0” the position where the two bars defining the joint are aligned and with the same orientation. The initial values of these angles are:

$$\alpha_1 = -\frac{3\pi}{4}\,rad \tag{7.15}$$

$$\alpha_2 = \frac{\pi}{4} + \arctan(1.5)\,rad \tag{7.16}$$

For the sensitivity analysis, the following array of objective functions is considered:

$$\boldsymbol{\psi} = \begin{bmatrix}\psi^1 \\ \psi^2 \\ \psi^3\end{bmatrix} \tag{7.17}$$

with

$$\psi^1 = \int_{t_0}^{t_F} (\mathbf{r}_2 - \mathbf{r}_{20})^{\mathrm{T}} (\mathbf{r}_2 - \mathbf{r}_{20})\,\mathrm{d}t \tag{7.18a}$$

$$\psi^2 = \int_{t_0}^{t_F} \dot{\mathbf{r}}_2^{\mathrm{T}} \dot{\mathbf{r}}_2 \mathrm{d}t \tag{7.18b}$$

$$\psi^3 = \int_{t_0}^{t_F} \ddot{\mathbf{r}}_2^{\mathrm{T}} \ddot{\mathbf{r}}_2 \mathrm{d}t \tag{7.18c}$$

In expressions (7.18), $\mathbf{r}_2$ is the position of the point identified as 2 in Figure 7.1, whereas $\dot{\mathbf{r}}_2$ and $\ddot{\mathbf{r}}_2$ are its velocity and acceleration. The term $\mathbf{r}_{20}$ represents the position of point 2 at the initial instant of time, being equal to:

$$\mathbf{r}_{20} = \begin{bmatrix} 0.5 & -2.0 & 0.0 \end{bmatrix}^{\mathrm{T}} \tag{7.19}$$

The group of parameters considered for the sensitivity analysis are the natural length of the springs, along with the mass of bar A1 and the local component $X$ of the position of its CoM. In addition, the length of the bar A1 is considered as part of the parameters of the system with the purpose of proving the calculation of the sensitivity analysis for a parameter that affects the recursive accumulation process of relative coordinate models.

$$\boldsymbol{\rho} = \begin{bmatrix} L_{s1} & L_{s2} & m_{A1} & r^G & L_{A1} \end{bmatrix}^{\mathrm{T}} \tag{7.20}$$

in which $r^G$ constitutes a simplified notation of $\left(\bar{\mathbf{r}}_{A1}^G\right)_x$.

The simulation tested consists in a 5 seconds motion of the mechanism subjected to gravitational and spring forces. The sensitivity analysis is applied to the dynamic response under these conditions.

### 7.1.2 Numerical results: dynamics

The dynamic formulations tested in this example are the semi-recursive ALI3-P and Matrix R formulations for RTdyn0 and RTdyn1 accumulations. The response of Matrix R in natural coordinates is used as reference in order to test the validity of the previous dynamic formulations.

| Formulation [1] | R0-ALI3-P | R1-ALI3-P | R0-MatrixR | R1-MatrixR |
|---|---|---|---|---|
| Tolerance | $10^{-10}$ | $10^{-10}$ | $10^{-10}$ | $10^{-10}$ |
| Penalty factor $\alpha$ | $10^9$ | $10^9$ | - | - |
| Projection matrix | $\mathbf{M}^d$ | $\mathbf{M}^d$ | - | - |
| Proj. scaling factor $\varsigma$ | 1.0 | 1.0 | - | - |
| Projection type | Non-iterative | Non-iterative | - | - |

[1]R0: RTdyn0; R1: RTdyn1.

Table 7.2: Configuration parameters for each formulation.

The dynamic simulation analyzes the motion of the mechanism subjected to the gravitational and spring-damper forces described in the previous section during 5 seconds, with a time step of 1 millisecond. Each formulation is configured with the

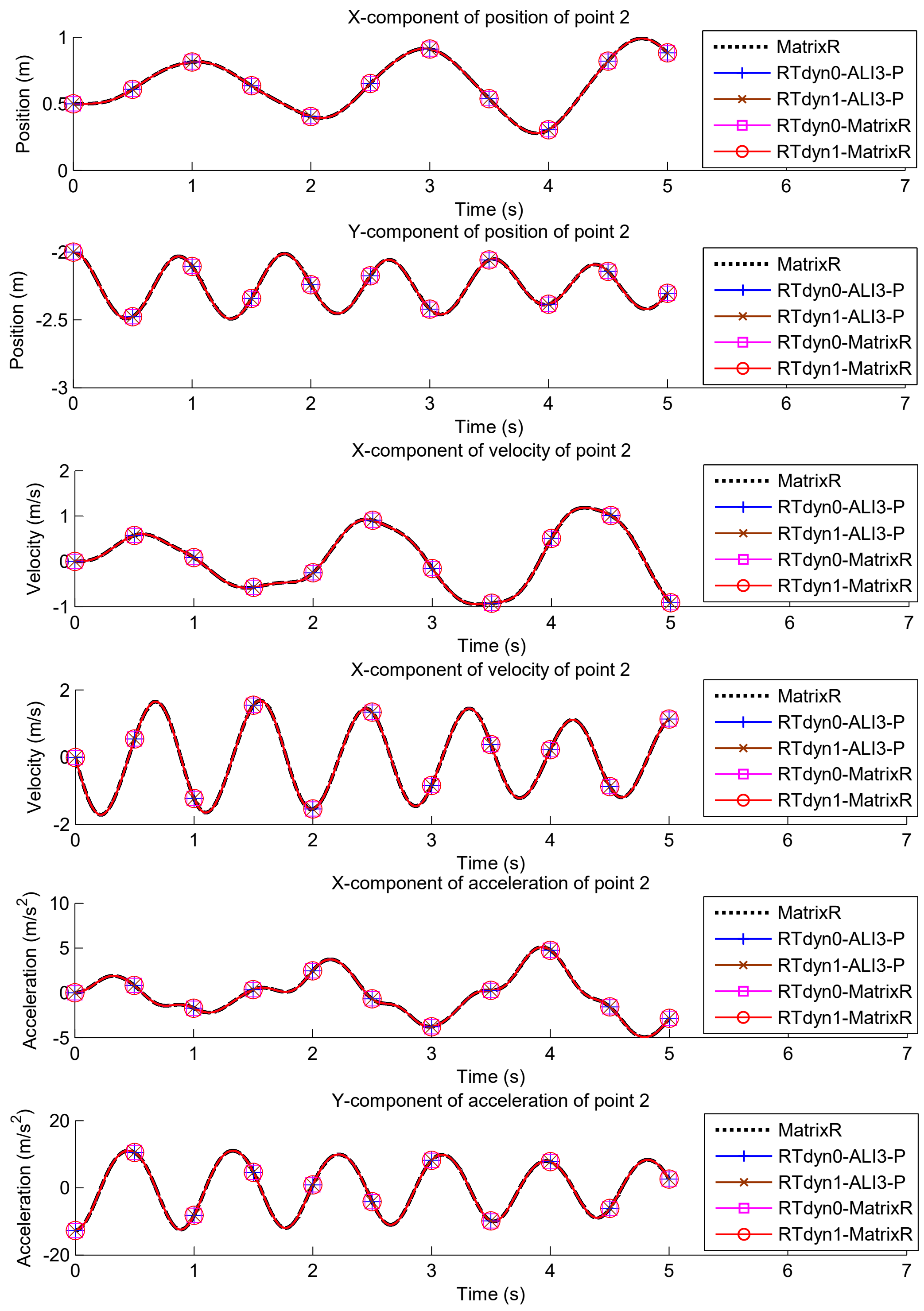


Figure 7.2: Position, velocity and acceleration of point 2 of the five-bar mechanism.

set of parameters specified in Table 7.2. The value of the penalty factor in ALI3-P formulations is selected arbitrarily from a range that usually provides a low number of iterations and low numerical errors ( $\alpha \in [10^6, 10^{12}]$). The scaling factor $\varsigma$ has been selected in such a way that the dissipation effect of the mass matrix projections (proved by García Orden and Dopico in [127]) does not affect the dynamics of the system. Moreover, the tolerance is referred to the fulfillment of an error criterion, which in this case is the 2-norm of the increment in the main variables delivered by the Newton-Raphson iteration (in both ALI3-P and Matrix R formulations).

In Figure 7.2, the position, velocity and acceleration of point 2 over time are displayed for the four formulations in relative coordinates as well as for the reference formulation (labeled as MatrixR). This figure evidences the good behavior in terms of accuracy of all the formulations presented, despite the differences in models (natural and relative coordinates), in the set of reference points (RTdyn0 and RTdyn1) and in the type of equations of motion (Matrix R and ALI3-P).

In order to compare times among formulations, the simulation time is incremented form 5 seconds to 50 seconds, with the mechanism subjected to the same forces. The computational times of the 50-seconds simulation are displayed in Table 7.3, comparing each one of the formulations being tested with the equivalent formulation in natural coordinates.

| Formulation | CPU time | CPU time natural | Ratio *nat/rel* |
|---|---|---|---|
| RTdyn0 ALI3-P | 1.906 | 2.391 | 1.255 |
| RTdyn1 ALI3-P | 1.813 | 2.391 | 1.319 |
| RTdyn0 MatrixR | 2.406 | 3.203 | 1.331 |
| RTdyn1 MatrixR | 2.250 | 3.203 | 1.424 |

Table 7.3: CPU time of relative coordinates formulations compared with the equivalent formulation in natural coordinates.

Despite the low number of bodies, forces and constraints of the mechanism, relative coordinate models show better performance in terms of computational time than the equivalent formulations in natural coordinates. This was not expected to happen for small problems like this, but the simplicity of the joint coordinate model, composed of 4 revolute joints, and the optimized code generated for relative coordinate models in MBSLIM are responsible for this reduction in CPU time.

### 7.1.3 Numerical results: sensitivity analysis

The sensitivity analysis of the dynamics is accomplished applying the following sensitivity formulations developed in this work:

- RTdyn0 ALI3-P: forward, continuous adjoint and discrete adjoint sensitivity formulations.
- RTdyn1 ALI3-P: forward, continuous adjoint and discrete adjoint sensitivity formulations.

- RTdyn0 Matrix R: forward and continuous adjoint sensitivity formulations.
- RTdyn1 Matrix R: forward and continuous adjoint sensitivity formulations.

The reference sensitivities are obtained from the natural coordinate Matrix R forward and adjoint sensitivity formulations, which have been implemented and tested in MBSLIM prior to the inception of the present work. The accuracy of the reference formulations has been also tested against finite differences, but since the evaluation of the reference formulation is not the aim of this numerical simulation, hereinafter the analytical reference solutions of Matrix R in natural coordinates will be considered as the correct solutions.

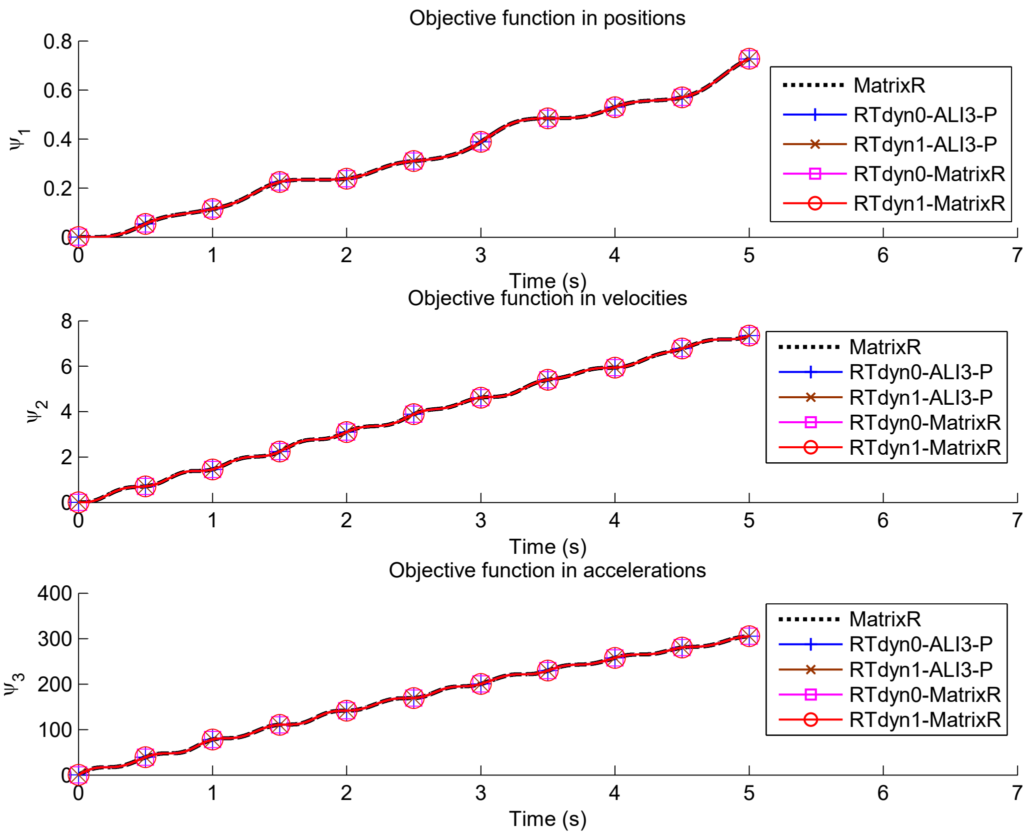


Figure 7.3: Five-bar objective functions integrated forward in time.

First of all, the results of each objective function are displayed in Figures 7.3 and 7.4, integrating forward and backward in time respectively. The reason of these two numerical time integrations relies on the fact that the same scheme of integration is used for the objective function and its gradient. By integrating the objective function forward in the direct sensitivity and backward in the adjoint sensitivity approaches, the objective function is evaluated only once per time step, and no repeated calculations or storage of the values of the objective function have to be done. These figures show an excellent convergence for all the dynamic relative coordinates formulations.

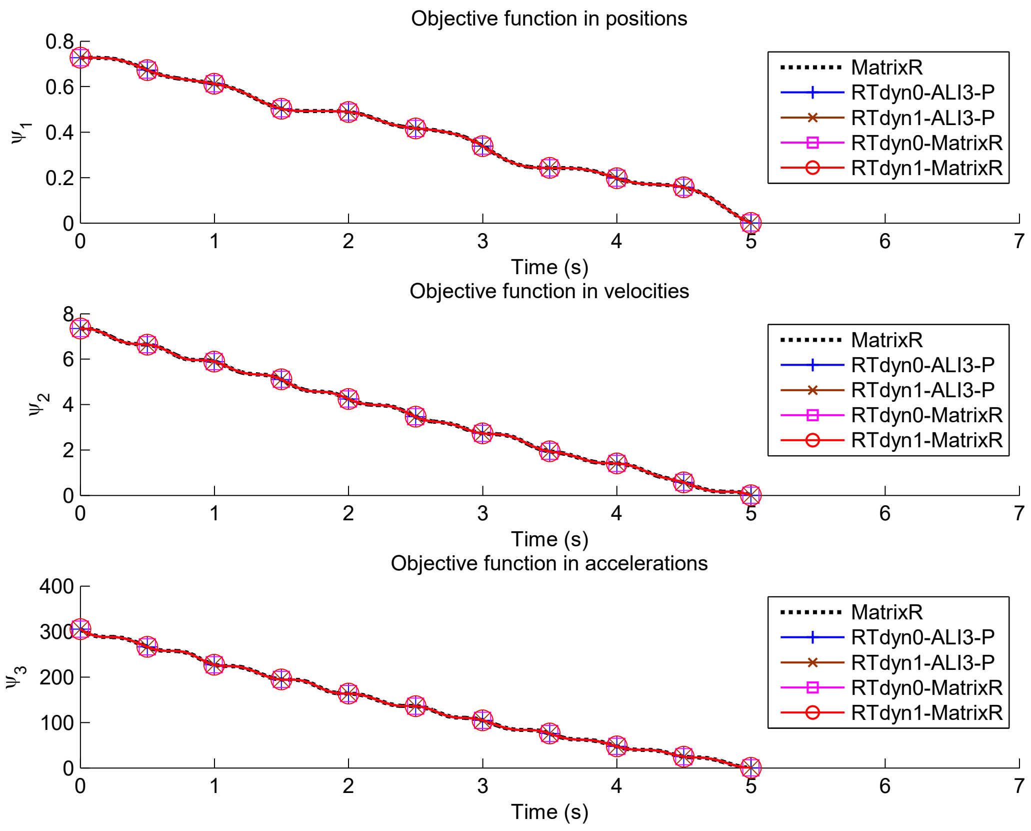


Figure 7.4: Five-bar objective functions integrated backward in time.

Now, let us begin with the sensitivity analysis of the objective function (7.17). For the sake of clearness, results are divided according to each one of the scalar objective functions conforming the array of objective functions. Moreover, the results using forward sensitivity formulations are separated from the ones using adjoint sensitivity formulations, both continuous and discrete, in order to clarify the graphics and to allow a more direct comparison among formulations.

Figure 7.5 portrays the coincidence of the results of the Matrix R forward sensitivity formulation in natural coordinates with the solution of the semi-recursive sensitivity formulations for the the first objective function (7.18a). Figure 7.6 makes apparent the convergence of the semi-recursive forward sensitivity formulations for the objective function (7.18b). The same good behavior in terms of accuracy can be observed in Figure 7.7, where the results of the semi-recursive forward sensitivity formulations (chapters 5.2.1 and 5.4.1) for the objective function (7.18c) are presented.

Figures 7.8, 7.9 and 7.10 display the results of the sensitivity analysis of the three components of the objective function array using all the semi-recursive adjoint sensitivity formulations considered in this work (chapter 5.2.2 and 5.4.2). It should be pointed out that two different semi-recursive ALI3-P adjoint formulations have been considered in this case, i.e. the discrete and continuous adjoint sensitivity formulations

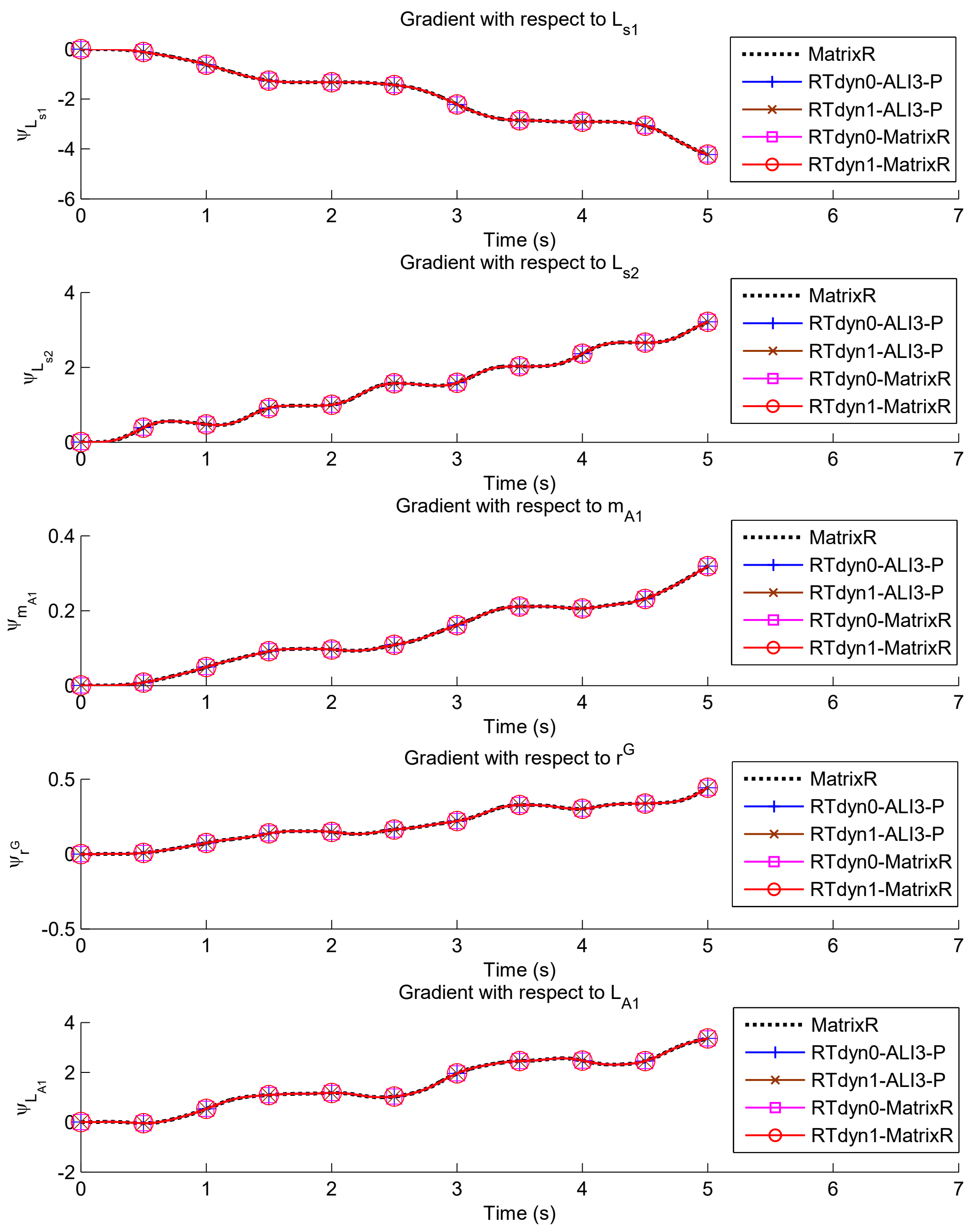


Figure 7.5: Sensitivity analysis of the objective function $\Psi^1$ using the DDM.

presented in chapter 5. Looking closer at the figures, an apparent slight divergence can be observed in the sensitivities with respect to the length of body A1 when ALI3-

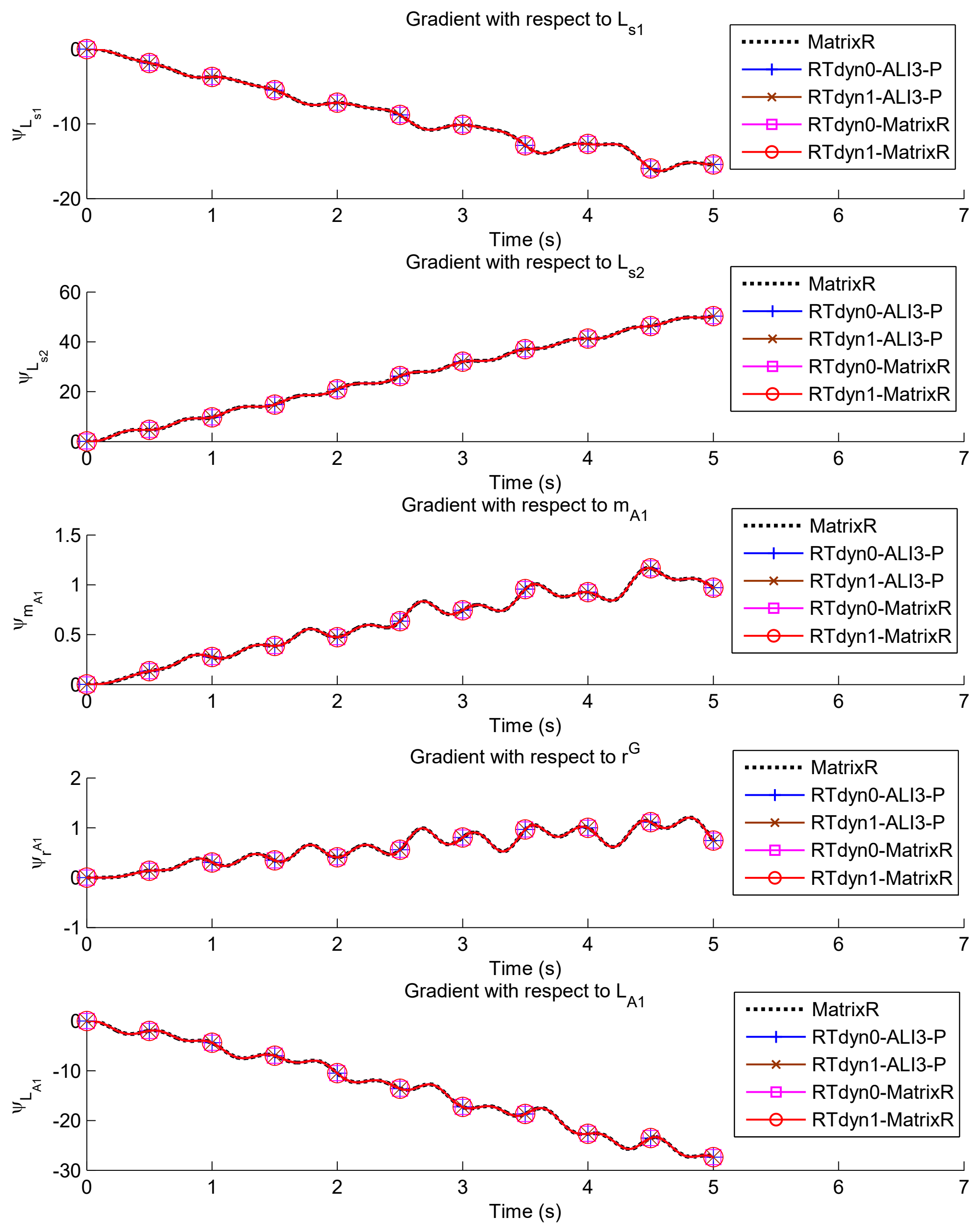


Figure 7.6: Sensitivity analysis of the objective function $\Psi^2$ using the DDM.

P and Matrix R formulations are compared. The divergence in the evolution of this component of the gradient is not an error or ill-conditioning of the ALI3-P problem,

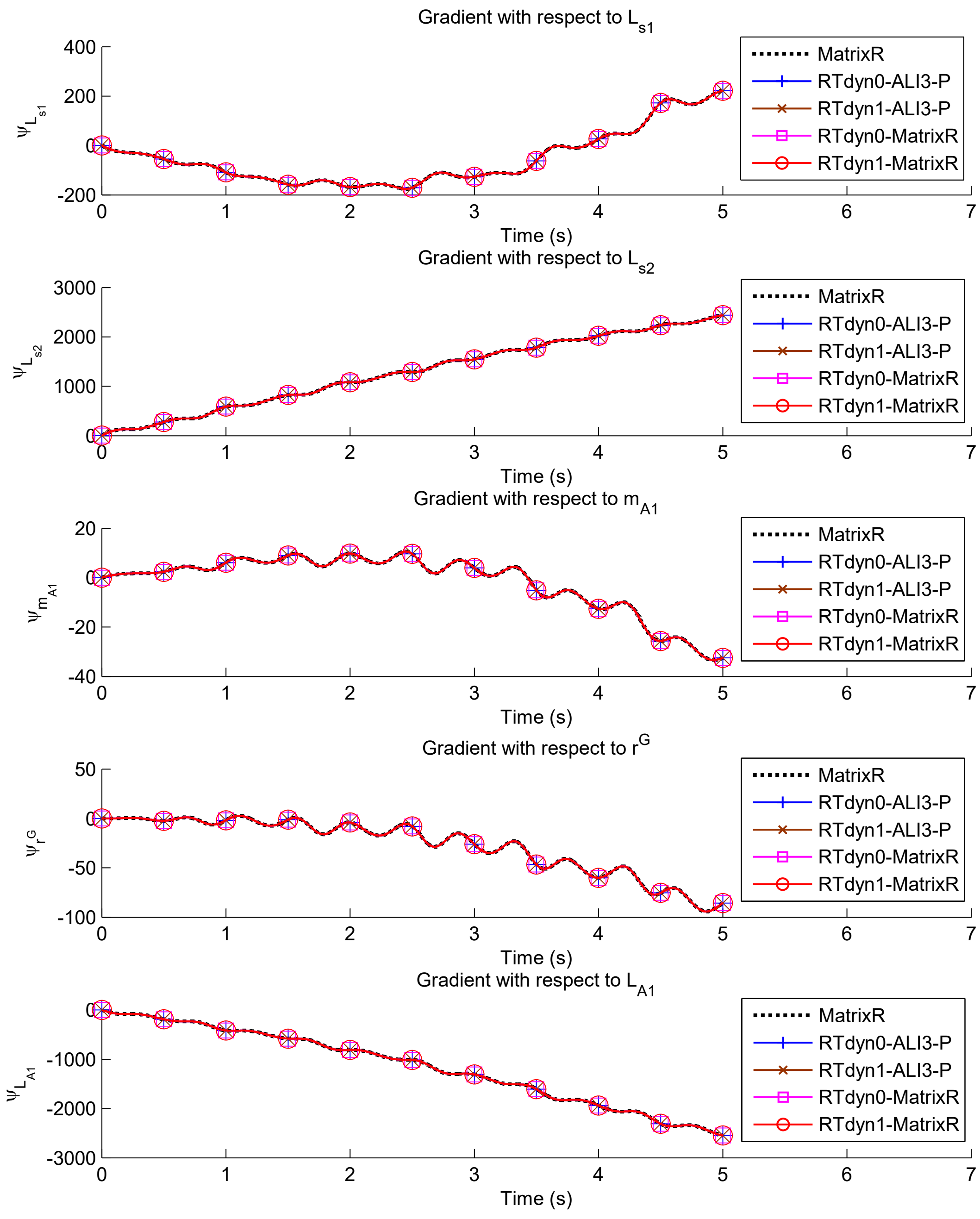


Figure 7.7: Sensitivity analysis of the objective function $\Psi^3$ using the DDM.

but it arises from the different adjoint schemes and its numerical integration. In this case, it should be reminded that the continuous adjoint methods encompass a group of

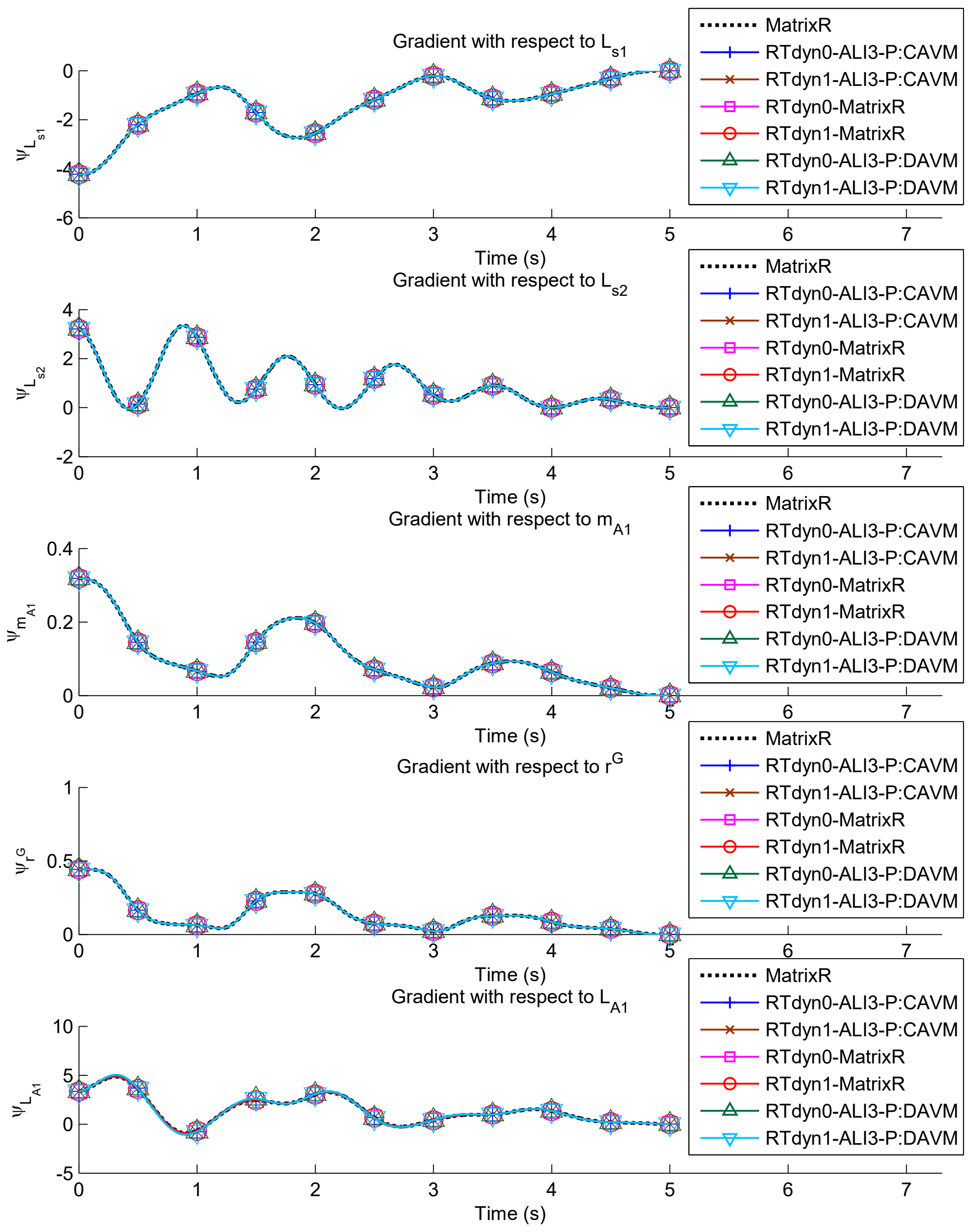


Figure 7.8: Sensitivity analysis of the objective function $\Psi^1$ using the AVM.

instant terms that have to be added at the initial and final times, which entails some "jumps" in the evolution of the gradient when these terms are added. Accordingly,

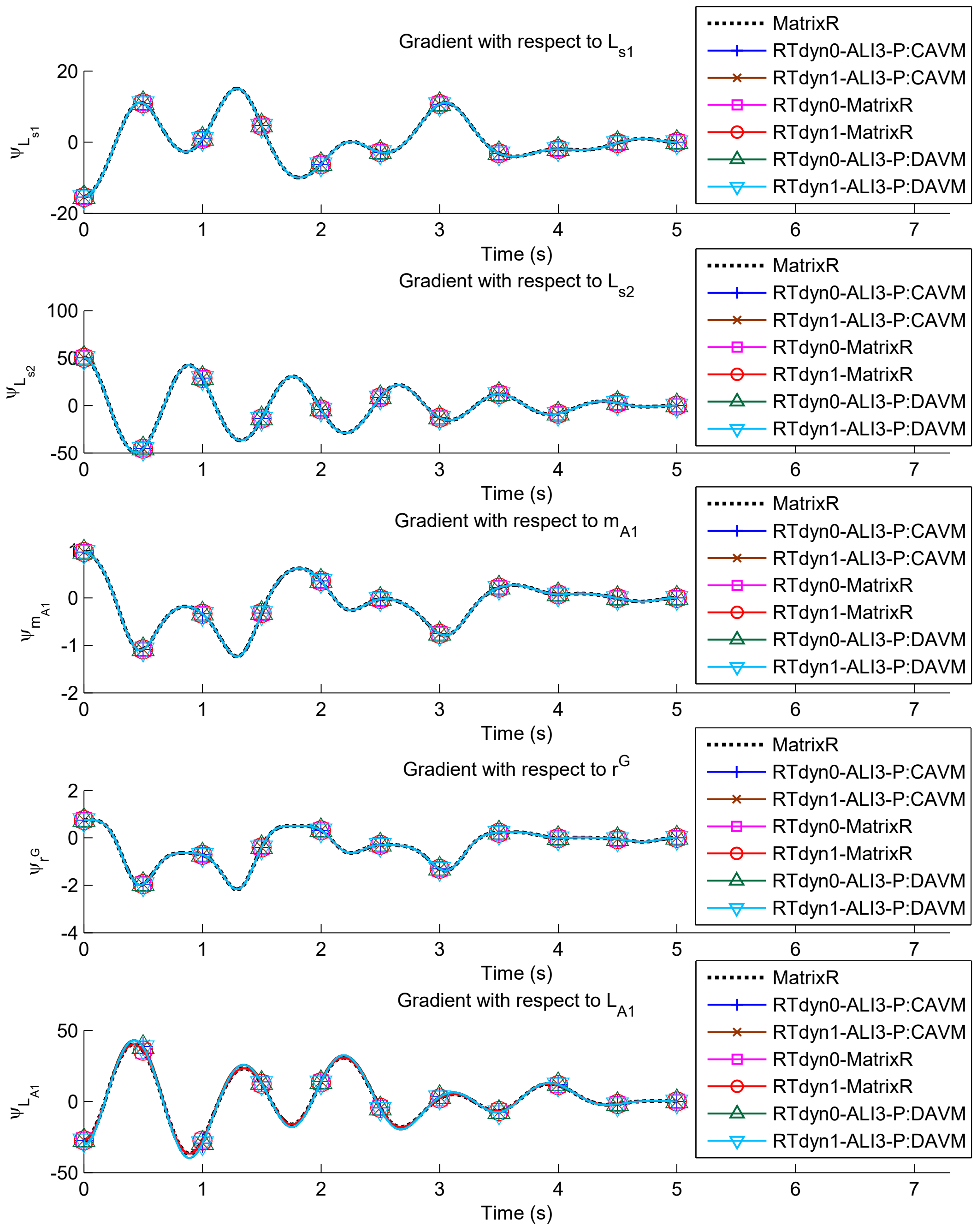


Figure 7.9: Sensitivity analysis of the objective function $\Psi^2$ using the AVM.

only the final results and not the evolution of the gradients are meaningful when the AVM is used.

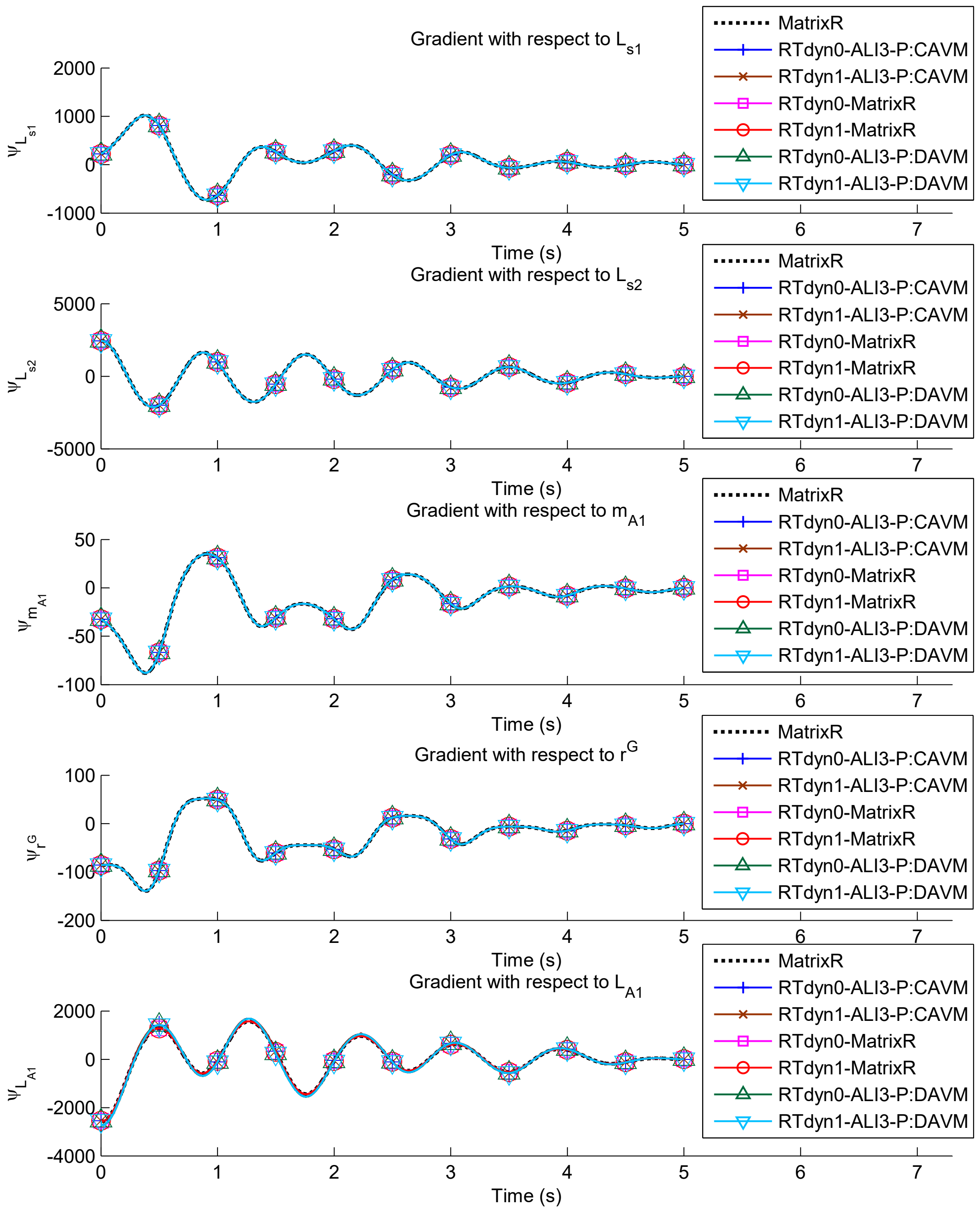


Figure 7.10: Sensitivity analysis of the objective function $\Psi^3$ using the AVM.

Although the previous figures seem to show a good convergence of all the formulations, this coincidence is made even more obvious in Tables 7.4, 7.5 and 7.6, where the final results of each gradient are displayed for each one of the components of the objective function.

These tables evidence a good convergence of all the formulations regardless of the

| Formulation | $\psi^1_{L_{s1}}$ | $\psi^1_{L_{s2}}$ | $\psi^1_{m_{A1}}$ | $\psi^1_{r_G}$ | $\psi^1_{L_{A1}}$ |
|---|---|---|---|---|---|
| Reference | -4.228 | 3.212 | 0.3186 | 0.4423 | 3.360 |
| RTdyn0 ALI3-P: DDM | -4.228 | 3.212 | 0.3186 | 0.4423 | 3.359 |
| RTdyn0 ALI3-P: CAVM | -4.228 | 3.212 | 0.3186 | 0.4423 | 3.360 |
| RTdyn0 ALI3-P: DAVM | -4.228 | 3.213 | 0.3186 | 0.4424 | 3.360 |
| RTdyn1 ALI3-P: DDM | -4.228 | 3.212 | 0.3186 | 0.4423 | 3.359 |
| RTdyn1 ALI3-P: CAVM | -4.228 | 3.212 | 0.3186 | 0.4423 | 3.360 |
| RTdyn1 ALI3-P: DAVM | -4.228 | 3.213 | 0.3186 | 0.4424 | 3.360 |
| RTdyn0 MatrixR: DDM | -4.228 | 3.212 | 0.3186 | 0.4423 | 3.360 |
| RTdyn0 MatrixR: CAVM | -4.228 | 3.212 | 0.3186 | 0.4423 | 3.360 |
| RTdyn1 MatrixR: DDM | -4.228 | 3.212 | 0.3186 | 0.4423 | 3.360 |
| RTdyn1 MatrixR: CAVM | -4.228 | 3.212 | 0.3186 | 0.4423 | 3.360 |

Table 7.4: Gradient of objective function $\Psi^1$

| Formulation | $\psi^2_{L_{s1}}$ | $\psi^2_{L_{s2}}$ | $\psi^2_{m_{A1}}$ | $\psi^2_{r_G}$ | $\psi^2_{L_{A1}}$ |
|---|---|---|---|---|---|
| Reference | -15.45 | 50.32 | 0.9700 | 0.7454 | -27.37 |
| RTdyn0 ALI3-P: DDM | -15.45 | 50.32 | 0.9700 | 0.7454 | -27.37 |
| RTdyn0 ALI3-P: CAVM | -15.45 | 50.32 | 0.9700 | 0.7453 | -27.37 |
| RTdyn0 ALI3-P: DAVM | -15.46 | 50.32 | 0.9707 | 0.7460 | -27.36 |
| RTdyn1 ALI3-P: DDM | -15.45 | 50.32 | 0.9700 | 0.7454 | -27.37 |
| RTdyn1 ALI3-P: CAVM | -15.45 | 50.32 | 0.9700 | 0.7453 | -27.37 |
| RTdyn1 ALI3-P: DAVM | -15.46 | 50.32 | 0.9707 | 0.7460 | -27.36 |
| RTdyn0 MatrixR: DDM | -15.45 | 50.32 | 0.9700 | 0.7454 | -27.37 |
| RTdyn0 MatrixR: CAVM | -15.45 | 50.32 | 0.9700 | 0.7454 | -27.37 |
| RTdyn1 MatrixR: DDM | -15.45 | 50.32 | 0.9700 | 0.7454 | -27.37 |
| RTdyn1 MatrixR: CAVM | -15.45 | 50.32 | 0.9700 | 0.7454 | -27.37 |

Table 7.5: Gradient of objective function $\Psi^2$

reference point, the dynamic formulation and the sensitivity analysis method selected. In fact, the biggest divergences are detected in the discrete adjoint variable method applied to the semi-recursive ALI3-P scheme, but it is never larger than a 0.5% of the reference result. In general, all the formulations described in the present work yield similar results, despite the remarkable differences among models, dynamic schemes of solution and sensitivity analysis methods.

Paying attention to Table 7.7, it is obvious that the computation of the sensitivity analysis using relative coordinate models is slower than the equivalent formulations in natural coordinates, even though the dynamics are computed faster, according to Table 7.3. This increase in CPU time is partially due to the concatenation of derivatives in relative coordinate models, whereas in natural coordinates models the derivative of each term is more direct. Besides, it can be seen that the direct differentiation method

| Formulation | $\psi^3_{L_{s1}}$ | $\psi^3_{L_{s2}}$ | $\psi^3_{m_{A1}}$ | $\psi^3_{r_G}$ | $\psi^3_{L_{A1}}$ |
|---|---|---|---|---|---|
| Reference | 221.8 | 2437 | -32.51 | -85.70 | -2547 |
| RTdyn0 ALI3-P: DDM | 221.7 | 2437 | -32.51 | -85.70 | -2547 |
| RTdyn0 ALI3-P: CAVM | 221.8 | 2437 | -32.51 | -85.70 | -2547 |
| RTdyn0 ALI3-P: DAVM | 221.1 | 2437 | -32.43 | -85.67 | -2547 |
| RTdyn1 ALI3-P: DDM | 221.7 | 2437 | -32.51 | -85.70 | -2547 |
| RTdyn1 ALI3-P: CAVM | 221.8 | 2437 | -32.51 | -85.70 | -2547 |
| RTdyn1 ALI3-P: DAVM | 221.1 | 2437 | -32.43 | -85.67 | -2547 |
| RTdyn0 MatrixR: DDM | 221.8 | 2437 | -32.51 | -85.70 | -2547 |
| RTdyn0 MatrixR: CAVM | 221.8 | 2437 | -32.51 | -85.70 | -2547 |
| RTdyn1 MatrixR: DDM | 221.8 | 2437 | -32.51 | -85.70 | -2547 |
| RTdyn1 MatrixR: CAVM | 221.8 | 2437 | -32.51 | -85.70 | -2547 |

Table 7.6: Gradient of objective function $\Psi^3$

| Formulation | CPU time | CPU time natural | Ratio *nat/rel* |
|---|---|---|---|
| RTdyn0 ALI3-P: DDM | 0.719 | 0.547 | 0.761 |
| RTdyn0 ALI3-P: CAVM | 0.781 | 0.750 | 0.960 |
| RTdyn0 ALI3-P: DAVM | 0.766 | 0.797 | 1.041 |
| RTdyn1 ALI3-P: DDM | 0.719 | 0.547 | 0.761 |
| RTdyn1 ALI3-P: CAVM | 0.781 | 0.750 | 0.960 |
| RTdyn1 ALI3-P: DAVM | 0.766 | 0.797 | 1.041 |
| RTdyn0 MatrixR: DDM | 1.109 | 0.703 | 0.634 |
| RTdyn0 MatrixR: CAVM | 1.109 | 0.844 | 0.761 |
| RTdyn1 MatrixR: DDM | 1.109 | 0.703 | 0.634 |
| RTdyn1 MatrixR: CAVM | 1.109 | 0.844 | 0.761 |

Table 7.7: CPU time of relative coordinates formulations compared with the equivalent one in natural coordinates.

produces a bigger gap between times of natural and relative coordinate models, while this deviation is reduced in the adjoint variable method. Regarding formulations, ALI3-P is generally faster than Matrix R formulations both in natural and relative models. The low number of parameters makes the direct differentiation method faster in all the cases, except in semi-recursive Matrix R formulations, where all the computational times for any sensitivity analysis are the same. One of the reasons of the similarity of CPU times in these formulations relies on the fact that the bulk of the computational effort is here centered in the calculation and assembly of each derivative regardless of the scheme of solution selected.

Looking at the selection of reference points, Table 7.7 shows that the unification of code and other developments considered in the MBSLIM implementation lead to a minimum deviation in time between RTdyn0 and RTdyn1, such that it is less than 1 millisecond in this numerical example for every one of the formulations tested.

## 7.2 Spatial slider crank

The spatial slider-crank is a four-body mechanism described in the Library of Computational Benchmark Problems of the IFToMM Technical Committee for Multibody Dynamics [134] based on the numerical example originally proposed in [114]. It has been used as benchmark problem in a work associated to the present thesis [60], in which the semi-recursive ALI3-P formulation was combined with the RTdyn0 and RTdyn1 versions of the semi-recursive accumulation.

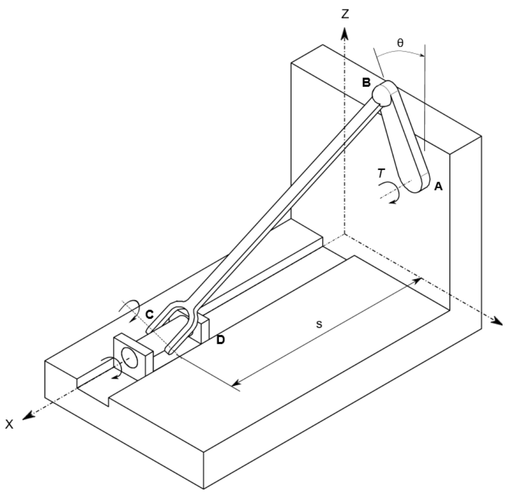


Figure 7.11: Spatial slider crank mechanism.

### 7.2.1 Multibody model

The spatial slider crank, represented in Figure 7.11, is composed of 3 movable bodies: the crank, connected to the ground (fixed body) by a revolute joint; the slider, linked to the ground by a prismatic joint; and the connecting rod, joined to the crank by a spherical joint and to the slider by a Cardan joint.

This multibody system is specially interesting in joint coordinate modeling due to the fact that gathers 4 different types of joints (revolute, prismatic, Cardan and spherical) and a closed loop. Moreover, only 3 bodies are involved, which even makes it possible writing the equations of motion and the accumulated mass matrices and forces vector by hand in order to test the method.

This numerical example has been used in the earliest stages of the implementation of the automatic opening of closed loops algorithm. Let us compare the four possible options resulting from opening the loop by eliminating one of the 4 joints of the system. Table 7.8 evidences that the best option is to eliminate the spherical joint, since the number of constraints and joint-variables generated is the lower of all the 4 cases. In fact, this example shows that the joints with the higher number of degrees of freedom should be eliminated first. For instance, comparing the model generated by the elimination of a revolute or prismatic joint with the one generated by the cut of the Cardan joint, Table 7.8 makes it clear that Cardan joints should be removed prior to one degree-of-freedom joints.

| Joint eliminated | Number of variables | Number of constraint equations |
|---|---:|---:|
| Revolute | 4+2+1=7 | 6+1=7 |
| Spherical | 1+2+1=4 | 3 |
| Cardan | 1+4+1=6 | 4+1=5 |
| Prismatic | 4+2+1=7 | 9+1=10 |

Table 7.8: Study of models generated by the elimination of different joints on the spatial slider crank

The joint coordinate model automatically generated by the MBSLIM multibody library consists of 2 kinematic chains gathering a prismatic, a revolute and a Cardan joint. This delivers a model described by 4 joint-coordinates subjected to 3 loop-closure constraint equations. For comparison purposes, MBSLIM delivers an equivalent natural coordinates model composed of 25 mixed coordinates (24 Cartesian coordinates plus an angular coordinate) subjected to 28 redundant constraints.

Once described the topology, let us focus on the kinematic and dynamic description of the model. First, it is convenient to identity the crank as body AB, the rod as body BC and the slider block as body CD according to the points between which each body is defined in Figure 7.11. The lengths of the crank and the rod are:

$$L_{AB} = 0.08\,m \tag{7.21}$$
$$L_{BC} = 0.3\,m \tag{7.22}$$

The mass of each body is:

$$m_{AB} = 0.12\,kg \tag{7.23}$$
$$m_{BC} = 0.5\,kg \tag{7.24}$$
$$m_{CD} = 2.0\,kg \tag{7.25}$$

while their local moments of inertia with respect to their centers of mass (located at the geometric center of each body) are:

$$\bar{\mathbf{J}}_{AB} = \begin{bmatrix} 10^{-4} & 0 & 0 \\ 0 & 10^{-4} & 0 \\ 0 & 0 & 10^{-4} \end{bmatrix} kg \cdot m^2 \tag{7.26}$$
$$\bar{\mathbf{J}}_{BC} = \begin{bmatrix} 4 \cdot 10^{-3} & 0 & 0 \\ 0 & 4 \cdot 10^{-3} & 0 \\ 0 & 0 & 4 \cdot 10^{-3} \end{bmatrix} kg \cdot m^2 \tag{7.27}$$
$$\bar{\mathbf{J}}_{CD} = \begin{bmatrix} 10^{-4} & 0 & 0 \\ 0 & 10^{-4} & 0 \\ 0 & 0 & 10^{-4} \end{bmatrix} kg \cdot m^2 \tag{7.28}$$

The dynamic simulation proposed in [134] consists in a free motion of the mechanism subjected to the action of gravity for an initial angular speed of 6 $rad/s$ for the crank. Nevertheless, the problem is modified here in order to include more sensitivity parameters. In this regard, a new spring-damper force is added between a point placed at the global position $\begin{bmatrix}0 & 0 & 0\end{bmatrix}^{\mathrm{T}}$ and point D with the characteristics specified in Table 7.9:

| Force | Stiffness $(N/m)$ | Damping $(Ns/m)$ | Natural length $(m)$ |
|---|---|---|---|
| Spring 1 | 10 | 1 | 0.25 |

Table 7.9: Coefficients of spring-damper forces acting on the spatial slider crank

For the sensitivity analysis, the gradient of the following vector of objective functions is sought:

$$\boldsymbol{\psi} = \begin{bmatrix}\psi^1 \\ \psi^2 \\ \psi^3\end{bmatrix} \tag{7.29}$$

with

$$\psi^1 = \int_{t_0}^{t_F} (\mathbf{r}_D - \mathbf{r}_{D0})^{\mathrm{T}} (\mathbf{r}_D - \mathbf{r}_{D0})\, \mathrm{d}t \tag{7.30a}$$

$$\psi^2 = \int_{t_0}^{t_F} \dot{\mathbf{r}}_D^{\mathrm{T}} \dot{\mathbf{r}}_D \mathrm{d}t \tag{7.30b}$$

$$\psi^3 = \int_{t_0}^{t_F} \ddot{\mathbf{r}}_D^{\mathrm{T}} \ddot{\mathbf{r}}_D \mathrm{d}t \tag{7.30c}$$

wherein

$$\mathbf{r}_{D0} = \begin{bmatrix}0.25 & 0 & 0\end{bmatrix}^{\mathrm{T}} \tag{7.31}$$

As sensitivity parameters, the mass of the slider, the stiffness and damping coefficients of the spring-damper along with its natural length are considered. Behold that the sensitivity analysis would deliver the impact of the mass and the spring-damper parameters on the dynamic response of the system.

$$\boldsymbol{\rho} = \begin{bmatrix}m_{CD} & k_{S1} & c_{S1} & L_{S1}\end{bmatrix}^{\mathrm{T}} \tag{7.32}$$

The sensitivity analysis is studied for the 5 second dynamic maneuver of the mechanism subjected to gravitational and spring-damper forces.

### 7.2.2 Numerical results: dynamics

The dynamic formulations tested in this section are the semi-recursive ALI3-P and the semi-recursive Matrix R formulations combined with the particular reference point selection of RTdyn0 and RTdyn1. The semi-recursive dynamic results delivered are compared in terms of accuracy with the Matrix R formulation in natural coordinates, and in terms of computational time with the equivalent formulations in natural coordinates.

The simulation is executed with a fixed time step of 1 millisecond and with the implicit trapezoidal rule as numerical integrator. Each one of the semi-recursive dynamic formulations are configured according to Table 7.10.

| Formulation [1] | R0-ALI3-P | R1-ALI3-P | R0-MatrixR | R1-MatrixR |
|---|---|---|---|---|
| Tolerance | $10^{-10}$ | $10^{-10}$ | $10^{-10}$ | $10^{-10}$ |
| Penalty factor $\alpha$ | $10^{10}$ | $10^{10}$ | - | - |
| Projection matrix | $\mathbf{M}^d$ | $\mathbf{M}^d$ | - | - |
| Proj. scaling factor $\varsigma$ | $10^{-4}$ | $10^{-4}$ | - | - |
| Projection type | Non-iterative | Non-iterative | - | - |

[1]R0: RTdyn0; R1: RTdyn1.

Table 7.10: Configuration parameters for each formulation in the dynamic simulation of the spatial slider crank.

In order to compare the dynamics obtained with the two constrained semi-recursive formulations combined with accumulation schemes RTdyn0 and RTdyn1, with the benchmark problem proposed in [134], a first simulation without the spring-damper and under the conditions specified in the cited reference is accomplished, with the results exposed in Figure 7.12. The reference solution obtained with Matrix R in natural coordinates and all the semi-recursive formulations are in concordance with the results presented in [134].

The incorporation of the spring-damper delivers a completely different dynamics, displayed in Figure 7.13. Behold that, in this case, the convergence of the five formulations considered is excellent despite the intrinsic differences between models and schemes of constraint enforcement (Matrix R and ALI3-P).

| Formulation | CPU time | CPU time nat. coord | Ratio *nat/rel* |
|---|---|---|---|
| RTdyn0 ALI3-P | 0.156 | 0.266 | 1.705 |
| RTdyn1 ALI3-P | 0.156 | 0.266 | 1.705 |
| RTdyn0 MatrixR | 0.297 | 0.531 | 1.788 |
| RTdyn1 MatrixR | 0.297 | 0.531 | 1.788 |

Table 7.11: CPU time of relative coordinates formulations compared with the equivalent formulation in natural coordinates.

One important advantage of joint-coordinate formulations relies on its high efficiency. Table 7.11 proves that, even though the spatial slider crank involves a low number of bodies, semi-recursive models perform better than natural coordinates models in terms of computational time. This table shows that every formulation is executed faster in relative coordinate models, becoming the semi-recursive ALI3-P

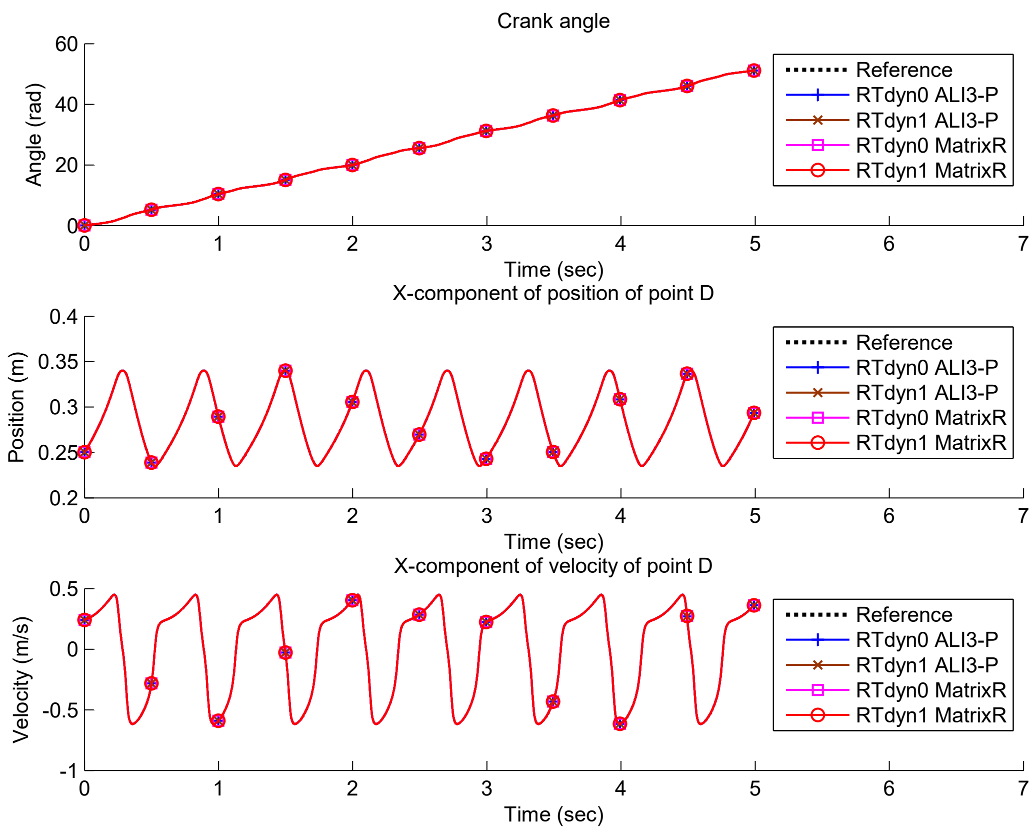


Figure 7.12: Dynamic simulation of spatial slider crank without spring under benchmark conditions.

formulation (both RTdyn0 and RTdyn1 approaches) the most efficient. Behold also that the divergence in computational times between RTdyn0 and RTdyn1 is minimal, which makes both approaches equally desirable for a dynamic simulation.

### 7.2.3 Numerical results: sensitivity analysis

The sensitivity analysis is executed on the dynamic maneuver with the spring-damper force between point D and a point fixed to the ground. The sensitivity analyses tested and compared in this numerical example are:

- RTdyn0 ALI3-P: forward, continuous adjoint and discrete adjoint sensitivity formulations.
- RTdyn1 ALI3-P: forward, continuous adjoint and discrete adjoint sensitivity formulations.
- RTdyn0 Matrix R: forward and continuous adjoint sensitivity formulations.
- RTdyn1 Matrix R: forward and continuous adjoint sensitivity formulations.

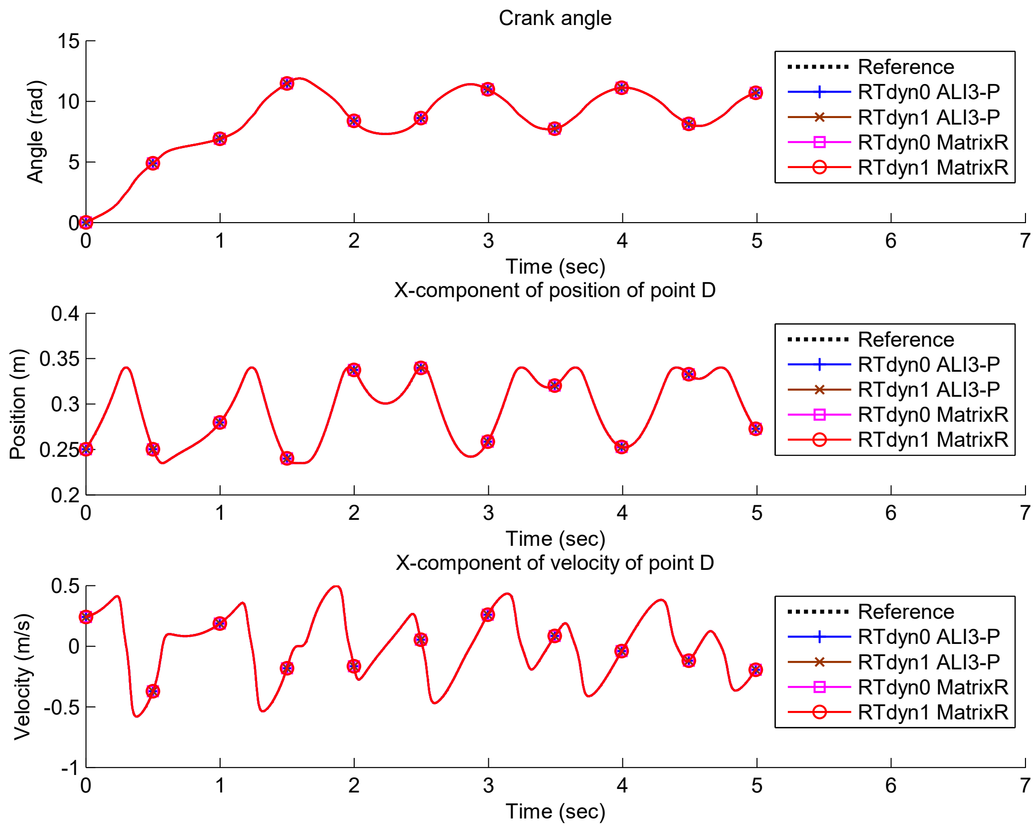


Figure 7.13: Dynamic simulation of spatial slider crank with a spring force.

The reference results for the semi-recursive direct and adjoint sensitivities are calculated by means of the Matrix R direct and adjoint sensitivity formulations in natural coordinates.

First, let us look at the results of the evaluation of the objective function over time for the five dynamic formulations used for the simulation of the spatial slider crank mechanism. Figure 7.14 shows, once again, the accurate convergence among formulations.

The gradient results of the objective function given by (7.29) and (7.30) with respect to the set of parameters specified on (7.32) applying the semi-recursive ALI3-P and Matrix R forward sensitivity formulations are presented on Figures 7.15, 7.16 and 7.17 for each of the objective functions.

Figures 7.15, 7.16 and 7.17 validate both the implementation and theory behind the analytical sensitivities of joint coordinate models following a semi-recursive scheme. It should be remarked that 3 different joint types appear in this multibody model, involving the particular analytical derivatives of the kinematic and recursive relations associated to the revolute joint, prismatic joint and Cardan joint. Moreover, the derivatives of the loop-closure constraint referred to the elimination of the spherical joint have been considered too.

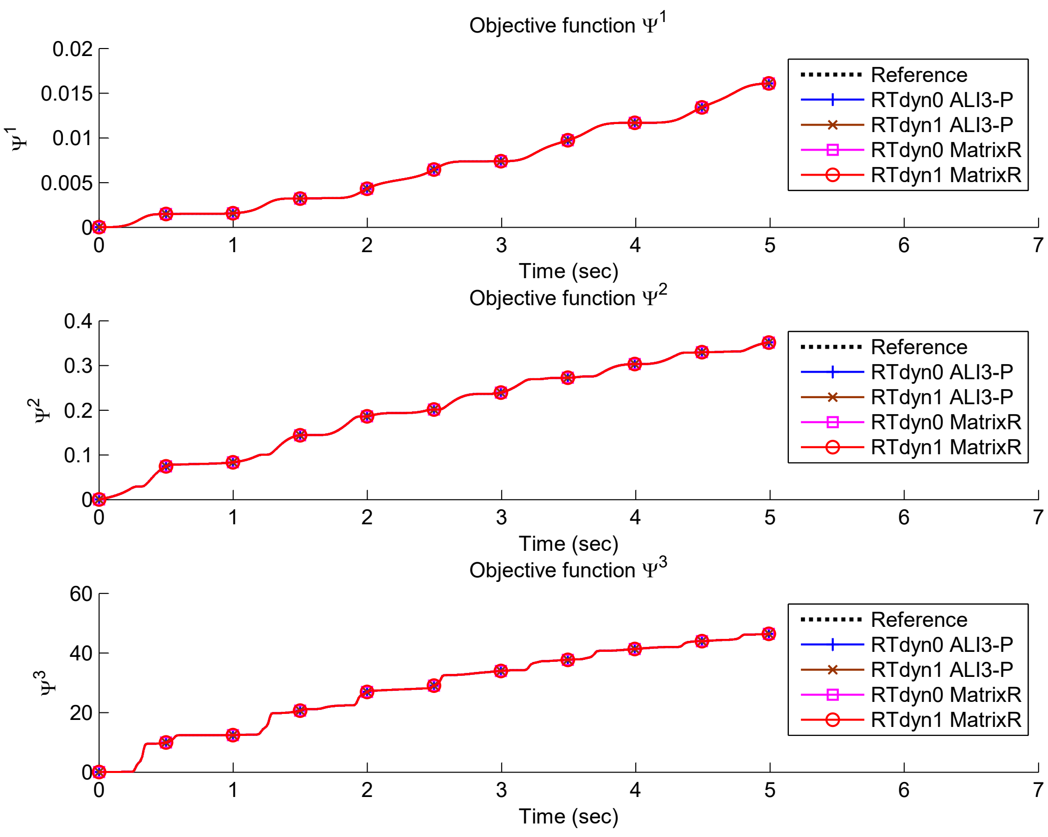


Figure 7.14: Evolution of the each of the spatial-slider-crank objective functions over time.

Besides, the elimination of the spherical joint during the automatic opening of closed loops creates two different kinematic chains, this is, a branched system. The differentiation of the accumulation and assembly processes is therefore also validated for this numerical example.

| Formulation | CPU time | CPU time natural | Ratio *nat/rel* |
|---|---:|---:|---:|
| RTdyn0 ALI3-P | 0.438 | 0.641 | 1.464 |
| RTdyn1 ALI3-P | 0.438 | 0.641 | 1.464 |
| RTdyn0 MatrixR | 1.031 | 0.984 | 0.954 |
| RTdyn1 MatrixR | 1.031 | 0.984 | 0.954 |

Table 7.12: CPU time of relative coordinates forward sensitivity formulations compared with the equivalent formulation in natural coordinates.

In terms of computational time, the gap between semi-recursive and global formulations displayed in Table 7.11 is reduced in the sensitivity analysis. Table 7.12 shows

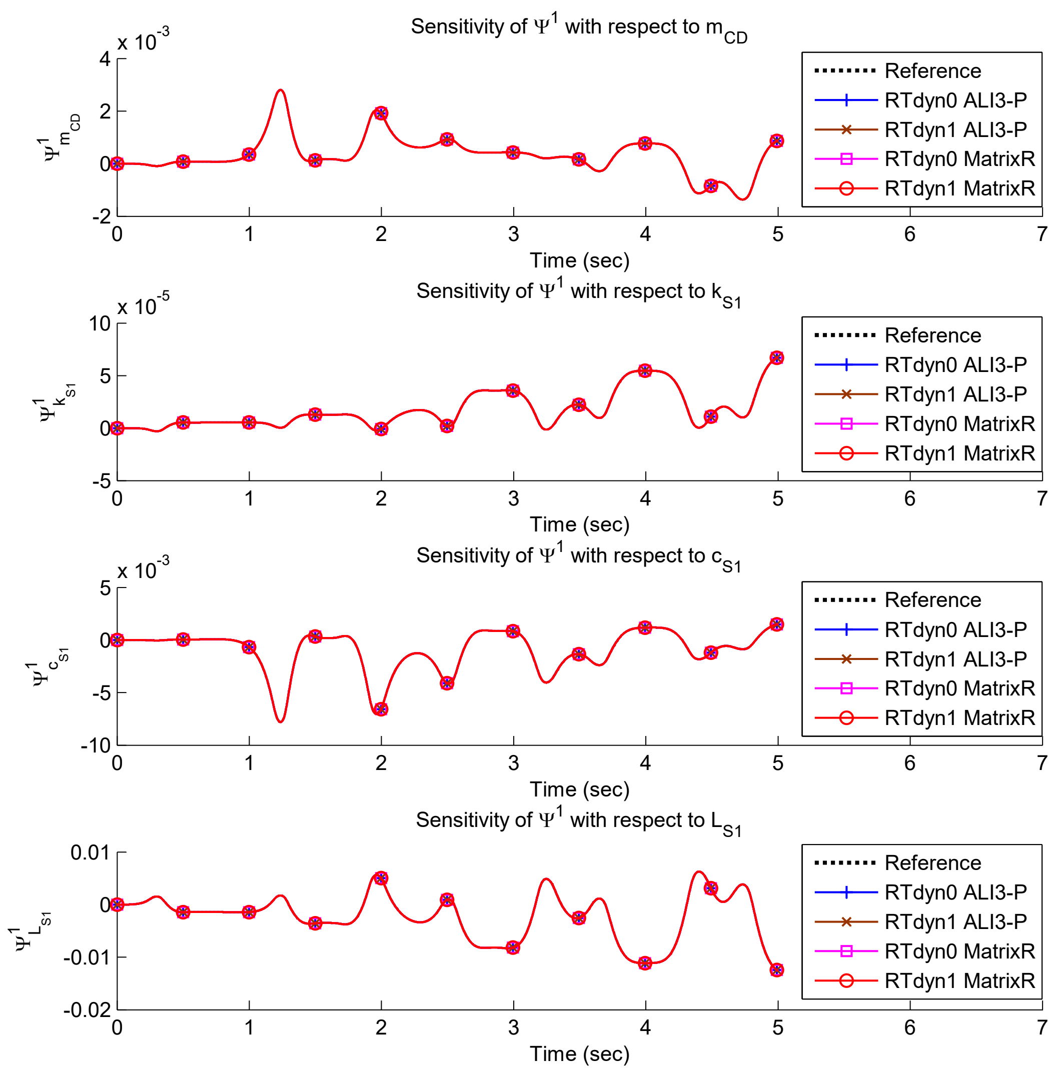


Figure 7.15: Forward sensitivity analysis of the objective function $\Psi^1$ for the spatial slider crank mechanism.

that semi-recursive formulations are a 46.4 % faster than the equivalent method in natural coordinates, but the result is inverted in semi-recursive Matrix R formulations, which are a 4.6% slower than global methods. Furthermore, comparing ALI3-P and Matrix R formulations, a bigger divergence is observed between the computational times of joint-coordinate formulations than in natural coordinate models. This efficiency deviation can be attributed to the presence of more intermediate derivatives in Matrix R formulations than in ALI3-P schemes.

The computational gains (and losses) observed in Table 7.12 do not justify the implementation effort required to obtain a general library in joint-coordinates for the

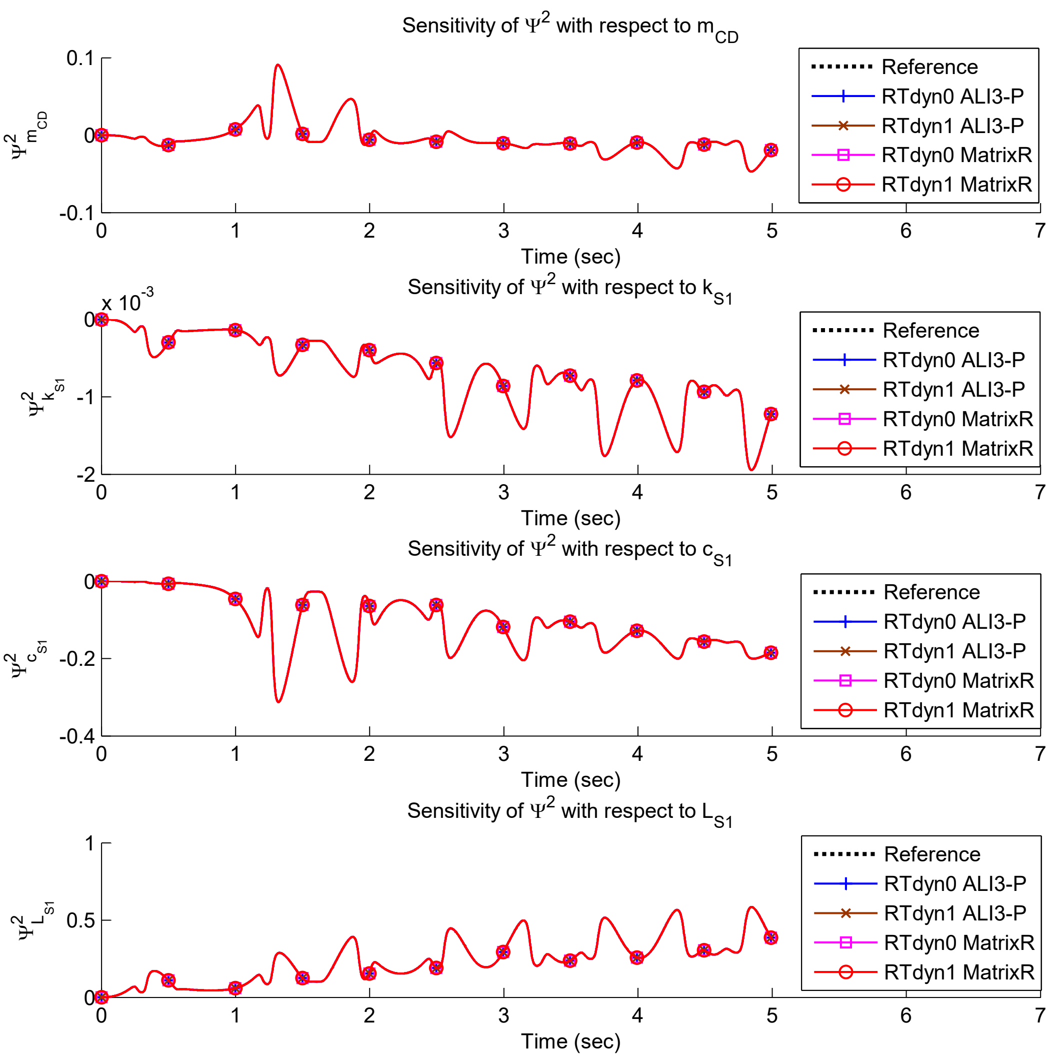


Figure 7.16: Forward sensitivity analysis of the objective function $\Psi^2$ for the spatial slider crank mechanism.

sensitivity analysis of the dynamics of a multibody system. Nevertheless, behold that the main problem of global methods (with natural or reference point coordinates) is that they generate a problem involving a long list of dependent coordinates and constraint equations. For very reduced problems, such as the spatial slider crack, global methods do not present important shortcomings since the number of variables and constraints is relatively low. For problems involving more bodies and joints, both semi-recursive Matrix R and ALI3-P formulations are expected to be more efficient, both at dynamic and sensitivity analysis levels.

Let us compare now the results of the continuous and discrete adjoint sensitivity

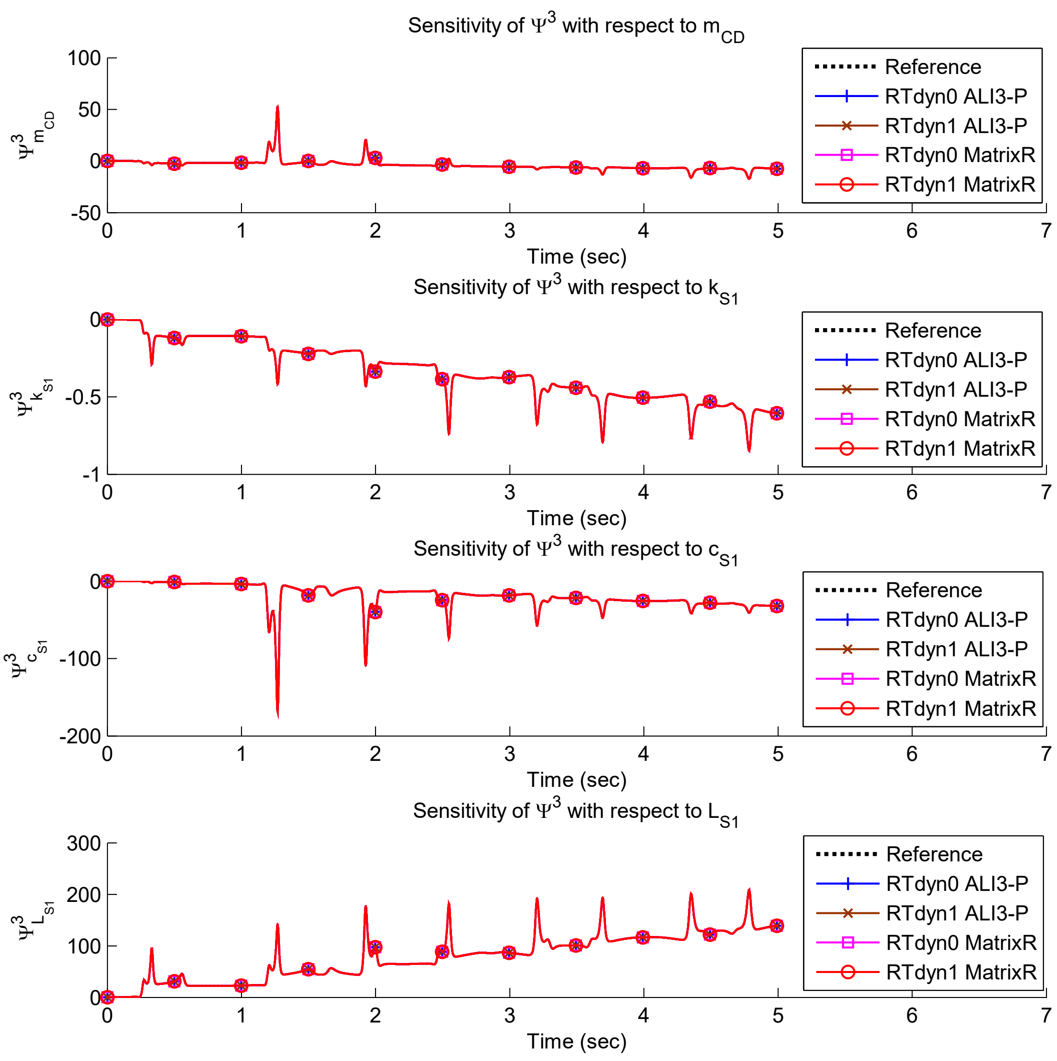


Figure 7.17: Forward sensitivity analysis of the objective function $\Psi^3$ for the spatial slider crank mechanism.

formulations. The present numerical example sheds light on the advantages of discrete adjoint variable methods with respect to continuous methods. With the time step selected for the dynamic simulation and conserved for the adjoint problem, the solution of the continuous adjoint variable ALI3-P equations involves the addition of important numerical errors that lead to inaccurate solutions. In other words, the continuous adjoint ALI3-P equations require a smaller time step to be solved. Since interpolation between time steps is not currently supported by MBSLIM for adjoint methods, the other option is to solve both dynamic and sensitivity problems with smaller time steps. In this case, the time step is slowed down to 0.1 milliseconds for

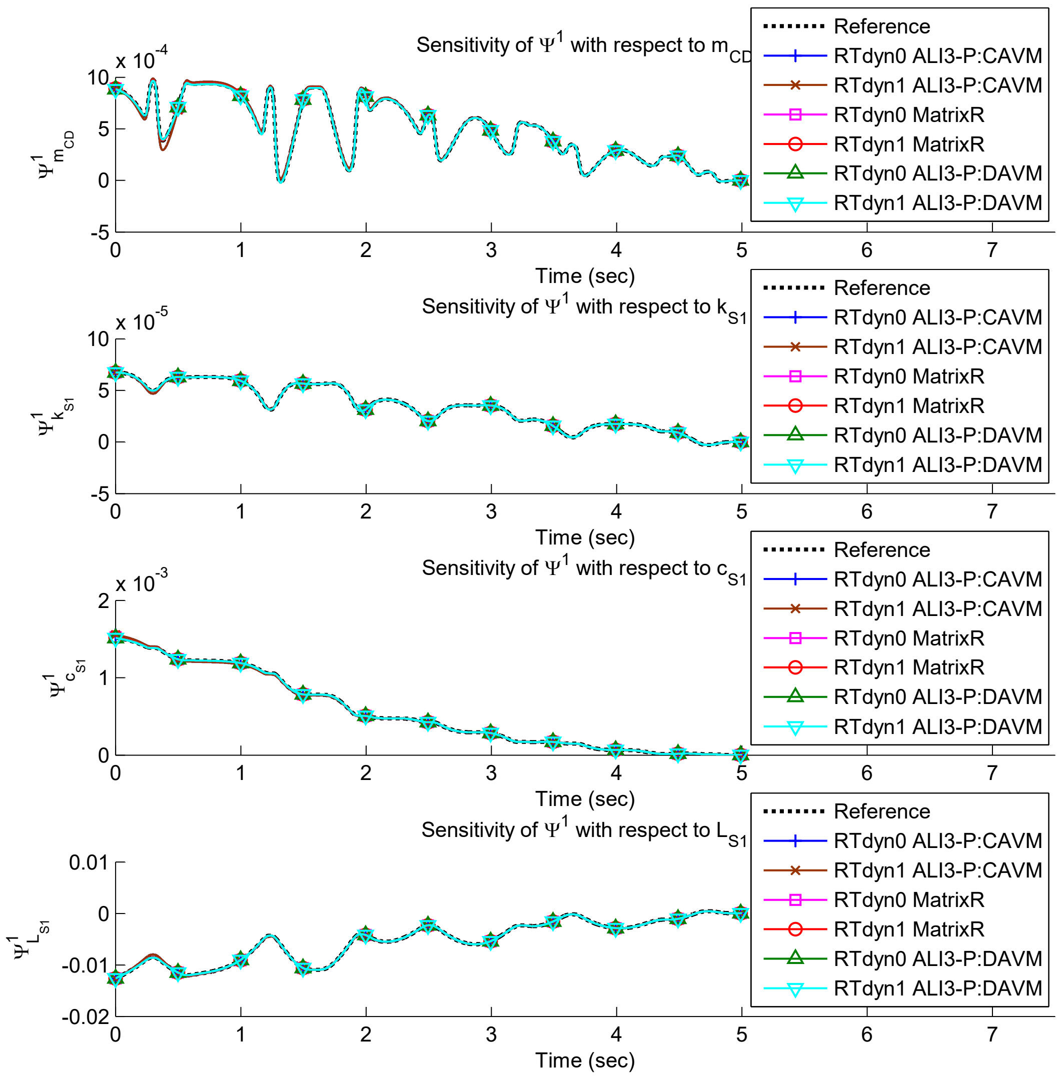


Figure 7.18: Adjoint sensitivity analysis of the objective function $\Psi^1$ for the spatial slider crank mechanism.

the global and topological continuous adjoint ALI3-P methods, while the rest of the formulations are still solved with a time step of 1 millisecond. Behold in Figures 7.18, 7.19 and 7.20 the level of accuracy of discrete and continuous semi-recursive adjoint sensitivity formulations even for this reduced time step.

Figures 7.18, 7.19 and 7.20 display a great convergence of the gradient results evaluated by means of a continuous adjoint variable method for global and semi-recursive Matrix R formulations and for the discrete approach of the semi-recursive ALI3-P formulation. These results clearly validate the continuous and discrete semi-recursive adjoint expressions. Moreover, note that the deviation between RTdyn0 and

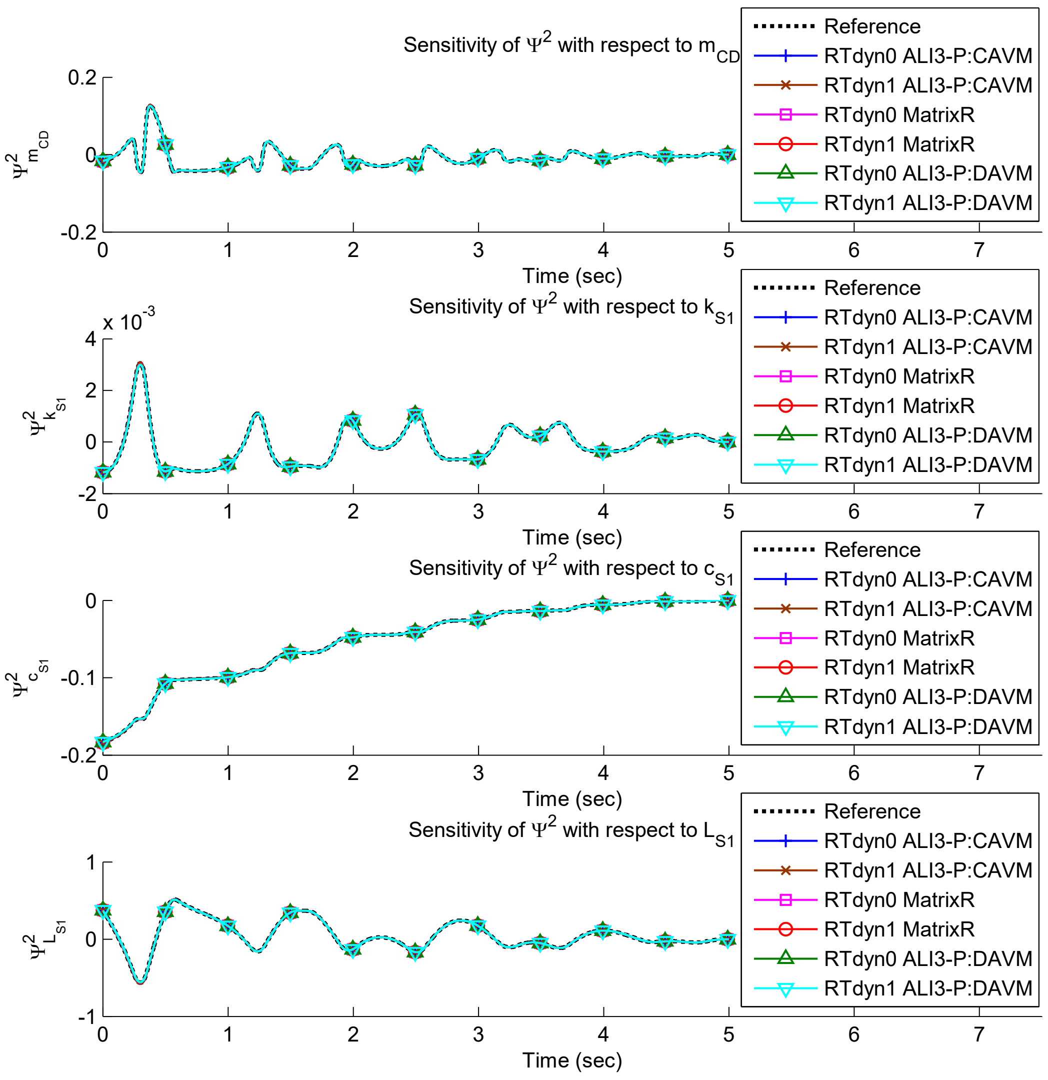


Figure 7.19: Adjoint sensitivity analysis of the objective function $\Psi^2$ for the spatial slider crank mechanism.

RTdyn1 approaches is extremely reduced despite, as presented in chapters 2 and 4, the different recursive kinematic and sensitivity expressions.

In Table 7.13, the performance of the adjoint formulations are assessed in terms of computational time, considering the same time step of 1 millisecond (despising the fact that the continuous semi-recursive ALI3-P adjoint variable method requires a lower time step in order to obtain the same accuracy level of other semi-recursive sensitivity formulations). Note that all the semi-recursive adjoint sensitivity formulations are faster than the equivalent natural coordinate formulations, being the fastest both discrete and continuous semi-recursive ALI3-P adjoint formulations. As it was expected, the similarities between the application of the continuous and discrete adjoint variable methods yield analog computational expenses.

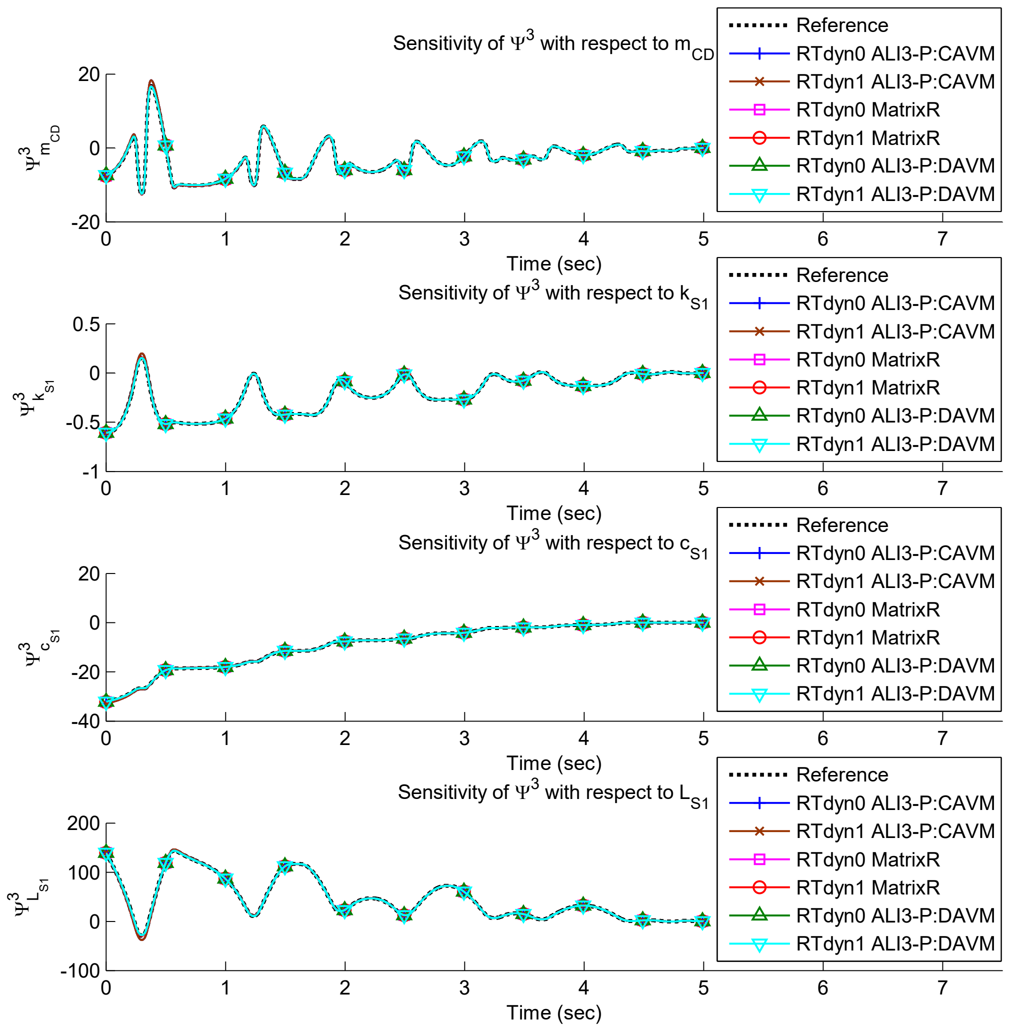


Figure 7.20: Adjoint sensitivity analysis of the objective function $\Psi^3$ for the spatial slider crank mechanism.

| Formulation | CPU time | CPU time natural | Ratio $nat/rel$ |
|---|---:|---:|---:|
| RTdyn0 ALI3-P: CAVM | 0.469 | 0.750 | 1.599 |
| RTdyn1 ALI3-P : CAVM | 0.469 | 0.750 | 1.599 |
| RTdyn0 MatrixR | 0.953 | 1.063 | 1.115 |
| RTdyn1 MatrixR | 0.953 | 1.063 | 1.115 |
| RTdyn0 ALI3-P: DAVM | 0.469 | 0.797 | 1.699 |
| RTdyn1 ALI3-P: DAVM | 0.469 | 0.797 | 1.699 |

Table 7.13: CPU time of semi-recursive adjoint sensitivity formulations compared with the equivalent formulation in natural coordinates.

## 7.3 Buggy vehicle

In the third numerical example, a more complex real-life multibody system is studied in order to prove both the validity of recursive dynamic and sensitivity formulations as well as their performance in terms of accuracy and efficiency. The vehicle model considered is based on a real-life buggy vehicle used in past years in control and state observation applications [124, 135, 136] and flexible multibody dynamics [122], among other issues. Recently, a model of this vehicle has been used in [3] and [4] to check the validity of the direct and adjoint variable sensitivity analysis of ALI3-P formulations, and in [60] to test the dynamic results of semi-recursive ALI3-P formulations.

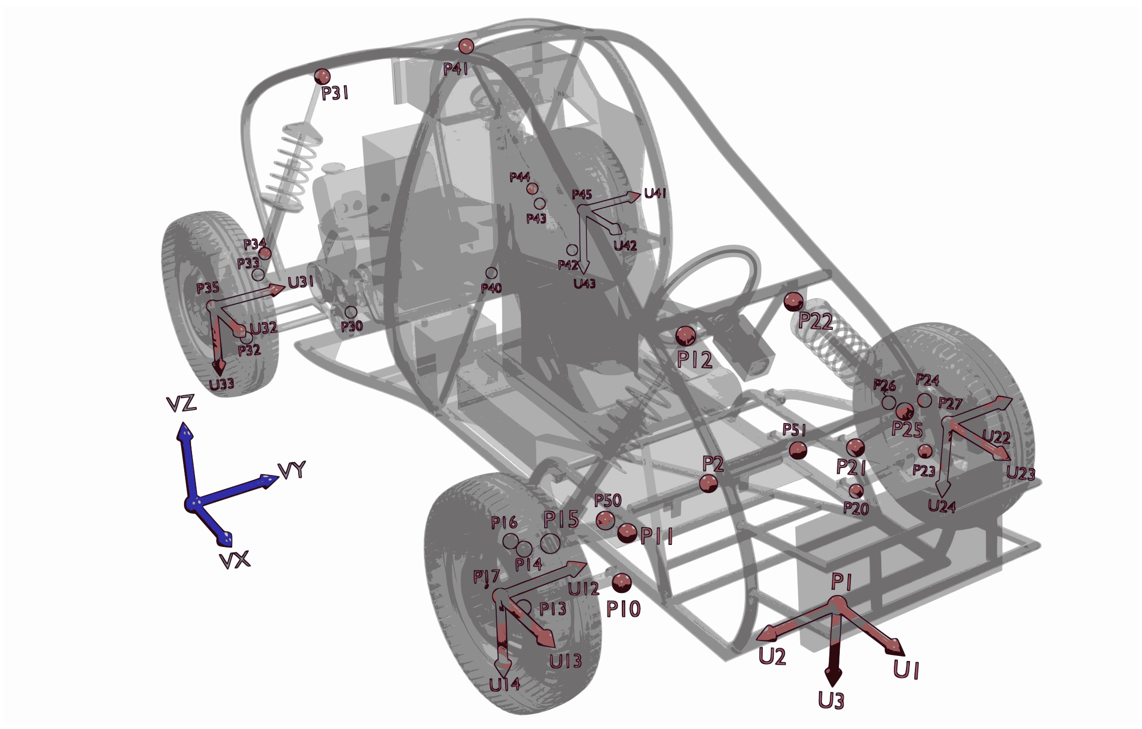


Figure 7.21: Four-wheeled vehicle, depicting the points and vectors used to define the model in MBSLIM.

### 7.3.1 Multibody model

As it has been commented in the introduction of this section, the model has been thoroughly studied in previous works [122, 124, 135, 136], thus only its main characteristics will be outlined here, referring the reader to the commented references for further detail.

In brief, the model consists in a four-wheeled buggy composed of 18 bodies with four articulated suspensions. The chassis is formed of a tubular structure on which 4 articulated suspensions are mounted. Both engine and pilot are considered as rigidly attached to the chassis, and their distribution contributes to a slight displacement of the center of mass with respect to the geometrical center of the chassis.

The front and rear axles have different suspension types, with a McPherson configuration in the rear axle and a double wishbone type for the frontal suspensions [135]. Each suspension system includes a physical spring-damper which is modeled by means of a spring-damper force, neglecting their effects in terms of masses and inertia. The spring-damper elements are leaned inwards in order to smoothen the suspension, as it can be observed on Figure 7.21, and they have been parameterized with the coefficients of Table 7.14.

| Force | Stiffness ($N/m$) | Damping ($Ns/m$) | Natural length ($m$) |
|---|---:|---:|---:|
| Front axle | 16000 | 10000 | 0.710 |
| Rear axle | 10595 | 6000 | 0.741 |

Table 7.14: Coefficients of spring forces acting on the five-bar linkage

The power is transmitted to the rear axle, while the two front wheels are driven by a steering rack linked to two connecting rods. Moreover, the coordinate associated to the steering rack is guided by means of a rheonomic constraint. This guidance will determine the dynamic maneuver executed.

The mechanism has been modeled with 33 points, 25 vectors, 4 angles (corresponding to each one of the wheels) and 5 distances (describing the motion of the steering rack and the spring-damper systems). Moreover, a rheonomous constraint is used as guidance of the steering rack. This information in terms of points, vectors, angles, distances and user-defined constraints is unique for natural and relative coordinate models, being the MBSLIM library responsible for generating the corresponding natural and relative coordinate models. The degrees of freedom defining the model are a combination of coordinates of points, vectors, angles and distances.

The set of forces applied on the model require a special mention. In addition to the gravitational forces related to the mass of each body, and the spring-damper forces of each one of the suspensions, contact and frictional forces have been considered to model the tires of the vehicle. In previous works [3, 4, 60], the model of contact used was based on a sphere-plane contact, but it has been substituted in the present simulations by a circumference-plane contact model. The computation of the contact and frictional forces is completely analytical (including the stiffness and damping matrices related to these forces) and they have been used in both natural and relative coordinate models.

The tires-ground normal contact is based on a Kelvin-Voigt contact model with hysteresis (see [137]) while frictional tire force components are evaluated by means of the simplified linearized tire model described in [138]. Both contact and frictional forces have been modeled with the coefficients specified in Table 7.15 (the nomenclature used in [137] for normal contact and in [138] for frictional forces is reused here).

| Force coefficient | Front axle | Rear axle |
|---|---|---|
| $K$ | 60430 | 60430 |
| $D_c$ | 100 | 100 |
| $D_R$ | 100 | 100 |
| $Radius$ | 0.30253 | 0.30253 |
| $\kappa_c$ | 0.8 | 0.8 |
| $\alpha_c$ | 0.2 | 0.2 |
| $\mu_x$ | 0.7 | 0.7 |
| $\mu_y$ | 0.7 | 0.7 |

Table 7.15: Coefficients of vehicle tire forces.

The natural coordinate model generated is composed of 180 mixed coordinates (171 Cartesian coordinates of points and vectors and 9 referred to to angles and distances) subjected to 178 constraint equations. Taking into account that the model has 14 degrees of freedom, and that the difference between the number of variables and constraint equations is 2, it is patent that it has redundant constraints. However, this is not a problem neither in natural or relative coordinates since MBSLIM supports them for each one of the formulations tested in this numerical example.

The topological model is slightly more difficult to obtain, requiring the identification of joints by means of relations of points and vectors between bodies, followed by the detection of closed loops and their opening. The joint coordinate model automatically generated by MBSLIM is composed of 18 kinematic joints defined by a total of 36 joint coordinates. During the generation of the model, 4 closed loops were detected and opened by the elimination of 4 spherical joints (16 coordinates and 4 normalization constraints were suppressed), and accordingly, 12 constraint equations of point coincidence were added to the model (3 per eliminated joint). Other constraint equations not defining joint relations, such as the rheonomic guidance of the steering rack, were added to the model, resulting a total of 26 constraint equations. The kinematic joints and the constraint equations present in the relative coordinate model are displayed on Figure 7.22.

At the initial position, the suspensions of the vehicle are not at the static equilibrium configuration since the chassis is slightly elevated, and therefore a suspension stabilization occurs during the first second of the simulation.

#### 7.3.1.1 First maneuver: step descent

Two maneuvers are tested with this multibody system. The first maneuver consists in the descent of a step of 1 $cm$ located at 5.5 $m$ from the origin, with a forward initial linear speed of 3 $m/s$ and with an angular speed of 11 $rad/s$ for each wheel. With this velocity, the step is reached approximately at $t = 2.0\,s$. The simulation time is

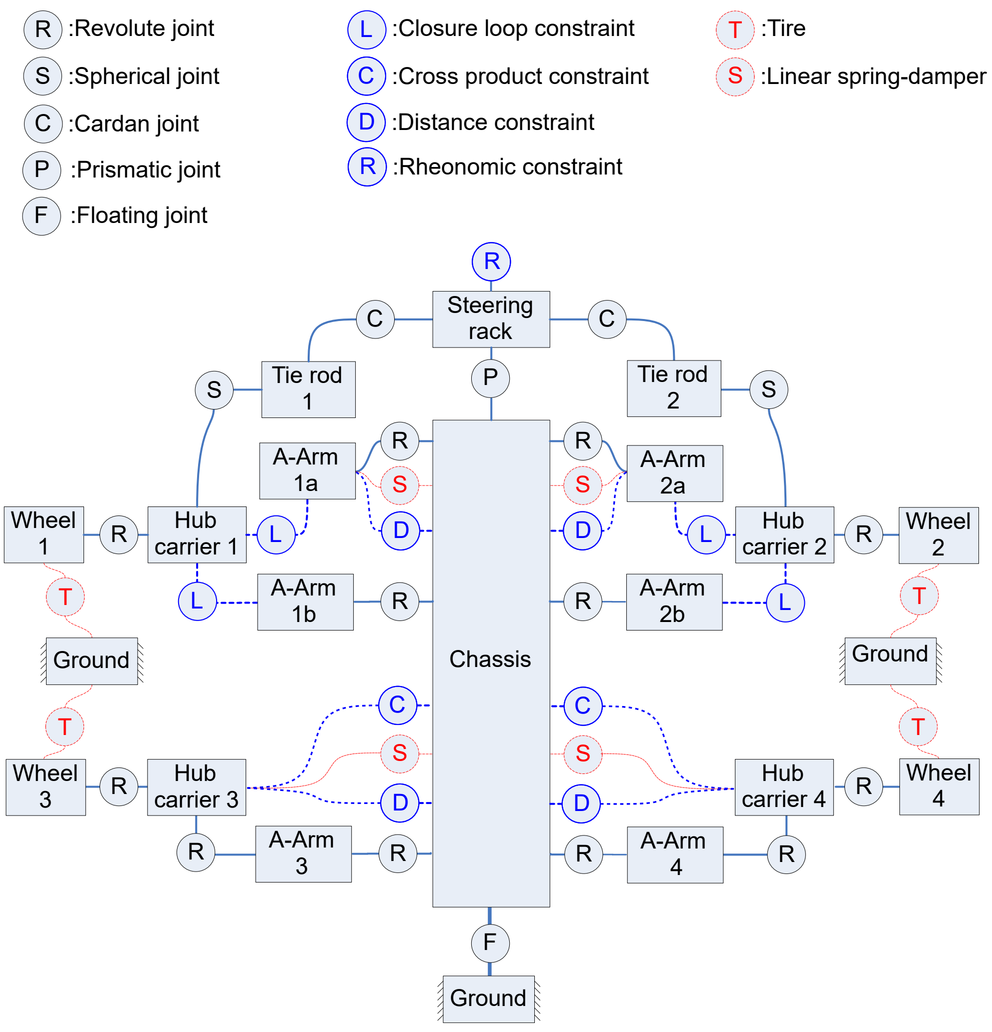


Figure 7.22: Kinematic joints, bodies and constraints of the topological model

4.5 $s$, no additional tractor forces are applied and the steering system is blocked with the front wheels straight.

For the sensitivity analysis of this maneuver, the following objective function is considered:

$$\psi = \int_{t_0}^{t_F} \ddot{r}_{1_z}^2 \, \mathrm{d}t \tag{7.33}$$

In which $\ddot{r}_{1_z}$ denotes the Z-component of the acceleration of point 1, located in the chassis. This objective function represents a measure of comfort, since it is evaluating the accelerations that the driver would experiment during the descent of the step, and is related to vibration measurements according to norm ISO 2631-1.

#### 7.3.1.2 Second maneuver: double lane change

The second maneuver lasts $12\,s$ and consists in a double lane change according to norm ISO 3888-1:1999. This norm determines the dimensions of the "test track for a severe lane-change manoeuvre" [139], and it has been employed to establish the limits of the maneuver for a vehicle width of $1.625\,m$ (which is the distance between the exterior side of each one of the front wheels). An user guide function, displayed in 7.23, has been adjusted to comply with the boundaries of the test track.

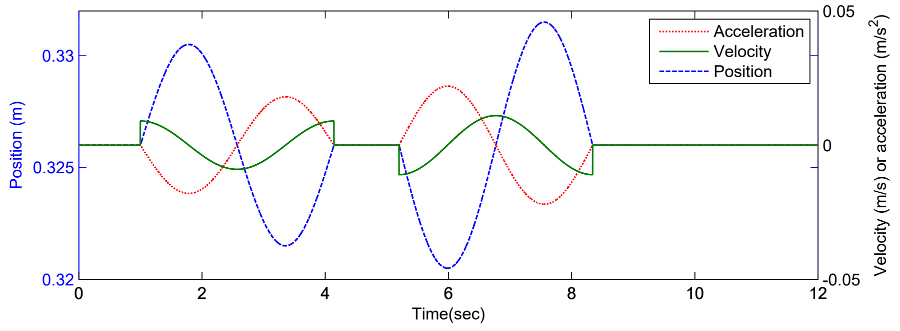


Figure 7.23: Driving function for the steering rack.

In this maneuver no tractor forces have been considered, thus motion is exclusively due to the inertial forces related to the initial velocity of the chassis and the wheels. In this regard, the initial velocities are 11.9 $m/s$ for the chassis in the forward direction, and 46 $rad/s$ for the wheels around their spin axis.

The lateral and inertial forces appearing in this maneuver entail a change in the roll $\phi$ of the chassis. The objective function of this maneuver measures the sum of squares of the roll rate over time. The gradient of this objective function will deliver a measure of the impact of each one of the parameters in the roll rate, and they could be eventually used to minimize the roll rate in this maneuver.

$$\psi = \int_{t_0}^{t_F} \dot{\phi}^2 \mathrm{d}t \tag{7.34}$$

### 7.3.2 Numerical results: dynamics

The first maneuver has been executed with a fixed time step of 1 millisecond, while the double-lane change maneuver has been simulated with time intervals of 10 milliseconds. The numerical integrator used in both maneuvers is the Newmark integrator with $\beta = 0.25$ and $\gamma = 0.5$, which is identical to the implicit trapezoidal rule. In Figure 7.24, a capture of the graphical interface configured for the vehicle dynamic simulation can be observed.

Concerning to semi-recursive Matrix R formulations, the degrees of freedom of the model include a set of angles and distances identified with joint coordinates, and another set of point and vector coordinates that is not included in the dependent relative coordinates vector. Therefore, a non-constant **B** matrix containing the variation of

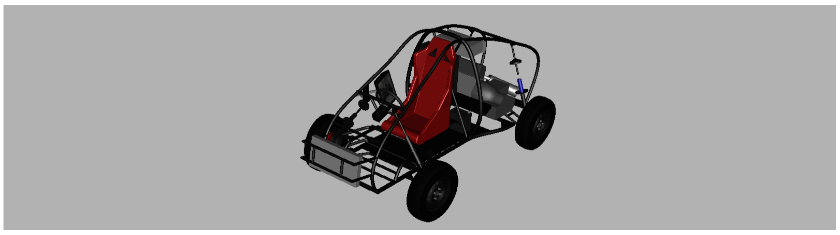


Figure 7.24: Graphical interface for the simulation of the buggy vehicle.

the degrees of freedom with respect to the dependent coordinates (see section 3.2.1) is also tested with this model.

#### 7.3.2.1 First maneuver: step descent

The descent of a step of 1 $cm$ entails a change in the position, velocity and accelerations of the $Z$ component of each point of the model, while the imposition of straight front wheels implies that any change in the $Y$ direction will be negligible. In this maneuver, the $X$ and $Z$ coordinates of point 1 belonging to the chassis are examined in terms of position, velocity and acceleration.

Results for the four semi-recursive formulations tested along with the reference results obtained with a global Matrix R formulation in natural coordinates, are displayed in Figure 7.25. Despite the completely different models (natural and joint coordinate models), reference point selection (RTdyn0 and RTdyn1) and formulations (Matrix R and ALI3-P), Figure 7.25 demonstrates the validity of the theoretical developments for semi-recursive formulations along with their implementation in MBSLIM.

| Formulation | CPU time | CPU time natural | Ratio $nat/rel$ |
|---|---:|---:|---:|
| RTdyn0 ALI3-P | 2.781 | 6.703 | 2.410 |
| RTdyn1 ALI3-P | 2.781 | 6.703 | 2.410 |
| RTdyn0 MatrixR | 5.422 | 9.781 | 1.804 |
| RTdyn1 MatrixR | 5.422 | 9.781 | 1.804 |

Table 7.16: CPU time of relative coordinates formulations compared with the equivalent one in natural coordinates.

The formulations tested in this maneuver are compared in terms of computational effort in Table 7.16. This table evidences the significant reduction in CPU time of relative coordinate formulations compared with the equivalent formulations in natural coordinates. It should be noted that all dynamic simulations have been executed considering a very low error tolerance ($10^{-10}$) for the Newton-Raphson iteration loop. For other tolerances, such as $10^{-7}$, the ratio between natural and relative coordinates solutions reaches 3.50 in ALI3-P formulations and 1.54 in Matrix R formulations, but in that case, the reduction in CPU time of semi-recursive formulations came from a lower number of iterations. For more demanding tolerances, like the one used to

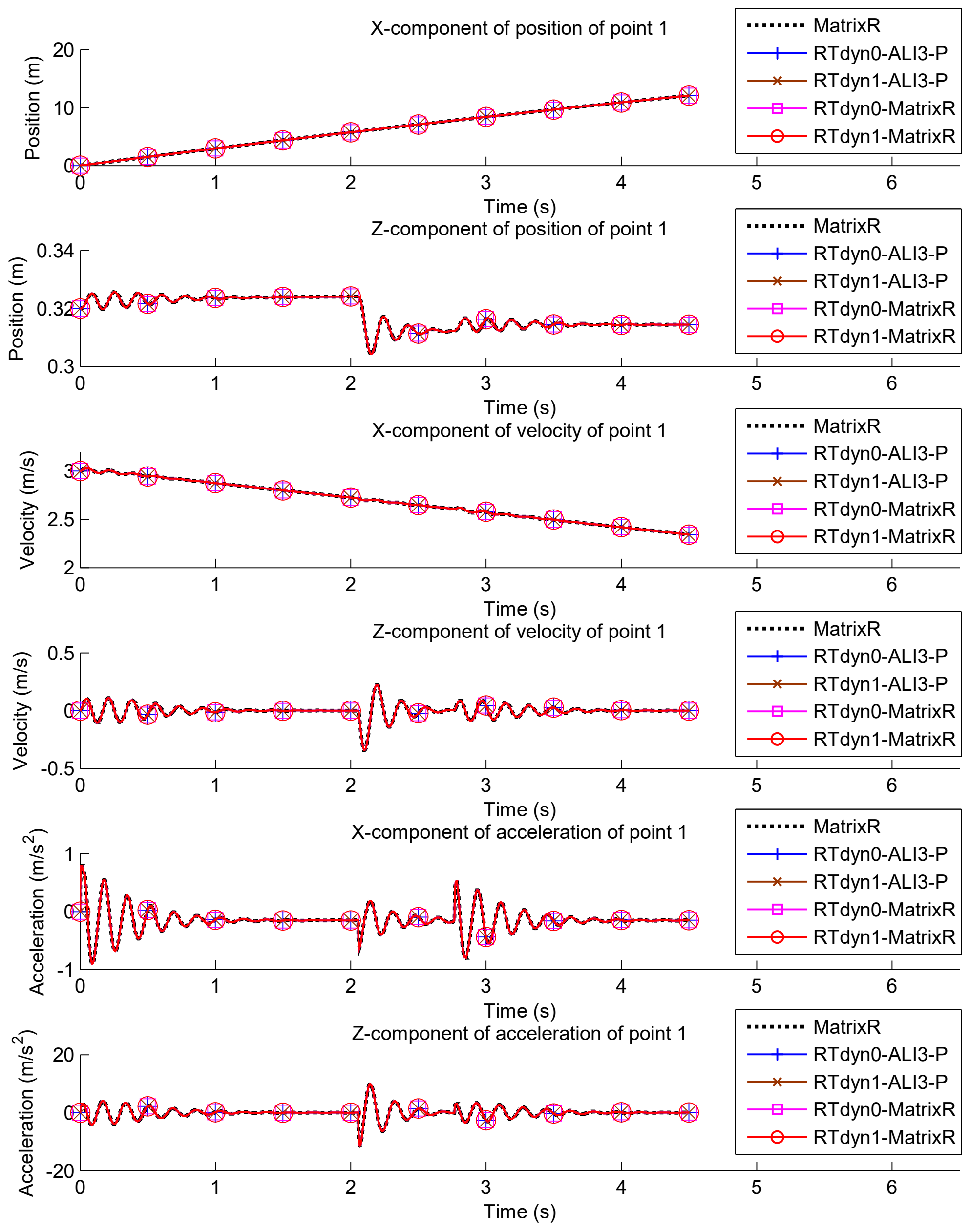


Figure 7.25: Position, velocity and acceleration of point 1 belonging to the chassis.

obtain the results of Table 7.16 and Figure 7.25, the number of iterations is very similar among formulations.

#### 7.3.2.2 Second maneuver: double lane change

A double lane change is a smoother maneuver than the step descent as long as there are no strong impacts, collisions or abrupt changes in forces or constraints. Therefore, both the dynamics and its sensitivities can be computed with an increased time step. In this case, the semi-recursive and global formulations are executed with a time step of 10 milliseconds.

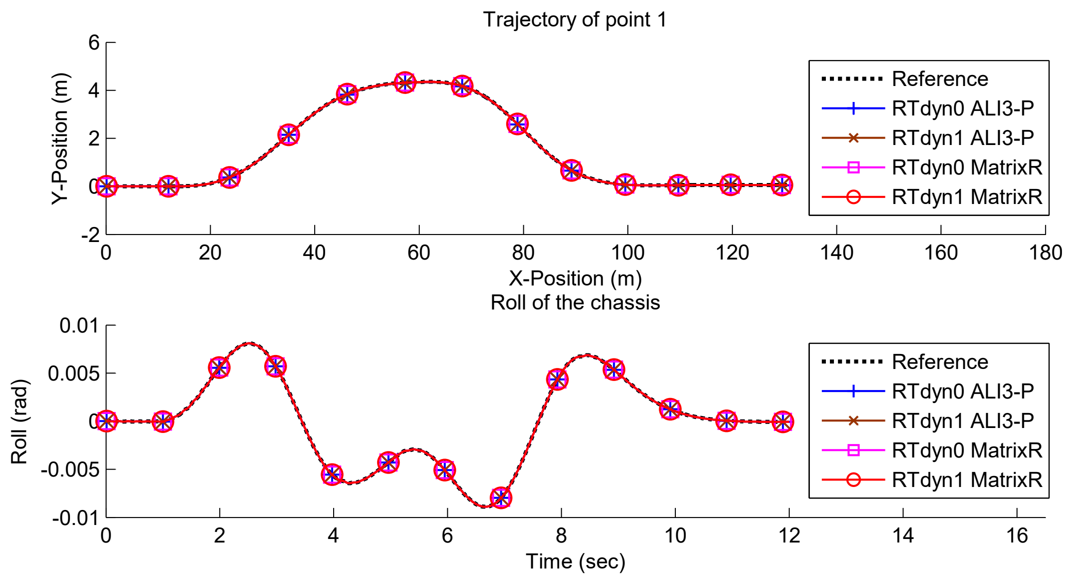


Figure 7.26: Trajectory of point 1 in the XY plane and roll of the chassis in the double lane change maneuver

In Figure 7.26, the trajectory in plane $XY$ of point 1 located at the frontal part of the chassis along with the roll of the chassis over time are displayed for the 4 semi-recursive formulations tested as well as for the reference solution. This figure reflects the great level of convergence of all joint coordinate formulations tested in this maneuver, even for this relatively high simulation time. Moreover, it is remarkable that two radically different models, such as the one expressed in terms of natural coordinates and the joint coordinate model, yield such similar behaviors.

| Formulation | CPU time | CPU time natural | Ratio $nat/rel$ |
|---|---|---|---|
| RTdyn0 ALI3-P | 1.047 | 3.750 | 3.582 |
| RTdyn1 ALI3-P | 1.047 | 3.750 | 3.582 |
| RTdyn0 MatrixR | 2.109 | 3.859 | 1.830 |
| RTdyn1 MatrixR | 2.109 | 3.859 | 1.830 |

Table 7.17: CPU time of semi-recursive dynamic formulations compared with the equivalent natural coordinate formulation in a double lane-change maneuver.

Semi-recursive formulations are compared with the equivalent global formulation in terms of computational effort in Table 7.17. Behold that semi-recursive ALI3-P formulations outperform any global formulation, and they are also two times faster than semi-recursive Matrix R formulations. Moreover, it should be remarked that semi-recursive Matrix R formulations are more efficient than the equivalent formulation in natural coordinates, even with the shortcoming of considering a non-constant **B** matrix (stemming from the user definition of Cartesian coordinates as degrees of freedom). With regard to semi-recursive formulations, it is clear that ALI3-P outperform Matrix R formulations in what refers to computational effort in this particular maneuver.

### 7.3.3 Numerical results: sensitivity analysis

The parameters selected for the sensitivity analysis in both maneuvers are the stiffness and damping coefficients of the four suspensions (equal values are regarded for the spring-dampers of each axle) along with the mass of the frame.

$$\boldsymbol{\rho}^{sens} = \begin{bmatrix} k_f & c_f & k_r & c_r & m_c \end{bmatrix}^{\mathrm{T}} \tag{7.35}$$

where $k_f$ and $c_f$ are the stiffness and damping coefficients of the front suspensions, $k_r$ and $c_r$ denote the stiffness and damping coefficients of the rear suspensions and $m_c$ represents the mass of the chassis.

#### 7.3.3.1 First maneuver: step descent

The gradient of the objective function displayed in Figure 7.27 is evaluated by means of the semi-recursive forward and adjoint sensitivity formulations described in chapter 5. The gradient results presented in Figures 7.31 and 7.32 for forward and adjoint methods respectively, reflect an excellent convergence of semi-recursive sensitivity formulations with the reference values obtained through the Matrix R forward and adjoint sensitivity formulation in natural coordinates.

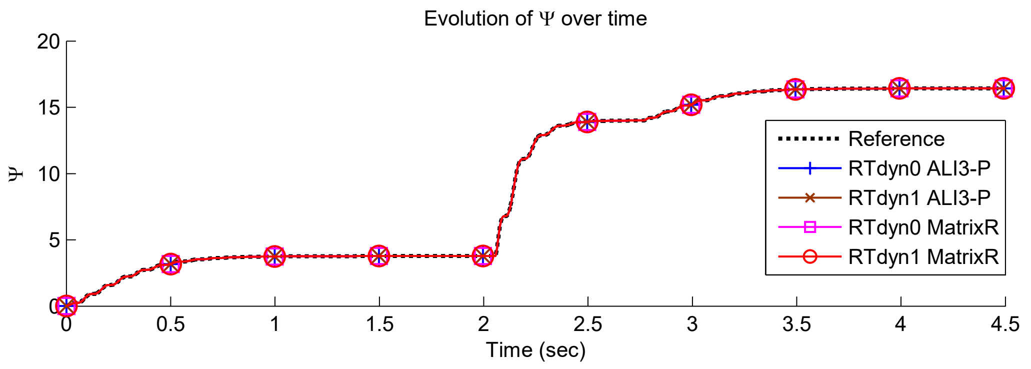


Figure 7.27: Evolution of the objective function over time on a step descent maneuver.

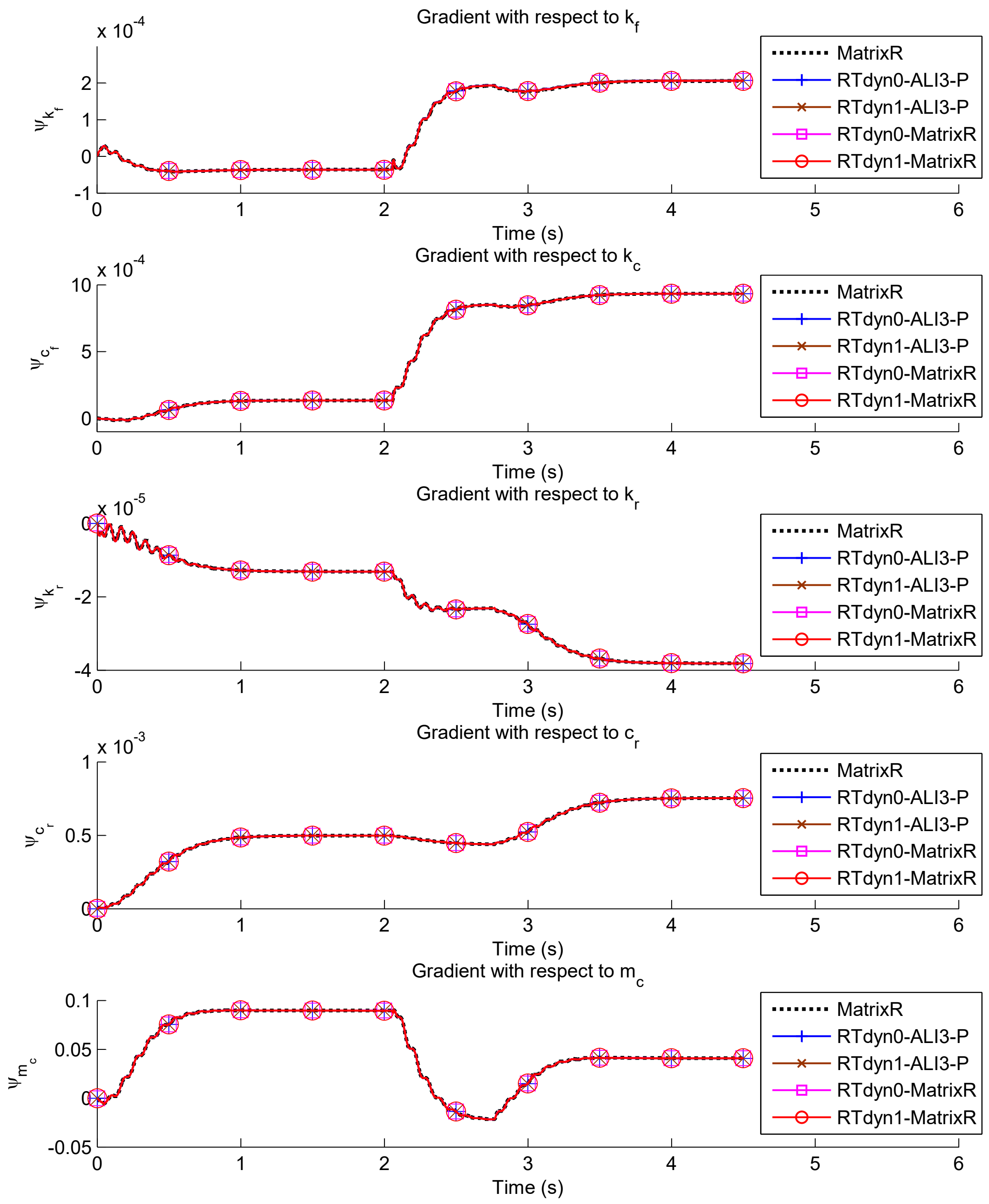


Figure 7.28: Gradient of the step-descent objective function using forward semi-recursive sensitivity formulations.

Paying attention to Table 7.18, it is clear that all the semi-recursive formulations display better performance than the equivalent ones in natural coordinates in terms of computational time. However, the time saving is not similar to the one obtained in the dynamics due to, on the one hand, the increase in the complexity in the differen-

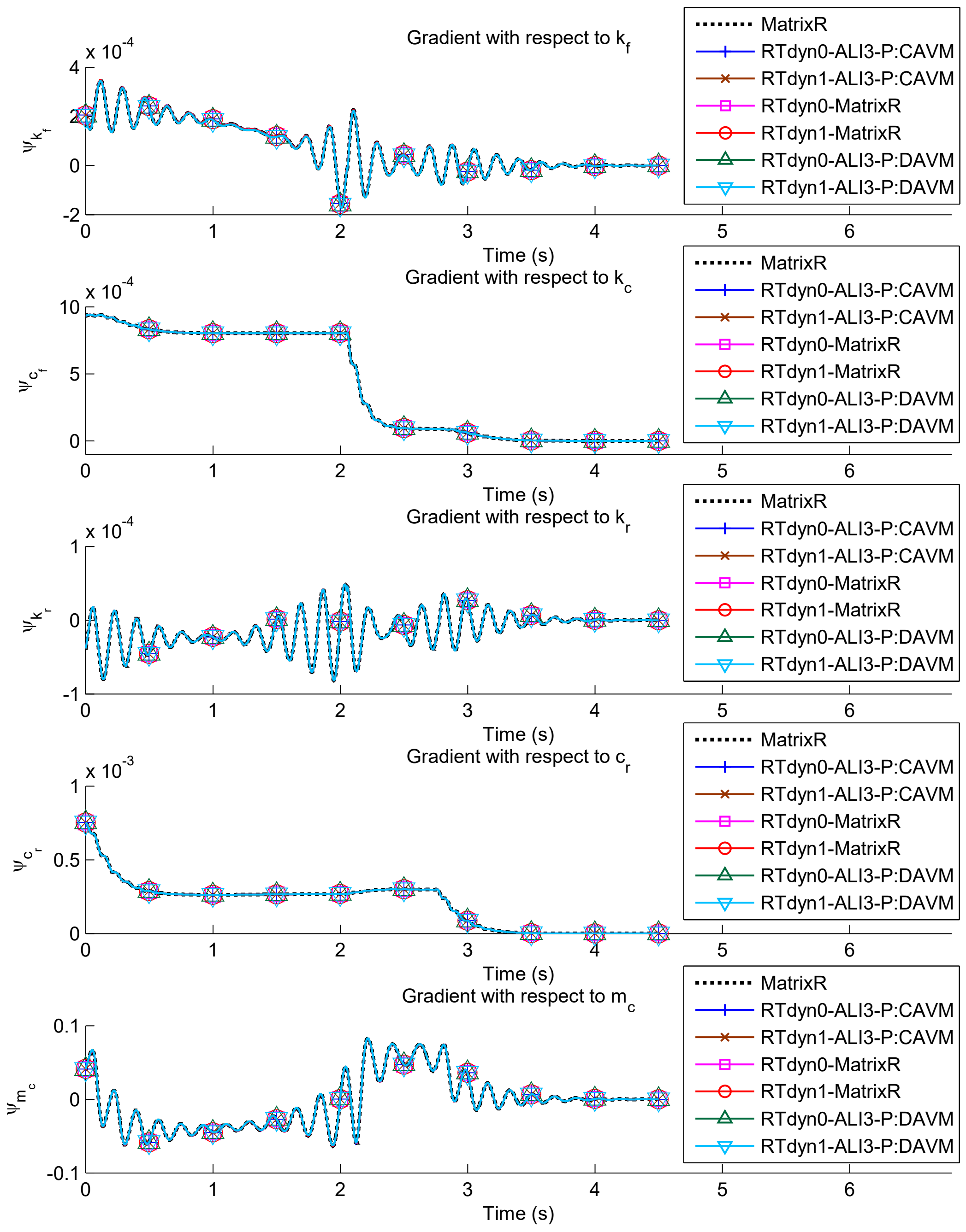


Figure 7.29: Gradient of the step-descent objective function using adjoint semi-recursive sensitivity formulations.

tiation, involving concatenations of products and derivatives, and on the other hand, due to the non-constant mass matrix which is assembled and differentiated in the semi-recursive sensitivity formulations at each time step, while in natural coordinates models it is always constant, and its derivative null.

| Formulation | CPU time | CPU time natural | Ratio *nat/rel* |
|---|---:|---:|---:|
| RTdyn0 ALI3-P: DDM | 8.188 | 9.484 | 1.158 |
| RTdyn0 ALI3-P: CAVM | 9.031 | 10.953 | 1.213 |
| RTdyn0 ALI3-P: DAVM | 8.781 | 11.047 | 1.258 |
| RTdyn1 ALI3-P: DDM | 8.188 | 9.484 | 1.158 |
| RTdyn1 ALI3-P: CAVM | 9.031 | 10.953 | 1.213 |
| RTdyn1 ALI3-P: DAVM | 8.781 | 11.047 | 1.258 |
| RTdyn0 MatrixR: DDM | 30.156 | 35.563 | 1.179 |
| RTdyn0 MatrixR: CAVM | 30.350 | 37.719 | 1.243 |
| RTdyn1 MatrixR: DDM | 30.156 | 35.563 | 1.179 |
| RTdyn1 MatrixR: CAVM | 30.250 | 37.719 | 1.243 |

Table 7.18: CPU time of semi-recursive sensitivity formulations compared with the equivalent one in natural coordinates for the step-descent maneuver

These two drawbacks entail a reduction in the efficiency of the semi-recursive sensitivity analysis formulations, even though they are in average a 20.9% faster in ALI3-P models than the equivalent formulation in natural coordinates, and 21.1% faster in Matrix R formulations, according to Table 7.18. Observe that despite semi-recursive Matrix R formulations show a slightly better CPU time ratio between natural and relative coordinate formulations, semi-recursive ALI3-P sensitivity formulations are still faster for every set of coordinates and every sensitivity method.

The problem related to the computation and assembly of the derivatives of the mass matrix can be simplified considering the evolution of the problem and the maneuver being simulated. The variation of the mass matrix with respect to the relative coordinates is, in general, a magnitude that varies slowly, so its values between two instants of time are almost equal. Accordingly, the possibility of evaluating $\mathbf{M}^d_{\mathbf{z}}$ each several time steps has emerged, and it has been proved in this maneuver. The results in terms of computational time are displayed in Table 7.19 .

If the sensitivity results with the evaluation of $\mathbf{M}^d_{\mathbf{z}}$ every two time steps are compared with the original solution, the maximum percentage of deviation found will be lower than 0.05% for each one of the formulations. Regarding that the maximum differences obtained among formulations are between 1 to 2 %, it can be stated that the error introduced by not computing $\mathbf{M}^d_{\mathbf{z}}$ each time step is not significant compared with the differences in the computation of each formulation.

Now, contrasting the results of Tables 7.18 and 7.19, a significant increase in the ratio of CPU times between natural and relative coordinates formulations is observed in ALI3-P methods. On the other hand, since the reduction in time is equal for each formulation, this effect has a lower impact on Matrix R formulations.

| Formulation | CPU time | CPU time natural | Ratio $nat/rel$ |
|---|---:|---:|---:|
| RTdyn0 ALI3-P: DDM | 7.313 | 9.484 | 1.297 |
| RTdyn0 ALI3-P: CAVM | 8.078 | 10.953 | 1.356 |
| RTdyn0 ALI3-P: DAVM | 7.906 | 11.047 | 1.397 |
| RTdyn1 ALI3-P: DDM | 7.313 | 9.484 | 1.297 |
| RTdyn1 ALI3-P: CAVM | 8.078 | 10.953 | 1.356 |
| RTdyn1 ALI3-P: DAVM | 7.906 | 11.047 | 1.397 |
| RTdyn0 MatrixR: DDM | 29.219 | 35.563 | 1.217 |
| RTdyn0 MatrixR: CAVM | 29.516 | 37.719 | 1.278 |
| RTdyn1 MatrixR: DDM | 29.219 | 35.563 | 1.217 |
| RTdyn1 MatrixR: CAVM | 29.516 | 37.719 | 1.278 |

Table 7.19: CPU time of semi-recursive sensitivity formulations compared with the equivalent one in natural coordinates for the step-descent maneuver with $\mathbf{M}^d_{\mathbf{\hat{z}}}$ evaluated each 2 time steps.

#### 7.3.3.2 Second maneuver: double lane change

The double lane change described in section 7.3.1.2 and simulated in section 7.3.2.2 is analyzed in terms of sensitivity in this section, considering the objective function (7.34) measuring the integral of the square of the roll rate over time. The sensitivity analyses of the dynamic formulations used in section 7.3.2.2 are here accomplished by means of the analytical direct and adjoint sensitivity formulations presented in chapter 5.

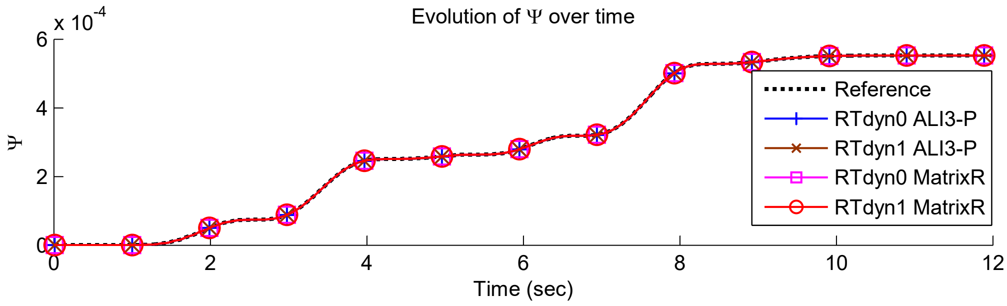


Figure 7.30: Evolution of the objective function over time on a double-lane-change maneuver.

First of all, let us assess the differences among objective function values evaluated through different dynamic formulations. Figure 7.30 shows that all the semi-recursive dynamic formulations converge to identical values of the objective function, and converge with the expected reference value.

As stated in section 7.3.2.2, this maneuver is smoother than a step descent since no discrete impact forces are present. Accordingly, this maneuver can be executed with a higher time step (10 milliseconds) without impairing accuracy, as proved for the dynamics. In the case of sensitivity analysis, a bigger divergence could be expected

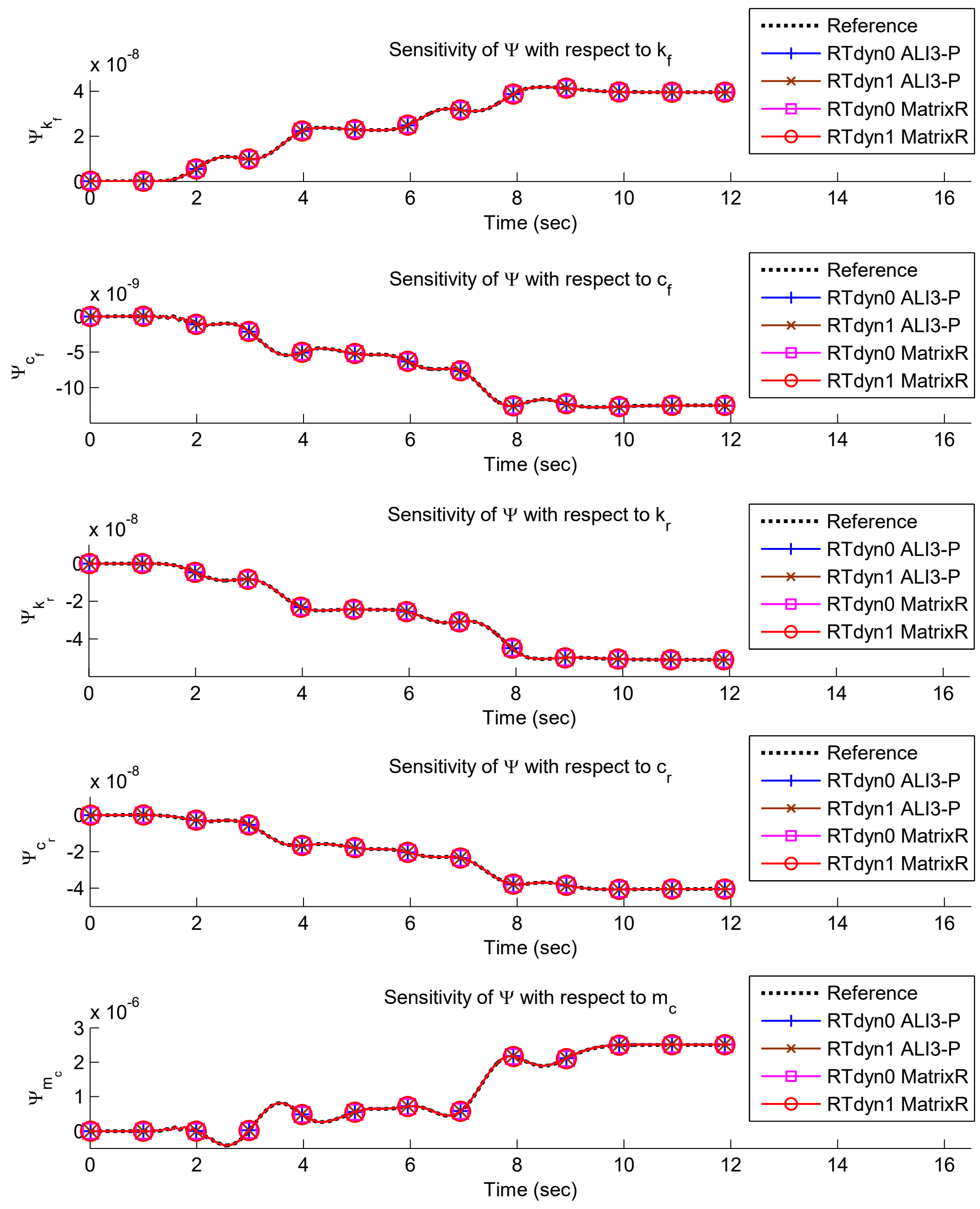


Figure 7.31: Gradient of the double-lane-change objective function using forward semi-recursive sensitivity formulations.

regarding that small variations in the dynamics are amplified in any sensitivity analysis. Fortunately, those sensitivity divergences are minimal, as it can be observed in Figures 7.31 and 7.32, comprising the evaluation of the gradient of the objective function (7.34) with the direct and adjoint sensitivity formulations presented in chapter 5.

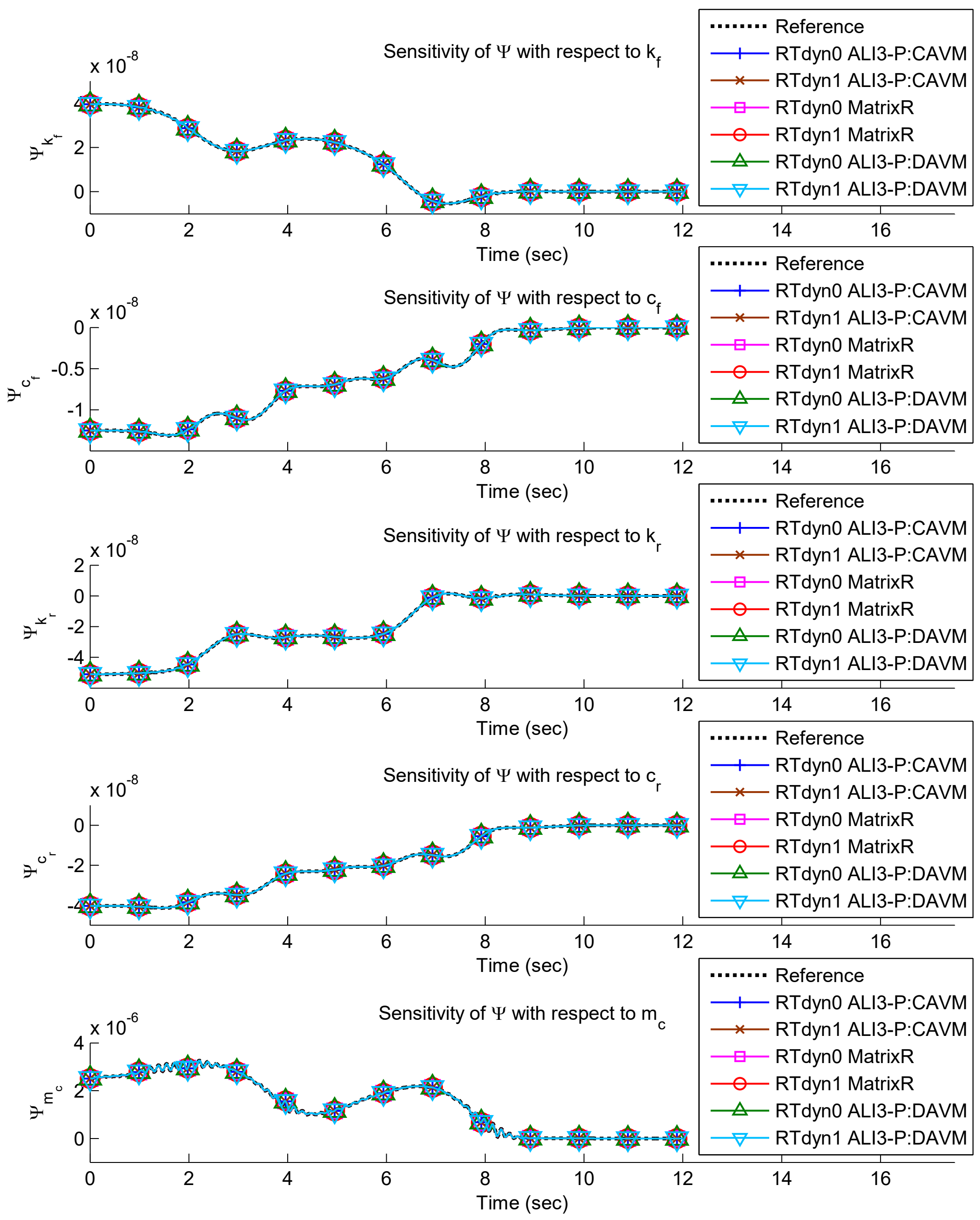


Figure 7.32: Gradient of the double-lane-change objective function using adjoint semi-recursive sensitivity formulations.

Behold that, in these two figures, ten semi-recursive sensitivity formulations have been tested, involving 2 different accumulation schemes (RTdyn0 and RTdyn1), two different constraints enforcements (Matrix R and ALI3-P), two different analytical differentiation methods (direct and adjoint variable methods) and even two different discretization philosophies (discrete and continuous adjoint variable methods). Above all, it should be remarked that this problem involves 18 bodies, 5 different types of joints, several constraint equations and a branched tree topology, thus most of the derivatives and schemes of differentiation for the composition of the semi-recursive sensitivity equations presented in this work have been proved in this example.

| Formulation | CPU time | CPU time natural | Ratio $nat/rel$ |
|---|---|---|---|
| RTdyn0 ALI3-P: DDM | 2.578 | 4.609 | 1.788 |
| RTdyn0 ALI3-P: CAVM | 2.766 | - | - |
| RTdyn0 ALI3-P: DAVM | 2.686 | 4.906 | 1.827 |
| RTdyn1 ALI3-P: DDM | 2.578 | 4.609 | 1.788 |
| RTdyn1 ALI3-P: CAVM | 2.766 | - | - |
| RTdyn1 ALI3-P: DAVM | 2.686 | 4.906 | 1.827 |
| RTdyn0 MatrixR: DDM | 8.484 | 11.609 | 1.368 |
| RTdyn0 MatrixR: CAVM | 8.563 | 11.969 | 1.404 |
| RTdyn1 MatrixR: DDM | 8.484 | 11.609 | 1.368 |
| RTdyn1 MatrixR: CAVM | 8.563 | 11.969 | 1.404 |

Table 7.20: CPU time of semi-recursive sensitivity formulations compared with the equivalent one in natural coordinates for the double-lane-change maneuver

Additionally, the case of semi-recursive Matrix R formulations requires a special mention. In this particular model, the degrees of freedom are a combination of angles and distances assimilable to joint coordinates along with some Cartesian coordinates of points and vectors. The inclusion of Cartesian coordinates as degrees of freedom is accomplished in this work without an imposition of additional constraints or without enlarging the dependent coordinates vector. Instead, a new variable **B** matrix has been defined, and both the dynamic equations of motion and the sensitivity expressions (both direct and adjoint variable methods) have been developed under this variability assumption. The buggy dynamic and sensitivity simulations validate both the theoretical developments and implementation of the sensitivity formulations and, in particular, the modified semi-recursive formulation for non-constant **B** matrices.

Let us inspect now the computational efficiency of each sensitivity evaluation paying attention to the CPU times required by each sensitivity formulation to compute the gradient of the objective function (7.34) for the double lane change maneuver, displayed in Table 7.20. Semi-recursive sensitivity formulations are more efficient than global formulations in any case, being semi-recursive ALI3-P formulations the fastest. The computational gains in semi-recursive formulations are around a 79.6% for semi-recursive ALI3-P sensitivity formulations and 38.6% for semi-recursive Matrix R sensitivity formulations (considering the average gains for RTdyn0 and RTdyn1 forward and adjoint sensitivity formulations). However, accuracy of global ALI3-P formulations is poorer than semi-recursive ALI3-P formulations in this case, which implies that the time step should be decreased for reaching the same accuracy level of semi-recursive formulations, with the corresponding impact on the computational cost. The computational burden of including a variable **B** matrix does not constitute a significant harm to the efficiency of the method. In fact, semi-recursive Matrix R sensitivity formulations require lower computational effort than global Matrix R sensitivity methods.

Comparing direct and adjoint differentiation methods, Table 7.20 shows a minimum difference in the computational expense, being slightly more efficient forward sensitivity formulations. The efficiency of adjoint variable methods is expected to grow with the increase in the number of parameters. With regards to the order between discretization and differentiation, discrete adjoint variable methods applied to semi-recursive and global ALI3-P formulations perform better that continuous methods, being even not possible to obtain accurate results with the continuous approach in natural coordinates for the time step selected.

### 7.3.4 Numerical results: design optimization

The objective function gradients obtained in section 7.3.3 by means of analytical methods are extremely useful in the optimization of the dynamics of multibody systems. It has been proved by several authors that optimization algorithms that handle the derivatives of the objective function usually deliver better performance than those which are based on the unique evaluation of objective function values.

In this section, the optimization of the design of the buggy vehicle is accomplished for the step descent and double lane change maneuvers. Each optimization problem is described as a minimization problem which seeks to obtain the set of parameter values that conquests the minimum value of an objective function. The solution of those problems has been successfully reached by means of third party gradient-based optimization algorithms, such as L-BFGS-B or the optimization algorithms included in the $fmincon$ Matlab function.

The set of parameters considered in both maneuvers are the stiffness and damping coefficients of the spring-damper suspension forces, as stated in (7.36). The aim of these problems is to optimize the suspensions, thus the mass of the chassis included as a sensitivity parameter in section 7.3.3 (according to (7.35)) does not play any role in the optimization.

The optimal design problems related to both maneuvers consist in the determination of the set of stiffness and damping coefficients of each suspension which minimizes the value of the corresponding objective function. The set of optimum design parameters are, thus:

$$\boldsymbol{\rho}^{opt} = \begin{bmatrix} k_f & c_f & k_r & c_r \end{bmatrix}^{\mathrm{T}} \tag{7.36}$$

Each optimization problem is constrained by some lower boundaries in the forces coefficients regarding that they could not have negative values. The upper limits are selected more arbitrarily in order to establish a range of usual or possible values of this type of mechanical elements. Both lower (**L**) and upper (**U**) boundaries are specified in (7.37).

$$\mathbf{L} = \begin{bmatrix}10 & 10 & 10 & 10\end{bmatrix}^{\mathrm{T}} \tag{7.37a}$$

$$\mathbf{U} = \begin{bmatrix}10^5 & 10^5 & 10^5 & 10^5\end{bmatrix}^{\mathrm{T}} \tag{7.37b}$$

Once proved that all the sensitivity formulations included in this paper reach the same gradient results, no explicit mention would be made to the dynamic and sensitivity analysis method used, being the optimization results applicable to any of those formulations. Moreover, no computational time is evaluated for any optimization, since it is directly related to the sensitivity formulation selected. Instead, the number of iterations is presented, and the computational effort can be inferred from this number and the CPU time relations presented in section 7.3.3 .

Each optimization problem unfolds as follows: first, the design optimization problem is described as a minimization problem; second, the optimization results are presented in terms of number of iterations and local optimum reached; and third, results are discussed.

#### 7.3.4.1 First maneuver: step descent

The step descent maneuver reflects the behavior of the vehicle in the presence of reduced obstacles or irregular pavements. In general, the suspensions of a vehicle are designed so as to minimize the effects of road surface irregularities experimented by the pilot, thus reducing vibrations transferred to the chassis. The objective function (7.33) considered in this maneuver is related to vibration measurements according to norm ISO 2631-1, thus their optimization will lead to minimum vibrations, and thus, to maximum pilot comfort.

The design optimization problem can be stated as a minimization problem with the form:

$$\min_{\boldsymbol{\rho}} \quad \psi = \int_{t_0}^{t_F} \ddot{r}_{1_z}^2 \, \mathrm{d}t \tag{7.38}$$

$$\text{s.t.} \quad \boldsymbol{\rho} \leq \mathbf{U} \tag{7.39}$$

$$\boldsymbol{\rho} \geq \mathbf{L} \tag{7.40}$$

with **U** and **L** being the upper and lower limits of the parameter values defined in (7.37). Behold that the lower limits **L** are indispensable since stiffness of damping coefficients cannot have negative values, while the upper boundaries **U** are free to be chosen by the user.

The initial values of the parameters at the onset of the optimization process are those used for the previous dynamic and sensitivity analyses. They are specified in Table 7.14. However, in order to prove the validity of the results obtained, other different starting configurations have been also tested, but they are not included here for the sake of clarity.

All the optimizations have been executed with a maximum number of optimization iterations of 100, a maximum number or objective function evaluations of 1000 and a stop criteria of the 2-norm of the parameter increment lower than $10^{-8}$. Table 7.21 gathers the optimization results obtained with each one of the optimization algorithms supported by the *fmincon* function of Matlab. According to that table, it is clear that Active Set along with SQP algorithms require the fewest number of optimization iterations to reach an optimum according to the stop criteria. Nevertheless, the Interior Point algorithm requires more iterations but can reach lower values of the objective function, thus attaining a better local optimum. Trust region reflective (TRR) optimization algorithms deliver the worst performance in this case, with a high number of iterations (it has been stopped by the maximum optimization iterations criterion) and with a high objective function value.

| Algorithm | $k_f$ | $c_f$ | $k_r$ | $c_r$ | $\Psi$ | Nº iter |
|---|---|---|---|---|---|---|
| Interior Point | 13731.4 | 245.896 | 13434.3 | 59.7709 | 1.4146 | 80 |
| TRR | 15217.6 | 7911.09 | 14514.9 | 4724.76 | 13.164 | 100 (max) |
| SQP | 13886.8 | 257.488 | 10962.8 | 96.9401 | 1.4601 | 30 |
| Active Set | 13923.9 | 251.196 | 10961.4 | 96.8143 | 1.4619 | 19 |

Table 7.21: Optimization results for the step-descent maneuver using the *fmincon* tool of Matlab

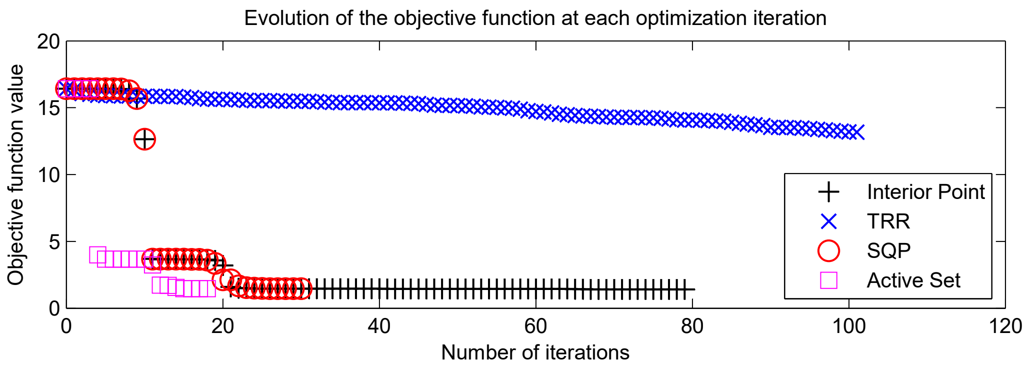


Figure 7.33: Evolution of objective function value per optimization iteration with the four algorithms available on *fmincon* for the step-descent maneuver.

In order to check the validity of the local optimum reached, the initial values of the coefficients have been modified, reaching in all the tests executed similar or worse final function values. Even though, it cannot be firmly declared that the solution reached with the Interior Point algorithm according to Table 7.21 is a global optimum. It can be only stated that it delivers the lowest of the minimum objective function values obtained.

Moreover, in Figure 7.33 the evolution of the objective function value at each iteration step is displayed for each one of the optimization algorithms proved. This figure sheds light on the process followed by each algorithm to reach the final optimum. While Interior Point methods entail the higher number of iterations, it is clear from this figure that they reach a value close to the final solution in about 21 iterations, and then continues iterating to reach a slightly lower value. On the contrary, Active Set and SQP algorithms attain lower values of the objective function in fewer iterations than Interior Point algorithms but they are stopped before the 20th and 31st iteration respectively.

In Figure 7.34, the evolution of the Z coordinate of the acceleration of point 1 (belonging to the chassis) over time is displayed for the original and the optimized set of parameters. It is clear from this figure that the optimized set of parameters is able to reduce the absolute values of the acceleration peaks during the stabilization of the suspensions (first second of the simulation) and during the descent of the step (after $t = 2\,sec$).

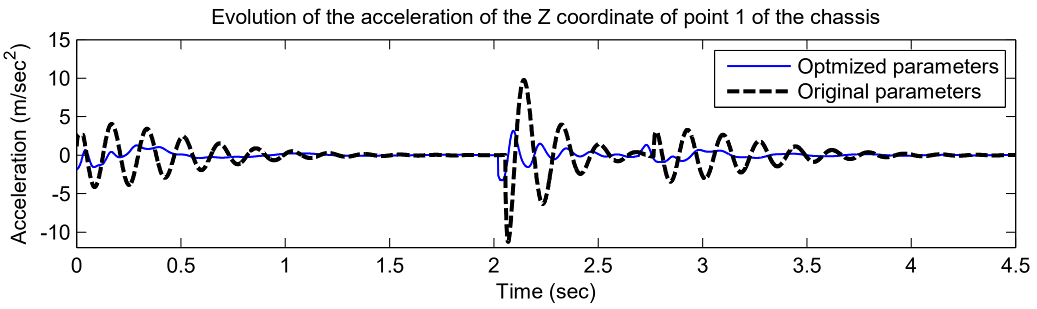


Figure 7.34: Evolution of the acceleration of the Z coordinate of point 1 over time on a step descent maneuver for the original and the optimized set of parameters.

#### 7.3.4.2 Second maneuver: double lane change

In this case, the design optimization problem seeks to optimize the suspensions for the double lane change maneuver in order to minimize the roll rate experimented by the chassis. In this regard, the optimization parameters are the same of the former maneuver, but the target is completely different. It should be reminded that right and left suspension forces coefficients are assumed to be equal, while they vary between front and rear axles.

The minimization problem stemmed from the suspension forces optimization can be stated as:

$$\min_{\boldsymbol{\rho}} \quad \psi = \int_{t_0}^{t_F} \dot{\phi}^2 \mathrm{d}t \tag{7.41}$$

$$\text{s.t.} \quad \boldsymbol{\rho} \leq \mathbf{U} \tag{7.42}$$

$$\boldsymbol{\rho} \geq \mathbf{L} \tag{7.43}$$

with $\mathbf{U}$ and $\mathbf{L}$ defined in (7.37).

However, it has been made patent in section 7.3.3.2 that the objective function and its gradient have very low values, thus it is convenient to scale the objective function in order to enhance the numerical conditioning of the optimization. Consequently, the minimization problem has been reformulated as:

$$\min_{\boldsymbol{\rho}} \quad \psi = k \int_{t_0}^{t_F} \dot{\phi}^2 \mathrm{d}t \tag{7.44}$$

$$\text{s.t.} \quad \boldsymbol{\rho} \leq \mathbf{U} \tag{7.45}$$

$$\boldsymbol{\rho} \geq \mathbf{L} \tag{7.46}$$

with the penalty factor $k = 10^5$. Behold that the scaling factor affects both the value of the objective function and its gradient.

| Algorithm | $k_f$ | $c_f$ | $k_r$ | $c_r$ | $\Psi$ | Nº iter |
|---|---|---|---|---|---|---|
| Interior Point | 11643.9 | 23308.5 | 20635.9 | 28819.1 | 7.7677 | 33 |
| TRR | 2987.53 | 14931.6 | 681.023 | 19643.2 | 1.8566 | 100(max) |
| SQP | 11646.4 | 23309.1 | 20632.1 | 28818.4 | 7.7677 | 34 |
| Active set | 515.281 | 11489.7 | 201.502 | 10047.4 | 1.2357 | 90 |

Table 7.22: Optimization results for the double-lane-change maneuver using the *fmincon* tool of Matlab

In this case, the optimization results must be carefully inspected. The minimization of the roll rate seeks to lower the center of mass so that it matches the roll center of the vehicle (see [140]). Looking at Table 7.22, the stiffness coefficients of front and rear suspensions are decreased in the optimization process in order to reduce the spring forces and lower the chassis. The optimization behaves as expected in this regard, but lowering the chassis conveys an undesired effect on the steering relations and on the distance between chassis and ground.

Instead of optimizing the suspensions, let us evaluate what is the optimal value of the local Z-position of the center of mass that minimizes the roll rate, with:

$$\min_{\bar{r}_z^G} \quad \boldsymbol{\psi} = k \int_{t_0}^{t_F} \dot{\phi}^2 \mathrm{d}t \tag{7.47}$$

$$\text{s.t.} \quad \bar{r}_z^G \leq 1 \tag{7.48}$$

$$\bar{r}_z^G \geq -1 \tag{7.49}$$

wherein the penalty factor is $k = 10^5$. This minimization problem is more direct, and can be solved in a few iterations with any of the previous optimization algorithms included in *fmincon*, according to Table 7.23 and Figure 7.35.

It should be noted that the local Z axis of the chassis and the global Z axis have opposite directions, thus an increase in the local Z-component lowers the CoM in the absolute reference frame. Oversimplifying, the optimization concludes that center of mass of the chassis should be lowered $0.2493 - (-0.2471) = 0.5203\,m$, which cannot be accomplished varying stiffness and damping coefficients without impairing the general dynamics of the vehicle. In this sense, a redesign of the frontal double wishbone and

| Algorithm | $(\bar{r}_z^G)_{ini}$ | $(\bar{r}_z^G)_{opt}$ | $\Psi$ | Nº iter |
|---|---|---|---|---|
| Interior Point | -0.2471 | 0.2493 | 0.4227 | 7 |
| TRR | -0.2471 | 0.2493 | 0.4227 | 8 |
| SQP | -0.2471 | 0.2493 | 0.4227 | 7 |
| Active set | -0.2471 | 0.2493 | 0.4227 | 6 |

Table 7.23: Optimization results for the double-lane-change maneuver with the Z local coordinate of the center of mass of the chassis as parameter.

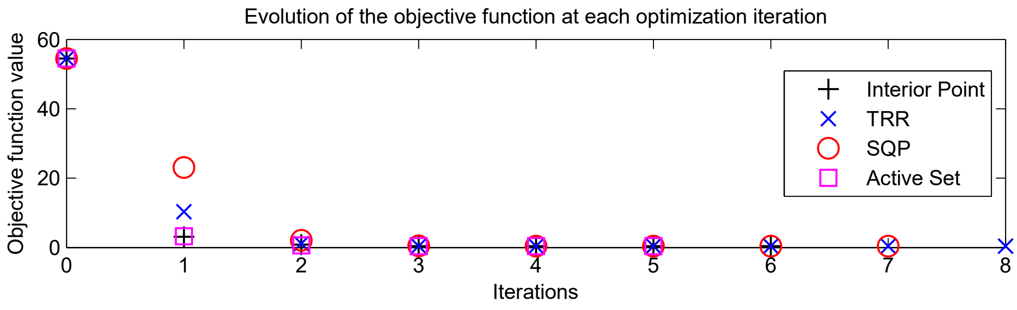


Figure 7.35: Evolution of objective function value per optimization iteration with the Z-component of the center of mass as parameter.

the rear McPherson suspensions should be addressed to obtain the minimum possible roll rate, which cannot be accomplished with the parameters selected.

The evolution of the roll rate for the original and optimized local Z coordinate of the chassis CoM is exposed in Figure 7.36. This figure makes clear that the optimized parameter is able to reduce the absolute value of the roll rate in every one of the curves executed by the buggy vehicle during the double lane change maneuver, even though its value is out of the feasible solutions.

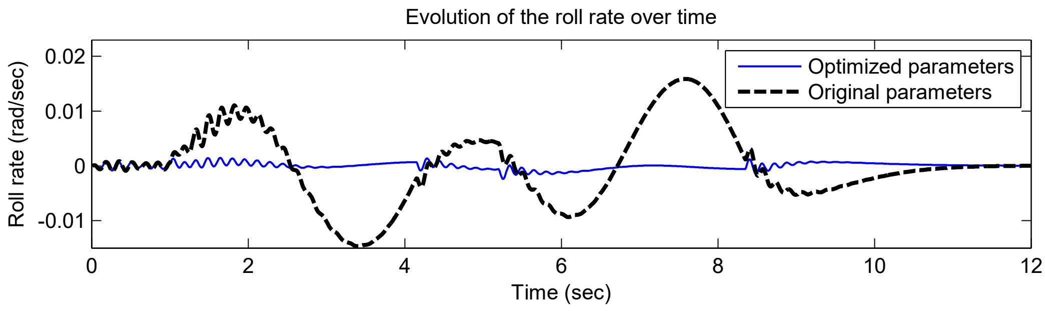


Figure 7.36: Evolution of the roll rate over time on a double lane change maneuver for the original and the optimized set of parameters.

In brief, an application case of the sensitivity analysis developments presented in previous chapters has been studied, and results have been discussed in terms of optimization algorithms and with respect to the final dynamic performance.

## 7.4 Bicycle

The dynamic analysis of bicycles is a subject that has attracted the attention of several researchers for decades [141]. This particular multibody system, composed of 4 bodies in its simpler configuration, has leaded to interesting studies in the field of system identification [142] and virtual sensors [143], among many others.

### 7.4.1 Multibody model

The model of bicycle used in the present work is based on the description presented in [144], but the contact between tire and ground, modeled in [144] with non-slipping rolling contact by means of holonomic and non-holonomic constraints, is substituted in this work by normal and frictional tire forces.

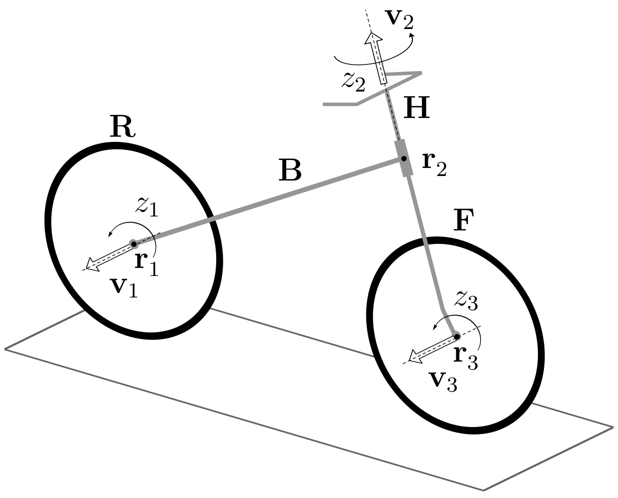


Figure 7.37: Model of bicycle.

The bicycle displayed in Figure 7.37 is composed of four bodies, i.e. the frame (B), the fork-handlebar (H), the front wheel (F) and rear wheel (R), which are linked by means of revolute joints. The body of the rider is considered rigidly attached to the frame, so its effects in terms of the magnitude and disposition of masses are incorporated to the frame. All the geometries, masses and inertia values of the model described in [144] match the ones used in the present model.

The normal contact model used for the interaction between each one of the tires and the ground is based on a Kelvin-Voigt contact model with hysteresis (see [137]). Contact detection and the indentation assessment is executed by means of an analytical circumference-plane model, already used in the vehicle model described in section 7.3. The frictional part of tire forces is evaluated through the linearized tire model described in [138]. The coefficients used to model the contact forces are displayed in Table 7.24, in which the nomenclature used in [137] for normal contact and in [138] for frictional forces has been reused.

| Force coefficient | Front wheel | Rear wheel |
|---|---|---|
| $K$ | 6000 | 6000 |
| $D_c$ | 100 | 100 |
| $D_R$ | 80 | 80 |
| $Radius$ | 0.35 | 0.30 |
| $\kappa_c$ | 0.15 | 0.15 |
| $\alpha_c$ | 0.15 | 0.15 |
| $\mu_x$ | 1.0 | 1.0 |
| $\mu_y$ | 1.0 | 1.0 |

Table 7.24: Coefficients of tire forces in the bicycle model.

The rear wheel and the steering are actuated by means of two external torques in order to control the velocity of the bicycle and its leaning angle. These external torques are modeled through non-cyclical splines parameterized by means of a series of values of the torques at different time steps, guaranteeing their continuity and differentiability. For the dynamic and sensitivity analysis, the splines displayed in Figure 7.38 parameterized with 16 points each will be regarded.

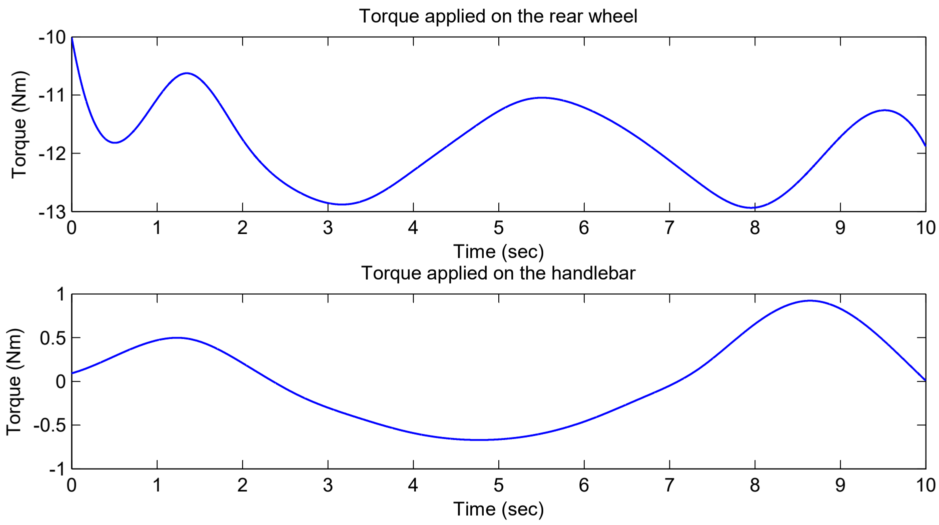


Figure 7.38: Torque applied on the rear wheel and handlebar modeled through spline.

The model is described by means of 3 points, 8 vectors and 3 angles, identified in Figure 7.37 as $z_1$, $z_2$ and $z_3$. Accordingly, the natural coordinates model generated is defined by means of 36 variables subjected to 30 constraint equations.

The joint coordinate model automatically generated by MBSLIM is composed of 3 revolute joints and a floating joint, since there are not kinematic relations between any of the bodies of the bicycle and and the ground. The angles added in the natural coordinate model are automatically assimilated by MBSLIM to the joint coordinates of the three revolute joints, and since there is no additional user constraint or closed loop in the system, there is only one constraint in this model related to the normalization of the Euler parameters of the floating joint.

Regarding the sensitivity analysis of the system, the objective function considered is a combination of an error in the linear velocity of the system and an error in the leaning angle, both with respect to the references presented in Figure 7.39 :

$$\psi = \int_{t_0}^{t_F} \left( \mid \phi - \phi_{ref} \mid + \mid \parallel \dot{\mathbf{r}}_1 \parallel - \dot{r}_{1ref} \mid \right) \mathrm{d}t \tag{7.50}$$

where $\phi$ is the leaning angle (using the notation presented in [144]), and $\dot{\mathbf{r}}_1$ is the velocity of the center of the rear wheel.

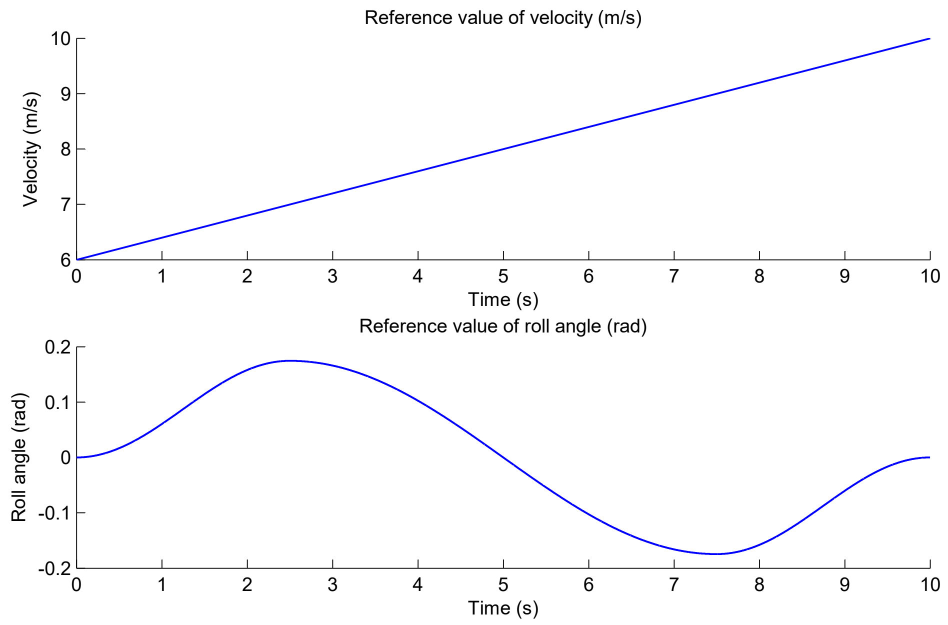


Figure 7.39: Reference velocity and roll angle for the dynamic simulation of the bicycle.

The group of instant torques used to compute the splines is regarded as the set of parameters of the sensitivity analysis. The change of the number of parameters, i.e. the modification of the number of instant torques used to calculate each spline, constitutes one of the most attractive features of this model, offering the possibility of comparing the performance of the different sensitivity formulations for different number of parameters.

The particular selection of parameters in this model allows to readily define an optimal control problem in terms of the set of parameters used to compute the splines and the objective function defined in (7.50). Even though the definition of the minimization problem is immediate, the process to reach the optimal solution implies that the dynamic simulation should be robust enough to support configurations that lead to the fall of the bicycle with impact forces and other “strange” configurations that could entail convergence problems. For that reason, an additional sphere-plane contact force has been added on the frame with the sphere center at a frame point artificially located 3 meters above the ground when the bicycle is upright. The radius of the contact sphere is 1 meter, while the plane of contact is the surface of the ground.

In order to limit the range of torques that can be applied to the rear wheel and handlebar, some upper and lower boundaries have been established, being $-10$ the lower limit and 200 the upper limit for each parameter.

## 7.4.2 Numerical results: dynamics

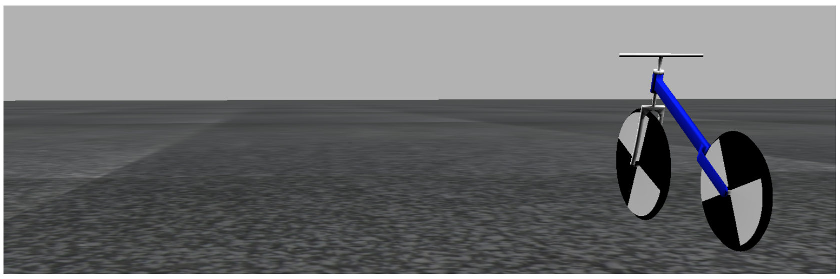


Figure 7.40: Graphical interface for the simulation of the bicycle.

The dynamic simulation consists in a 10 seconds maneuver subjected to gravity, to a traction torque on the rear wheel and to a steering torque applied on the fork-handlebar. A capture of the graphic interface configured for the dynamic simulation of the bicycle can be seen in Figure 7.40.

This model constitutes a clear example of an open-loop system, which in this case is composed of 3 revolute joints (frame-fork, front wheel-fork and rear wheel-frame) and a floating joint. However, the particular modeling of the floating joint through Euler parameters implies that no unconstrained formulation can be applied. In this regard, the normalization constraint of the Euler parameters obliges to enforce it through semi-recursive Matrix R or ALI3-P schemes, for instance.

Nevertheless, in order to test the fully-recursive formulation presented in section 2.4.2, the peculiarity of the Euler parameters normalization constraint in floating joints should be recalled. In brief, since only three of the four parameters are independent, three of them can be selected as main variables for the solution of each fully-recursive acceleration. Other possibility is to impose the Euler parameters normalization constraint at acceleration level when equation (2.247e) is solved for the floating joint. This leads to an index-1 classical Lagrange method for constraints applied on the base body, equivalent to the formulation described in [36]. This method has not been generally described in the thesis because it has only been implemented in the MBSLIM multibody library for constraints applied to the base body, thus it is not a general algorithm yet. Moreover, the formulation matches the index-1 fully-recursive formulation described in [36] for constraints applied on the base body, thus the reader is referred to this work for further detail.

In Figure 7.41, the seven methods tested display an excellent level of convergence, even for index-1 fully-recursive approaches.

| Formulation | CPU time | CPU time natural | Ratio *nat/rel* |
|---|---|---|---|
| RTdyn0 ALI3-P | 0.781 | 3.047 | 3.901 |
| RTdyn1 ALI3-P | 0.781 | 3.047 | 3.901 |
| RTdyn0 MatrixR | 1.031 | 3.063 | 2.971 |
| RTdyn1 MatrixR | 1.031 | 3.063 | 2.971 |
| RTdyn0 Index-1 FR | 0.484 | - | - |
| RTdyn1 Index-1 FR | 0.375 | - | - |

Table 7.25: CPU time of relative coordinates sensitivity formulations compared with the equivalent formulation in natural coordinates.

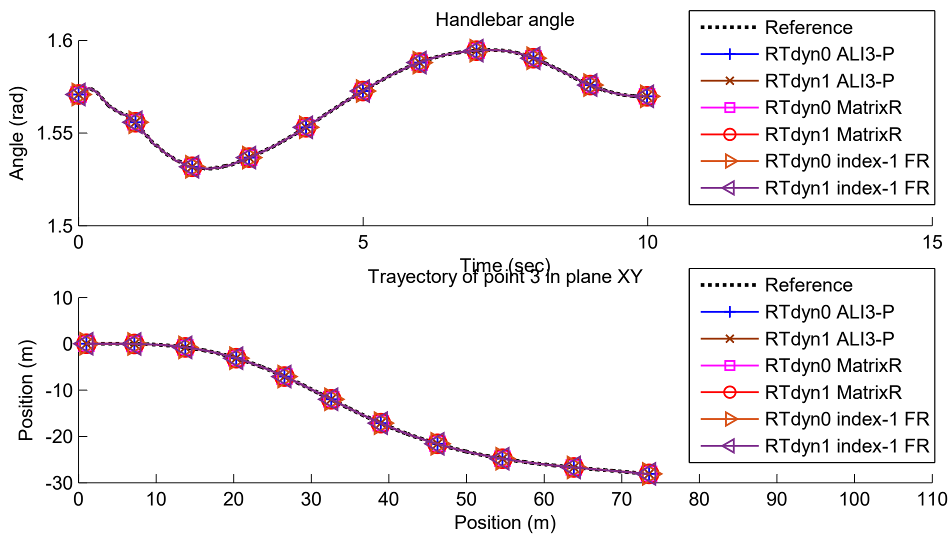


Figure 7.41: Reference velocity and roll angle for the dynamic simulation of the bicycle.

The four semi-recursive formulations along with the two fully-recursive ones are examined in terms of computational time in Table 7.25, displaying a much better performance than natural coordinate formulations. In this case, semi-recursive ALI3-P formulations are 3.901 times faster than the global ALI3-P method, while the semi-recursive Matrix R formulation, despite being slower than semi-recursive ALI3-P, are still almost 3 times more efficient than the equivalent global formulation. The worse behavior of natural coordinates comes from the traditional problem of Cartesian coordinate formulations with high speed rotating vectors, which involves an increase in the number of iterations, with the consequent impact on the computational expense.

Moreover, Table 7.25 proves that fully-recursive dynamic simulations are the fastest for this particular experiment. It should be commented likewise that the fully-recursive RTdyn1 approach is much more efficient than RTdyn0. This divergence can be attributed to the direct MBSLIM implementation of the method without exploiting the similarities between these two accumulation schemes, unlike in semi-recursive methods.

### 7.4.3 Numerical results: sensitivity analysis

In this section, the sensitivity analysis is accomplished, using the semi-recursive ALI3-P and Matrix R direct and adjoint sensitivity formulations presented in chapter 5. Moreover, the results of the application of the direct differentiation method to the fully-recursive formulation with base-body constraints imposed by meas of an index-1 scheme is evaluated. All the sensitivity results are compared with the reference response obtained with the natural coordinates Matrix R direct and adjoint sensitivity formulations for the same time step.

Let us begin with the fully-recursive forward sensitivity formulation, evaluated for a shorter simulation time (5 seconds). Figure 7.42 evidences that the fully-recursive gradient starts to diverge from the reference at 4.5 seconds approximately, being the results obtained after this time completely inaccurate. In order to understand better this behavior, let us inspect the fulfillment of the sensitivities of the constraints vector at position, velocity and acceleration levels in Figure 7.44. Let us remind here that the biggest drawback of the equations of motion of a dynamic system modeled with an index-1 DAE system is the drift-off error that could lead to divergences in the dynamics. To the best of the author's knowledge, the effect of drift-off in the sensitivities has not been as thoroughly studied as for dynamic analyses, and it is included here for that reason.

Since no additional enforcement of constraints at position or velocity levels has been carried out during the dynamics, the constraints vector at position and velocity levels acquire increasing errors during the simulation, even though not so high to affect the dynamic results, as it is shown in Figure 7.43. On the contrary, sensitivity analysis includes this effect and amplifies it in such a manner that this drift-off error entails that the sensitivity analysis diverges.

This problem constitutes a perfect example of the index-1 formulation problems extended to sensitivities. Partially, this formulation has not been extended and generalized in MBSLIM to support any type of constraints both for dynamics and sensitivity analysis due to this drift-off problems. In fact, other constrained fully-recursive formulations have been explored, but robust and stable schemes require the addition of coupling terms that lead to very complex formulations and/or to couplings that oblige to resort to a semi-recursive solution. It should also be remarked that this problem stems from the imposition of the constraint equations. The open-loop fully-recursive sensitivity expressions have been proved and verified in other numerical examples, delivering good and stable behaviors even for large simulation times.

On the other hand, semi-recursive sensitivity formulations display an excellent

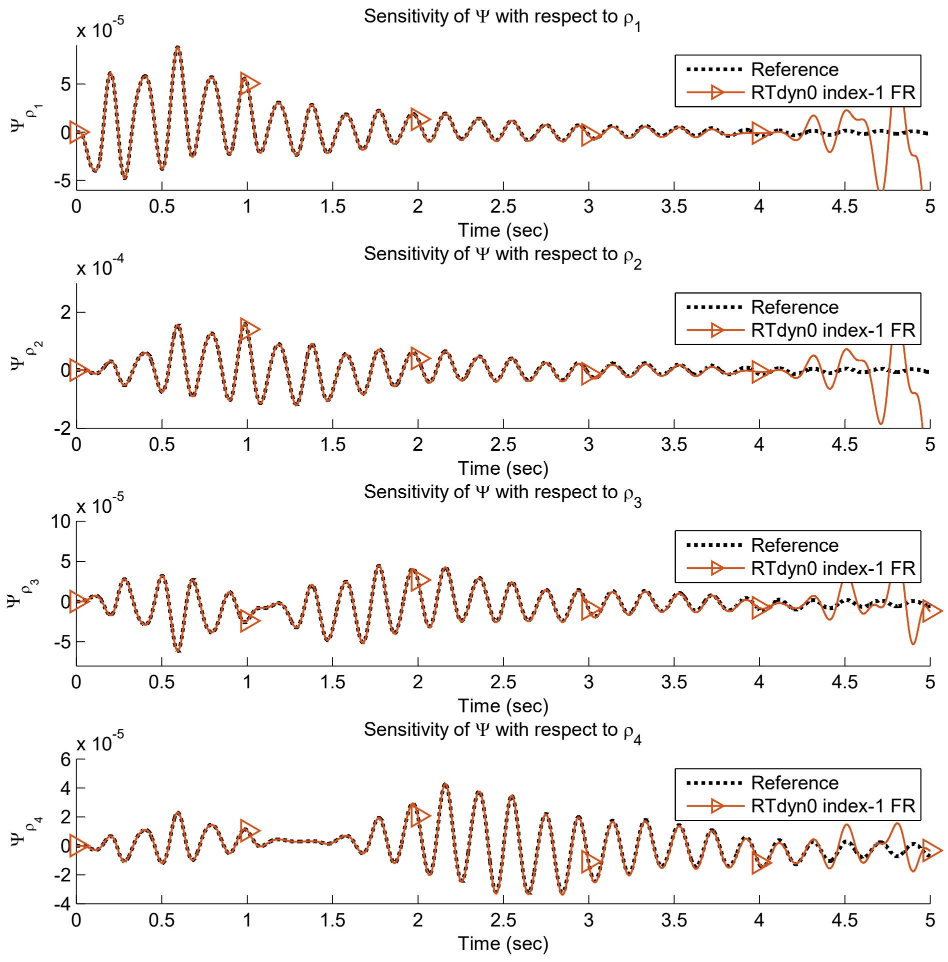


Figure 7.42: Sensitivity of fully-recursive formulation with an index-1 enforcement of constraints for a RTdyn0 accumulation.

performance in terms of accuracy as shown in Figure 7.46, which exhibits the gradient of the objective function displayed on Figure 7.45 with respect to the first four sensitivity parameters (as sample) for the complete 10 seconds simulation.

Adjoint variable methods show as well great convergence levels, as it can be observed in Figure 7.47. Once again, it should be remarked that here two different models are compared, based on relative and natural coordinates, two different formulations (Matrix R and ALI3-P), two accumulation schemes (RTdyn0 and RTdyn1), two differentiation schemes (forward and adjoint sensitivity formulations) and two discretization schemes (discrete and adjoint variable methods), and equivalent results are reached with a high level of convergence.

In this numerical example, semi-recursive methods have proved a more efficient dynamic performance than the equivalent formulation in natural coordinates, as shown in Table 7.25. The sensitivity analysis computational times, gathered in Table 7.26,

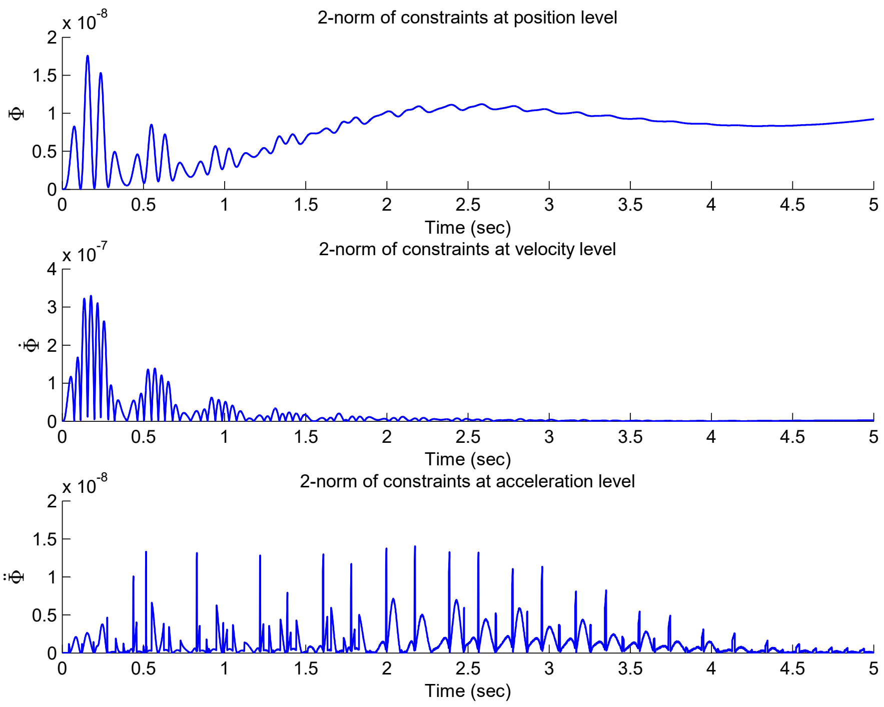


Figure 7.43: Fulfillment of constraints at position, velocity and acceleration levels.

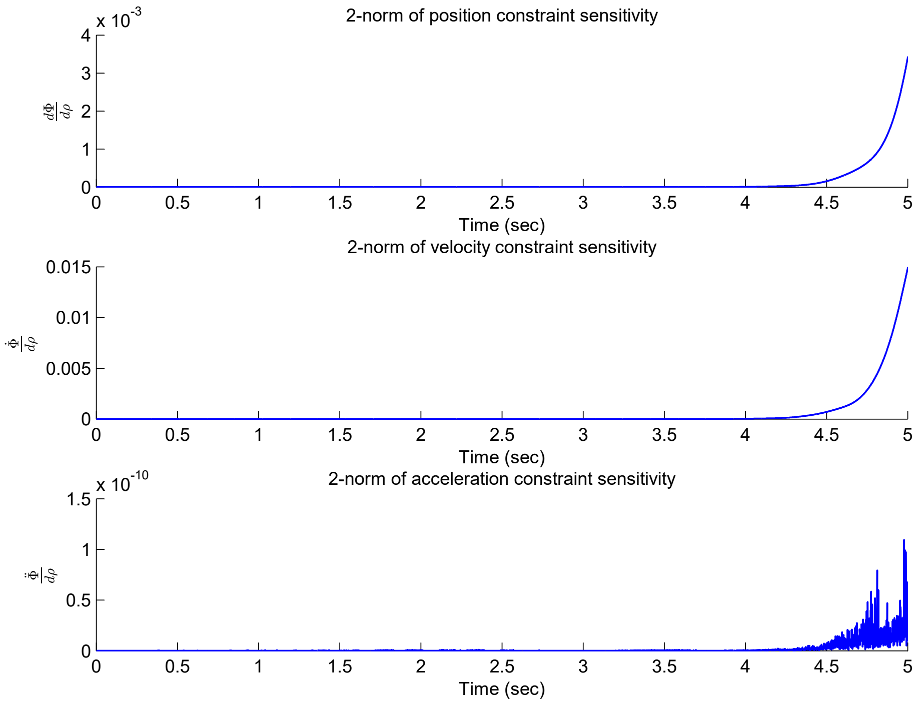


Figure 7.44: Constraints sensitivities at position, velocity and acceleration levels.

also exhibit that less computational effort is required by joint coordinate models than

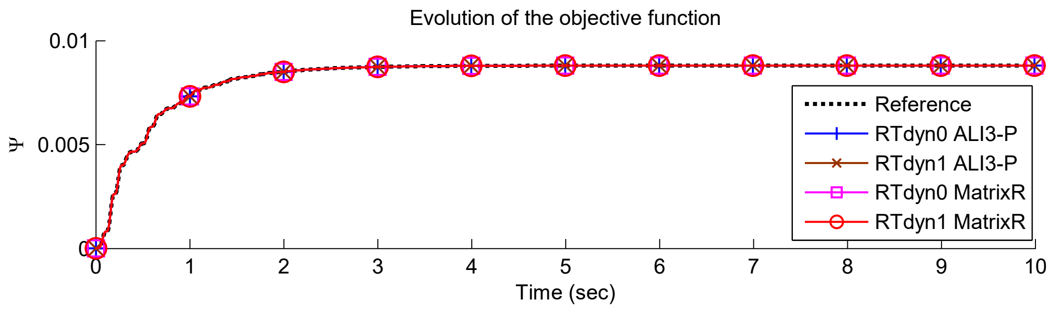


Figure 7.45: Evolution of the objective function over time.

| Formulation | CPU time | CPU time naturals | Ratio *nat/rel* |
|---|---|---|---|
| RTdyn0 ALI3-P: DDM | 1.967 | 10.063 | 5.116 |
| RTdyn0 ALI3-P: CAVM | 1.750 | 5.578 | 3.187 |
| RTdyn0 ALI3-P: DAVM | 1.718 | 5.469 | 3.183 |
| RTdyn1 ALI3-P: DDM | 2.078 | 10.063 | 4.843 |
| RTdyn1 ALI3-P: CAVM | 1.891 | 5.578 | 2.950 |
| RTdyn1 ALI3-P: DAVM | 1.844 | 5.469 | 2.966 |
| RTdyn0 MatrixR: DDM | 6.219 | 8.484 | 1.364 |
| RTdyn0 MatrixR: CAVM | 6.141 | 9.203 | 1.499 |
| RTdyn1 MatrixR: DDM | 6.313 | 8.484 | 1.344 |
| RTdyn1 MatrixR: CAVM | 6.219 | 9.203 | 1.480 |

Table 7.26: CPU time of semi-recursive sensitivity formulations compared with the equivalent one in natural coordinates.

by the corresponding sensitivity formulation in natural coordinates. Moreover, the great gains in semi-recursive ALI3-P formulations compared to the equivalent global method should be further explained . Since the sensitivity problem has 32 parameters, 32 systems of equations have to be solved in forward sensitivity formulations, and since the bigger effort in global ALI3-P formulations is devoted to the resolution of these systems of equations, it is the most impaired method for an increase in the number of parameters. In fact, global ALI3-P adjoint methods display much better computational efficiency.

Besides, RTdyn0 and RTdyn1 show small differences in CPU time according to Table 7.26, being RTdyn0 the fastest. The better performance of RTdyn0 comes from the particular definition of the floating joint that highly simplifies the recursive kinematic relations that appear in the accumulation of masses and forces. An additional reason relies on the reduced number of bodies and joints, which makes that time savings related to the floating joint determine the computational performance of the dynamic and sensitivity formulations.

Other interesting study that can be easily addressed with this multibody system consists in the evaluation of the effect of an increase of the number of parameters in the efficiency of each sensitivity formulation. For the sake of clarity, two constant

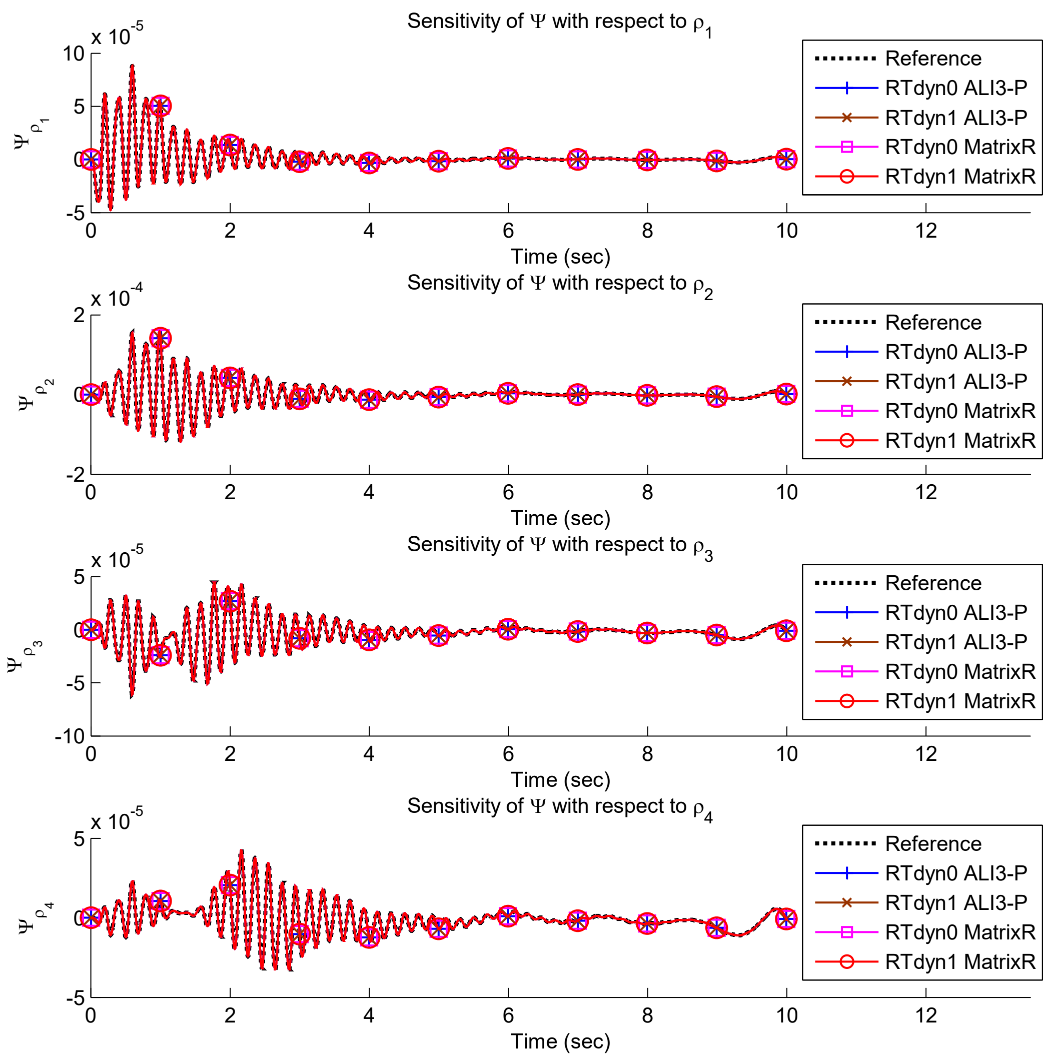


Figure 7.46: Objective function gradient obtained by means of semi-recursive forward sensitivity formulations.

splines will be used in order to guarantee that the dynamic maneuver is preserved between simulations with different number of parameters. For comparison purposes, a null torque will be applied on the handlebar and a constant momentum of $-12Nm$ on the rear wheel.

It is evident from Table 7.27 that adjoint methods are extremely well suited for a high number of parameters, being their efficiency much less affected by the increase in the parameters size than direct differentiation methods. However, semi-recursive Matrix R methods display a reduced decreasing in its computational expense. In those formulations, the composition of the equations of motion is more computationally

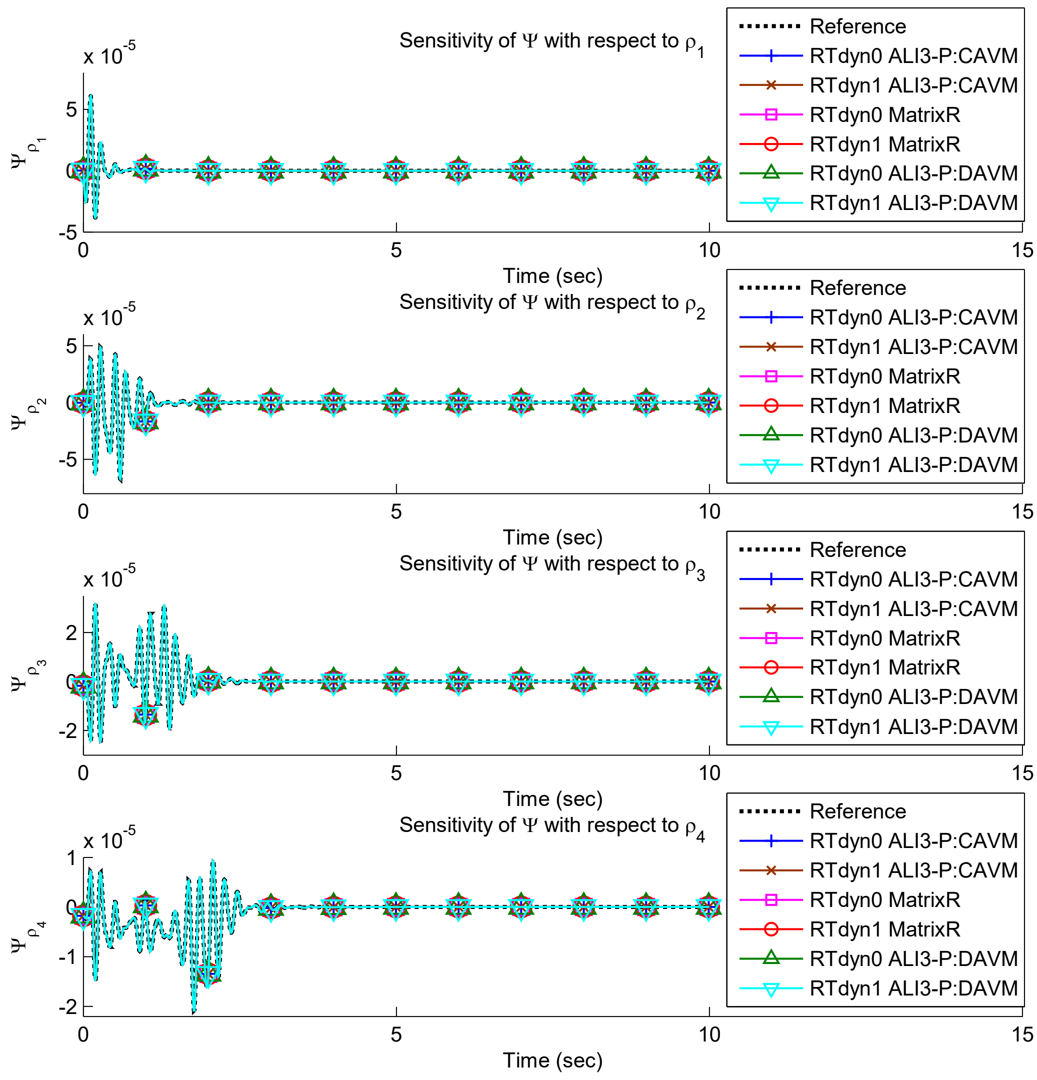


Figure 7.47: Objective function gradient obtained by means of semi-recursive adjoint sensitivity formulations.

demanding than the solution of the system. Since adjoint methods only reduce the number of systems to be solved, but require the same (or more) terms than direct differentiation methods, the computational expense remains almost unaltered. In the case of semi-recursive ALI3-P formulations, the solution of the equations of motion is more cumbersome, thus, the computational effort of adjoint methods is significantly lower than the one demanded by direct differentiation methods.

| | Number of parameters | | | | |
|---|---|---|---|---|---|
| Formulation | 32 | 100 | 200 | 500 | 1000 |
| RTdyn0 ALI3-P: DDM | 1.906 | 2.406 | 3.219 | 5.359 | 9.203 |
| RTdyn0 ALI3-P: CAVM | 1.734 | 1.938 | 2.047 | 2.328 | 2.953 |
| RTdyn0 ALI3-P: DAVM | 1.703 | 1.766 | 1.906 | 2.266 | 2.922 |
| RTdyn1 ALI3-P: DDM | 2.063 | 2.578 | 3.375 | 5.813 | 10.375 |
| RTdyn1 ALI3-P: CAVM | 1.813 | 1.969 | 2.219 | 2.781 | 3.906 |
| RTdyn1 ALI3-P: DAVM | 1.781 | 1.984 | 2.189 | 2.813 | 3.875 |
| RTdyn0 MatrixR: DDM | 5.969 | 7.422 | 9.891 | 17.672 | 30.391 |
| RTdyn0 MatrixR: CAVM | 5.890 | 7.172 | 9.641 | 16.875 | 28.953 |
| RTdyn1 MatrixR: DDM | 6.063 | 7.563 | 10.172 | 18.125 | 31.359 |
| RTdyn1 MatrixR: CAVM | 6.016 | 7.375 | 9.781 | 17.438 | 29.500 |

Table 7.27: CPU time comparative among sensitivity formulations with different numbers of parameters.

### 7.4.4 Numerical results: optimal control

With the sensitivity analysis defined in the former section, it is possible to address an optimal control problem by means of a gradient-based optimization algorithm. In this case, the L-BFGS-B bound constrained optimization algorithm [132] will be considered.

The optimization problem can be described as a minimization problem with the form:

$$\min_{\boldsymbol{\rho}} \quad \boldsymbol{\psi} = \int_{t_0}^{t_F} (\mid \phi - \phi_{ref} \mid + \mid\parallel \dot{\mathbf{r}}_1 \parallel - \dot{r}_{1ref} \mid) \, \mathrm{d}t \tag{7.51}$$

$$\text{s.t.} \quad \rho_i \leq U \tag{7.52}$$

$$\rho_i \geq L \tag{7.53}$$

with

$$\boldsymbol{\rho} = \begin{bmatrix} \rho_1 & \dots & \rho_i & \dots & \rho_p \end{bmatrix}^{\mathrm{T}} \tag{7.54}$$

and with $U = 200$ and $L = -10$.

Since there is only one objective function and multiple parameters, the maneuver sought is complex and there are multiple deviations and multiple paths to follow, it is convenient to divide the optimization problem into smaller optimization problems with reduced simulation times. This partitioning consists in the division of the total 10 seconds simulation into smaller optimizations with increasing time steps, all of them starting from time $t_0 = 0s$. The simulation time is increased by 1 second between optimizations. It should be remarked that this partitioning solution has been reached after several attempts and reformulations of the optimal control problem, being this the simplest an most effective.

The values of the parameters used in the dynamics and sensitivity analysis sections are the results obtained from this optimal control problem. Thus, they have not been used as initial guess points to start the optimization problem. Instead, the initial rear wheel torque consists in a constant momentum of $-12Nm$, while the handlebar torque is initialized with a constant null momentum. With this initial configuration, the bicycle moves in a straight line and with a variable linear velocity.

The optimization problem is solved by means of the L-BFGS-B algorithm developed by Nocedal and Morales and described in [132]. In this case, this algorithm has been selected in order to prove other options than the $fmincon$ tool of Matlab and to test its performance. In fact, this problem has been also executed with Matlab in order to prove the validity of the results, but since the performance of the $fmincon$ optimization algorithms have been assessed in the vehicle numerical experiment of section 7.3, here the focus will be put upon the L-BFGS-B algorithm.

The performance of the modified version of the L-BFGS-B described in [132] is determined by a set of parameters which control the stop criteria and the definition of upper and lower bounds, among other features. For the optimization which leads to the results discussed in the dynamics and sensitivity analysis sections, the variable associated to the stop criteria ($factr$) has been set to 10, which is equivalent to "extremely high accuracy", according to the optimization algorithm. The stop criteria associated to the maximum absolute value of the projected gradient ($pgtol$) has been configured to $10^{-5}$. The upper and lower bounds are activated according to the limits commented above.

With the parameterization of the L-BFGS-B algorithm described above, an excellent optimum is expected, but at the cost of a high number of iterations. Remember that the optimization problem is addressed by means of a set of sequential optimization problems with increasing simulation times. As a consequence, the final number of iterations is extremely high, but since each analytical gradient evaluation requires only a few seconds according to to the results of section 7.4.3, the complete optimal control problem is addressed in a few minutes.

| $factr$ | $pgtol$ | $\Psi$ | Nº iter |
|---:|---|---:|---:|
| $10^1$ (high accuracy) | $10^{-5}$ | $8.8012 \cdot 10^{-3}$ | 3635 |
| $10^7$ (medium accuracy) | $10^{-5}$ | $8.8033 \cdot 10^{-3}$ | 2690 |
| $10^{12}$ (low accuracy) | $10^{-5}$ | $1.8586 \cdot 10^{-2}$ | 42 |

Table 7.28: Optimization results for different parameterizations of the L-BFGS-B algorithm.

In Table 7.28, the results obtained for other configurations of the L-BFGS-B algorithm are displayed. Optimization with medium and high accuracy display very close values related to the objective function value, while low accuracy optimization leads to poorer local optimums. In regard to the number of iterations, the computational cost grows with the accuracy level demanded. It should be remarked that the number of iterations displayed in Table 7.28 is the number of iterations of each intermediate optimization summed altogether.

Due to the high number of parameters involved in the optimization, the final

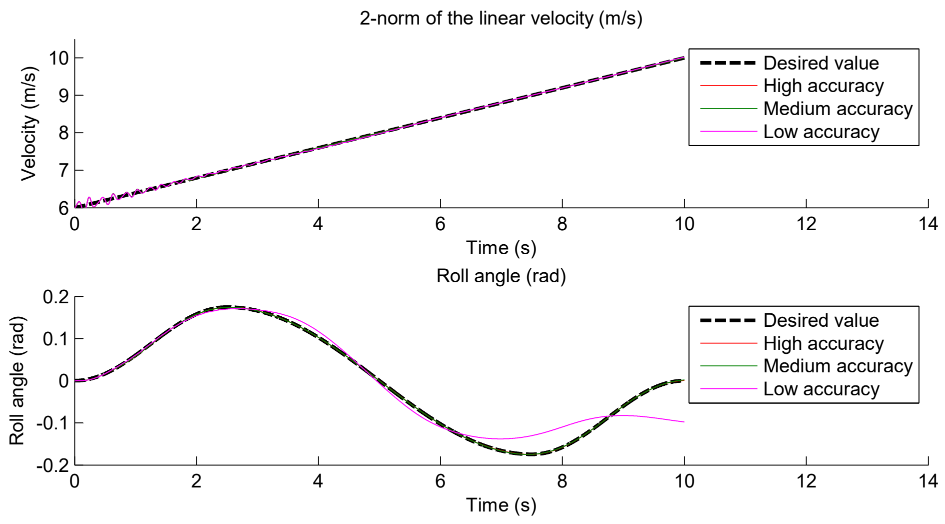


Figure 7.48: Dynamic response of the bicycle after the optimization with L-BFGS-B algorithm with high, low and medium accuracy compared with the desired velocity and roll angle.

dynamic responses with the three sets of optimized parameters are assessed rather than examining optimized arrays value by value. The linear velocity and the roll angle are compared in Figure 7.48 with respect to the desired response for the cases of high, medium and low accuracy. Behold that high and medium accuracy optimizations fall very close to the desired responses, while the low accuracy optimization is not able to reach such convergence levels.

## 7.5 Ship anchor maneuver

The fifth numerical experiment considered consists in the dynamic simulation of the lift anchor maneuver described in [145]. The main interest of this problem relies on the large concatenation of bodies composing a single kinematic chain. Long single-chain models lead to large fully-dense system matrices in semi-recursive accumulation schemes which are computationally expensive to assemble. In this regard, the best performance is expected from fully-recursive formulations.

It should be commented that the evaluation of contact-frictional forces is extremely computationally demanding in this problem and strongly conditions the efficiency of the simulation regardless of the formulation selected

### 7.5.1 Multibody model

The mechanism is composed of 64 bodies including an anchor in the end of the kinematic chain, linked to a special end link by means of a revolute joint and a series of 61 chain links of diverse type. The relative motion between chain links is described by Cardan joints with the exception of the swivel link shown in Figure 7.49 which includes an intermediate revolute joint in order to prevent the twisting of the chain.

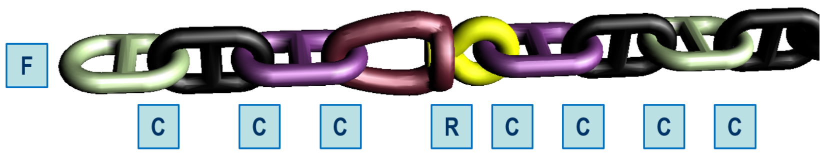


Figure 7.49: Detail of the swivel link in the anchor chain, with kinematic joints identified as “C” (Cardan), “R” (revolute) and “F” (floating).

The chain lifting is simulated by means of a rheonomous constraint applied to a point belonging to the extreme link of the chain placed in the opposite tip of the anchor. Moreover, the contact of each link and the anchor with the hull of the ship is evaluated by means of the contact-frictional force model described in [145]. The multibody model is thus subjected to contact-frictional forces, gravity and a rheonomous guidance constraint.

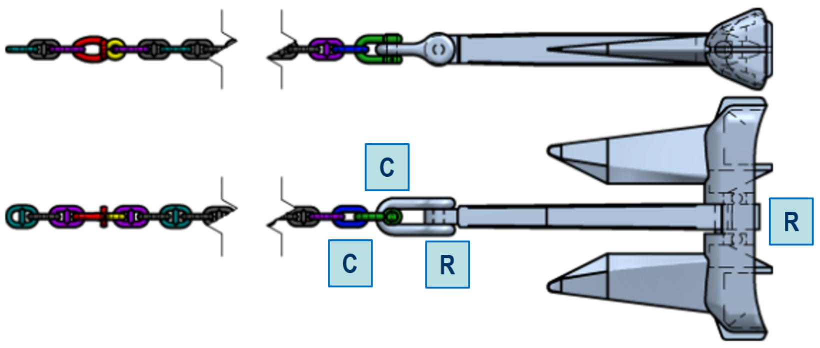


Figure 7.50: Detail of the anchor chain tip, with kinematic joints identified as “C” (Cardan), and “R” (revolute).

The system of coordinates selected to model a mechanism involving a large concatenation of bodies in a single chain strongly determines the efficiency of its kinematic or dynamic simulation. Natural coordinates generate a model with 702 dependent variables and subjected to 700 constraints. Despite the enormous size of the model, the system matrices generated are strongly sparse. Taking advantage of sparse algebra and sparse solvers, the dynamics of this mechanism can be solved efficiently in fully-Cartesian coordinates.

Relative coordinate models, on the contrary, generate a model described by a floating joint, 3 revolute joints and 60 Cardan joints, delivering a total of 130 joint coordinates. Since there are not closed loops, no loop-closure constraint has to be added, and only four constraints equations have to be considered in the base body:

three rheonomous driving constraints and a normalization constraint for the floating joint Euler parameters.

## 7.5.2 Numerical results: dynamics

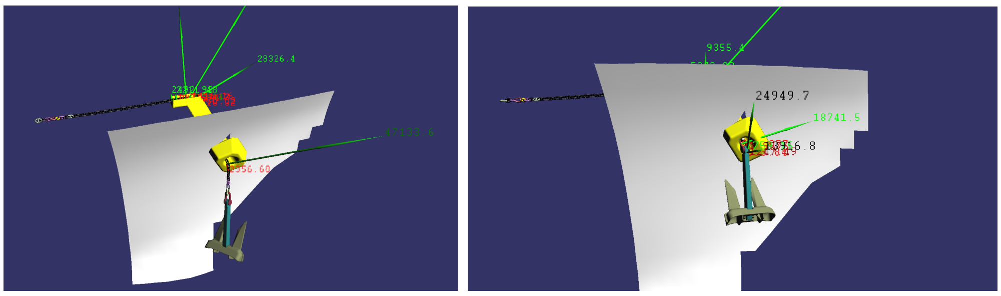


Figure 7.51: Simulation of the lifting anchor maenuver.

The forward dynamic simulation consists in a 20 second lifting anchor maneuver with a time step of 0.05 milliseconds (this reduced time step is required by the contact model considered). Two captures of the dynamic simulation are shown in Figure 7.51.

First of all, lets us evaluate the dynamic performance of joint coordinate models. The semi-recursive generation of the equations of motion yields a fully dense symmetric mass matrix whose composition could result as computationally expensive as the solution of natural coordinate models for a large enough number of bodies. On the contrary, fully-recursive algorithms do not involve the generation of the full system mass matrix, but elemental matrices that are accumulated body-by-body from the tips of the mechanism to the base. This mechanism is, thus, better suited to be solved with a fully-recursive algorithm, leading to the solution of small systems for each joint, instead of a large dense system of size $130 \times 130$.

Nevertheless, it should be reminded that fully-recursive algorithms are based on a fixed-point solution of the equations of motion, which is less robust than the Newton-Raphson schemes used in the semi-recursive ALI3-P or Matrix R formulations. Contact forces, in this case, could require to use smaller time steps in fully-recursive algorithms than in semi-recursive methods, thus impairing their efficiency. Nevertheless, for the sake of clarity, all the formulations will be executed with the same time step of 0.05 milliseconds.

For simplicity, only the semi-recursive and global ALI3-P formulations along with the fully-recursive one will be tested because we know that the efficiency of the Matrix R global and semi-recursive formulation will be much worse, specially in a system with a large number of degrees of freedom like the one considered in this section.

Table 7.29 evidences that, for a given fixed time step, fully-recursive formulations perform much more efficiently than any other formulation in relative or natural coordinates. Moreover, in this case, the global ALI3-P formulation in natural coordinates display a better behavior than the topological semi-recursive ones.

Behold that this case has been explicitly selected so as to display the drawbacks of semi-recursive formulations with large single chain concatenations. In this particular

| Formulation | CPU time | CPU time natural | Ratio *nat/rel* |
|---|---|---|---|
| RTdyn0 ALI3-P | 1505.82 | 1115.46 | 0.741 |
| RTdyn1 ALI3-P | 1467.34 | 1115.46 | 0.760 |
| RTdyn0 Index-1 FR | 356.72 | - | - |
| RTdyn1 Index-1 FR | 358.03 | - | - |

Table 7.29: CPU time of relative coordinates sensitivity formulations compared with the equivalent formulation in natural coordinates for the lift anchor maneuver.

mechanism, contact forces are more cumbersome to be assembled in relative coordinates than in natural coordinates (thus the stiffness and damping matrices) and the mass matrix turns out to be fully dense, thus impairing the computational efficiency. It should be also clarified that kinematic chains with fewer chain links perform much more efficiently in semi-recursive topological than in global formulations.

Although this numerical example disagrees with the idea that relative coordinate modeling is more efficient, it also evidences the advantages of having both global and topological formulations implemented in the general multibody library MBSLIM, making it possible to select the best set of coordinates and formulation for each mechanism. Moreover, it also highlights that there is no universally best set of coordinates and formulation, because their performance depends on the type of mechanism considered.

### 7.5.3 Numerical results: sensitivity analysis

This problem has been studied only at a dynamic level in order to prove the limitations of the dynamic formulations developed in this work. Moreover, the CPU-times for the sensitivity analysis are even more disadvantageous for the semi-recursive formulations than in the case of the forward dynamics.

In future works, it is intended to address different optimizations on the lifting anchor maneuver.

# Chapter 8

# Conclusions and future work

This chapter gathers the conclusions of the work developed during this thesis and outlines the future research topics opened by this work and which are still unfinished or have not been addressed yet.

## 8.1 Conclusions

The most important conclusions about the methods developed in this thesis are the following:

- A more general and systematic description of the already existing topological semi-recursive methods is provided, extending them for an arbitrary reference point and connecting them with the classical kinematics and dynamics equations of the relative motion. From these general equations, two particular cases have been implemented, corresponding to two different choices of reference points: RTdyn0 (CoM of each body as reference point), and RTdyn1 (point of each body coincident with the global origin as reference point).

- Fully-recursive formulations for open-loop systems have also been covered for an arbitrary selection of reference points.

- The kinematic problems of constrained multibody systems modeled in joint coordinates have been extended to support degrees of freedom not included in the dependent joint coordinate vector.

- Semi-recursive methods have been successfully combined with two constraint enforcement techniques, delivering the independent coordinate semi-recursive Matrix R formulation and the dependent coordinate semi-recursive ALI3-P formulation.

- The sensitivity analyses of unconstrained open-loop formulations have been addressed for the semi-recursive and the fully-recursive methods.

- All the derivatives of recursive kinematic relations, accumulations and assembly procedures have been attained by means of analytical differentiation. Moreover, derivatives have been carefully inspected and tested in order to guarantee maximum accuracy and efficiency.
- The sensitivity of constrained kinematic problems has been accomplished for joint coordinate models, delivering a kinematic forward sensitivity formulation and a kinematic adjoint sensitivity formulation.
- The sensitivity analysis of semi-recursive Matrix R formulations has been successfully developed, yielding the forward and adjoint semi-recursive Matrix R sensitivity formulations.
- The semi-recursive ALI3-P dynamic formulations have also been studied in terms of sensitivity using the direct differentiation method and the discrete and continuous adjoint variable method. As a result, three sensitivity formulations with two different choices of reference points each have been developed and implemented.

Moreover, it should be remarked that all the kinematic, dynamic and sensitivity formulations included in this thesis document have been implemented in the general purpose multibody library MBSLIM as general formulations, this is, they are intended for any definition of a multibody model. This implementation has the following features:

- The definition of a model is unique in MBSLIM, thus joint coordinate models have been programmed to be automatically generated from the set of points, vectors, angles, distances and constraints specified by the user.
- The opening of closed loops in joint coordinate models has been coded to be executed automatically too.
- Natural coordinate and joint coordinate models now coexist in MBSLIM, which makes possible to address any kinematic, dynamic or sensitivity problem in any coordinates by simply selecting the appropriate formulation.
- Joint coordinate models have been programmed to be solved with dense solvers, although the sparsity of some terms and derivatives have been harnessed by reusing the sparse algebra functions already included in MBSLIM for natural coordinate models.
- All the methods implemented support redundant constraints like the previously existing natural coordinate formulations.
- All the derivatives involved in the abovementioned joint coordinate sensitivity formulations have been analytically developed and programmed except, the partial derivatives of some forces with respect to parameters, such as spline forces, which are computed by means of automatic differentiation supported by the

Eigen library. This proves that the sensitivity methods programmed can be easily extended for complex derivatives of particular forces without the need of generating analytical derivatives in every case.

The MBSLIM implementation has been tested with five numerical examples: a five-bar mechanism, a spatial slider crank, a buggy vehicle, a bicycle and a chain-anchor system. From the solution of the dynamic, sensitivity analysis, design optimization and optimal control problems of those multibody models, the following conclusions can be inferred:

- The dynamic and sensitivity formulations developed in this work have been tested in complex numerical experiments involving normal contact, friction, tire forces or rheonomic constraints.

- All the dynamic and sensitivity analyses tested provide highly accurate results despite the differences in multibody models (natural coordinates and joint-coordinates), dynamic formulations (Matrix R and ALI3-P) or sensitivity formulations (forward, continuous adjoint or discrete adjoint).

- Topological semi-recursive methods are usually but not always faster than global methods, as it has been shown in the ship anchor maneuver. This highlights the importance of supporting different approaches and coordinate types in a multibody library such as MBSLIM.

- Computational gains observed for topological semi-recursive methods with respect to global methods in dynamic formulations are drastically reduced in sensitivity formulations due to the higher complexity of semi-recursive derivatives. Nevertheless, semi-recursive sensitivity formulations clearly outperform the efficiency of global methods in many problems, as proved in the sensitivity analysis of the buggy vehicle and the bicycle.

- Topological fully-recursive methods are almost always the best option in terms of computational efficiency when open-loop systems are simulated. However, these methods gather a series of issues when they are combined with constraint equations which make them less attractive for general closed-loop systems, as it has been shown in the bicycle experiment.

- The design optimization and optimal control problems have made apparent that the sensitivity formulations developed in this work are appropriate to be used in the optimization of multibody systems. The solution of those problems has been accomplished by means of third party gradient-based optimization algorithms, such as L-BFGS-B or the optimization algorithms included in the $fmincon$ Matlab function.

## 8.2 Future work

During the study of joint coordinate modeling, recursive methods, constrained formulations and sensitivity analyses, some lines of investigation have emerged but they have not been addressed yet. Some of the most interesting topics are listed below:

- **Fully-recursive formulations and sensitivities for closed-loop systems**. This line has been explored, but no significant advance has been reached resulting in worse formulations than the semi-recursive ones. The future line of investigation is related to robust and accurate fully-recursive formulations and their sensitivity analyses.

- **Deeper study of gradient-based optimization methods**. In this document, most of the effort has been devoted to the derivation of the sensitivity analysis of dynamics formulations based on analytical expressions. The research on gradient-based methods and its combination with the equations of motion in design optimization and optimal control have not been addressed in depth in this work.

- **Increase efficiency of recursive formulations**. There is still a gain margin for recursive formulations due to implementation inefficiencies, such as repeated or unnecessary calculations.

- **Parallelization**. In sensitivity analysis, it could be very profitable to parallelize some evaluations in order to reduce the computational effort.

- **Flexible multibody formulations**. The developments introduced in the present thesis document related to rigid bodies can be extended to flexible multibody systems, both in terms of dynamics and sensitivities. This line represents the most important of the commented future lines of research, and it is intended to require the author efforts during the next few months.

# Bibliography

[1] D. Dopico, Y. Zhu, A. Sandu, and C. Sandu, "Direct and adjoint sensitivity analysis of multibody systems using Maggi's equations," in *Proceedings of the ASME 2013 International Design Engineering Technical Conferences &Computers and Information in Engineering Conference. IDETC/CIE 2013*, 8 2013.

[2] D. Dopico, A. Sandu, C. Sandu, and Y. Zhu, "Sensitivity analysis of multibody dynamic systems modeled by ODEs and DAEs," in *Multibody Dynamics. Computational Methods and Applications* (Z. Terze, ed.), vol. 35 of *Computational Methods in Applied Sciences*, ch. Sensitivity Analysis of Multibody Dynamic Systems Modeled by ODEs and DAEs, pp. 1–32, Springer, 2014.

[3] D. Dopico, F. González, A. Luaces, M. Saura, and D. García-Vallejo, "Direct sensitivity analysis of multibody systems with holonomic and nonholonomic constraints via an index-3 augmented Lagrangian formulation with projections," *Nonlinear Dynamics*, May 2018.

[4] D. Dopico, A. Sandu, and C. Sandu, "Adjoint sensitivity index-3 augmented Lagrangian formulation with projections," *Mechanics Based Design of Structures and Machines*, 2021.

[5] D. Dopico, A. Luaces, U. Lugrís, M. Saura, F. González, E. Sanjurjo, and R. Pastorino, "MBSLIM: Multibody Systems in Laboratorio de Ingeniería Mecánica." `http://lim.ii.udc.es/MBSLIM`, 2009-2016.

[6] N. Orlandea, M. Chace, and D. Calahan, "A sparsity-oriented approach to the dynamic analysis and design of mechanical systems—part 1," *Journal of Manufacturing Science and Engineering, Transactions of the ASME*, vol. 99, no. 3, p. 773 – 779, 1977.

[7] N. Orlandea and D. A. Calahan, "A sparsity-oriented approach to the dynamic analysis and design of mechanical systems—part 2," *Journal of Manufacturing Science and Engineering, Transactions of the ASME*, vol. 99, no. 3, p. 780 – 784, 1977.

[8] J. Wittenburg, *Dynamics of systems of rigid bodies.* Wiesbaden: Vieweg+Teubner Verlag, 1977.

[9] O. Brüls and A. Cardona, “On the use of Lie group time integrators in multibody dynamics,” *Journal of Computational and Nonlinear Dynamics*, vol. 5, no. 3, p. 1 – 13, 2010.

[10] J. Garcia de Jalon, J. Unda, and A. Avello, “Natural coordinates for the computer analysis of multibody systems,” *Computer Methods in Applied Mechanics and Engineering*, vol. 56, no. 3, p. 309 – 327, 1986.

[11] J. García de Jalón, M. Serna, and R. Avilés, “Computer method for kinematic analysis of lower-pair mechanisms-II position problems,” *Mechanism and Machine Theory*, vol. 16, no. 5, p. 557 – 566, 1981.

[12] J. Garcia de Jalon, M. Serna, and R. Avilés, “Computer method for kinematic analysis of lower-pair mechanisms-I velocities and accelerations,” *Mechanism and Machine Theory*, vol. 16, no. 5, p. 543 – 556, 1981.

[13] M. A. Serna, R. Avilés, and J. Garcia de Jalon, “Dynamic analysis of plane mechanisms with lower pairs in basic coordinates,” *Mechanism and Machine Theory*, vol. 17, no. 6, p. 397 – 403, 1982.

[14] J. Garcia de Jalon, “Twenty-five years of natural coordinates,” *Multibody System Dynamics*, vol. 18, no. 1, p. 15 – 33, 2007.

[15] E. Bayo, J. Garcia de Jalon, A. Avello, and J. Cuadrado, “An efficient computational method for real time multibody dynamic simulation in fully cartesian coordinates,” *Computer Methods in Applied Mechanics and Engineering*, vol. 92, no. 3, p. 377 – 395, 1991.

[16] J. García de Jalón, J. M. Jiménez, A. Avello, F. Martín, and J. Cuadrado, “Real time simulation of complex 3-D multibody systems with realistic graphics,” in *Real-Time Integration Methods for Mechanical System Simulation* (E. J. Haug and R. C. Deyo, eds.), (Berlin, Heidelberg), pp. 265–292, Springer Berlin Heidelberg, 1991.

[17] F. González, D. Dopico, R. Pastorino, and J. Cuadrado, “Behaviour of augmented Lagrangian and Hamiltonian methods for multibody dynamics in the proximity of singular configurations,” *Nonlinear Dynamics*, vol. 85, no. 3, p. 1491 – 1508, 2016.

[18] M. Vermaut, F. Naets, and W. Desmet, “A flexible natural coordinates formulation (FNCF) for the efficient simulation of small-deformation multibody systems,” *International Journal for Numerical Methods in Engineering*, vol. 115, no. 11, p. 1353 – 1370, 2018.

[19] J. Cuadrado, R. Gutiérrez, M. Naya, and P. Morer, “A comparison in terms of accuracy and efficiency between a MBS dynamic formulation with stress analysis and a non-linear FEA code,” *International Journal for Numerical Methods in Engineering*, vol. 51, pp. 1033–1052, 07 2001.

[20] R. Featherstone, “The calculation of robot dynamics using articulated-body inertias,” *The International Journal of Robotics Research*, vol. 2, no. 1, p. 13 – 30, 1983.

[21] P. N. Sheth and J. Uicker, J. J., “IMP (Integrated Mechanisms Program), A Computer-Aided Design Analysis System for Mechanisms and Linkage,” *Journal of Engineering for Industry*, vol. 94, pp. 454–464, 05 1972.

[22] M. A. Chace and D. A. Smith, “DAMN—A Digital Computer Program for the Dynamic Analysis of Generalized Mechanical Systems,” *SAE*, vol. 80, pp. 696–991, 1971.

[23] B. Paul, “Dynamic analysis of planar machines with specified force inputs,” *Proceedings of the Third Applied Mechanics Conference*, 1973.

[24] B. Paul, “Dynamic analysis of machinery via program dymac,” *SAE Technical Papers*, 1977.

[25] Y. Stepanenko and M. Vukobratović, “Dynamics of articulated open-chain active mechanisms,” *Mathematical Biosciences*, vol. 28, no. 1-2, p. 137 – 170, 1976.

[26] M. Vukobratović, D. Stokić, and D. Hristić, “New control concept of anthropomorphic manipulators,” *Mechanism and Machine Theory*, vol. 12, no. 5, p. 515 – 530, 1977.

[27] M. Vukobratović, “Dynamics of active articulated mechanisms and synthesis of artificial motion,” *Mechanism and Machine Theory*, vol. 13, no. 1, p. 1 – 18, 1978.

[28] R. Waters, “Mechanical arm control,” *M.I.T. Artificial Intelligence Lab. Memo.*, 1979.

[29] W. Armstrong, “Recursive solution to the equations of motion of n-link manipulator,” *Proceedings 5th WorldCongress on Theory of Machines and Mechanisms*, 1979.

[30] J. Luh, M. Walker, and R. Paul, “On-line computational scheme for mechanical manipulators,” *Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME*, vol. 102, no. 2, p. 69 – 76, 1980.

[31] M. Walker and D. Orin, “Efficient dynamic computer simulation of robotic mechanisms,” *Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME*, vol. 104, no. 3, p. 205 – 211, 1982.

[32] J. Hollerbach, “A recursive Lagrangian formulation of maniputator dynamics and a comparative study of dynamics formulation complexity,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-10, no. 11, p. 730 – 736, 1980.

[33] W. Jerkovsky, “The structure of multibody dynamics equations,” *Journal of Guidance, Control, and Dynamics*, vol. 1, no. 3, p. 173 – 182, 1978.

[34] R. Featherstone, *Robot dynamics algorithms.* Kluwer Academic Publishers, Dordrecht, The Netherlands, 1987.

[35] D.-S. Bae and E. Haug, “A recursive formulation for constrained mechanical system dynamics: Part I. Open loop systems,” *Mechanics of Structures and Machines*, vol. 15, no. 3, pp. 359–382, 1987.

[36] D.-S. Bae and E. Haug, “A recursive formulation for constrained mechanical system dynamics: Part II. Closed loop systems,” *Mechanics of Structures and Machines*, vol. 15, no. 4, pp. 481–506, 1988.

[37] D.-S. Bae and E. Haug, “A recursive formulation for constrained mechanical system dynamics: Part III. Parallel processor implementation,” *Mechanics of Structures and Machines*, vol. 16, no. 2, p. 249 – 269, 1988.

[38] S. Kim and E. Haug, “A recursive formulation for flexible multibody dynamics, part I: Open-loop systems,” *Computer Methods in Applied Mechanics and Engineering*, vol. 71, no. 3, p. 293 – 314, 1988.

[39] S. Kim and E. Haug, “A recursive formulation for flexible multibody dynamics, part II: Closed loop systems,” *Computer Methods in Applied Mechanics and Engineering*, vol. 74, no. 3, p. 251 – 269, 1989.

[40] H. Lai, E. Haug, S. Kim, and D. Bae, “A decoupled flexible-relative co-ordinate recursive approach for flexible multibody dynamics,” *International Journal for Numerical Methods in Engineering*, vol. 32, no. 8, p. 1669 – 1689, 1991.

[41] S. S. Kim and M. J. Vanderploeg, “A general and efficient method for dynamic analysis of mechanical systems using velocity transformations,” *Journal of Mechanical Design, Transactions of the ASME*, vol. 108, no. 2, p. 176 – 182, 1986.

[42] P. Nikravesh and G. Gim, “Systematic construction of the equations of motion for multibody systems containing closed kinematic loops,” *Journal of Mechanical Design, Transactions of the ASME*, vol. 115, no. 1, p. 143 – 149, 1993.

[43] A. Jain, “Unified formulation of dynamics for serial rigid multibody systems,” *Journal of Guidance, Control, and Dynamics*, vol. 14, no. 3, p. 531 – 542, 1991.

[44] A. Avello, J. Jiménez, E. Bayo, and J. Garcia de Jalon, “A simple and highly parallelizable method for real-time dynamic simulation based on velocity transformations,” *Computer Methods in Applied Mechanics and Engineering*, vol. 107, no. 3, p. 313 – 339, 1993.

[45] J. M. Jimenez, *Formulaciones cinemáticas y dinámicas para la simulación en tiempo real de sistemas de sólidos rígidos.* PhD thesis, Universidad de Navarra, 1993.

[46] J. Rodríguez, *Análisis eficiente de mecanismos 3D con metodos topológicos y tecnología de componentes en Internet.* PhD thesis, University of Navarre, 2000.

[47] D. Negrut, R. Serban, and F. Potra, "A topology-based approach to exploiting sparsity in multibody dynamics: Joint formulation," *Mechanics of Structures and Machines*, vol. 25, no. 2, pp. 221–241, 1997.

[48] D. Bae, J. Han, and H. Yoo, "A generalized recursive formulation for constrained mechanical system dynamics," *Mechanics of Structures and Machines*, vol. 27, no. 3, pp. 293–315, 1999.

[49] D. S. Bae, J. M. Han, J. H. Choi, and S. M. Yang, "A generalized recursive formulation for constrained flexible multibody dynamics," *International Journal for Numerical Methods in Engineering*, vol. 50, no. 8, pp. 1841–1859, 2001.

[50] S. Kim, "A subsystem synthesis method for efficient vehicle multibody dynamics," *Multibody System Dynamics*, vol. 7, no. 2, p. 189 – 207, 2002.

[51] K. Anderson, "An order n formulation for the motion simulation of general multi-rigid-body constrained systems," *Computers and Structures*, vol. 43, no. 3, p. 565 – 579, 1992.

[52] K. Anderson and J. Critchley, "Improved 'order-n' performance algorithm for the simulation of constrained multi-rigid-body dynamic systems," *Multibody System Dynamics*, vol. 9, no. 2, p. 185 – 212, 2003.

[53] J. Cuadrado, J. Cardenal, P. Morer, and E. Bayo, "Intelligent simulation of multibody dynamics: Space-state and descriptor methods in sequential and parallel computing environments," *Multibody System Dynamics*, vol. 4, pp. 55–73, 02 2000.

[54] D. Dopico, *Formulaciones semi-recursivas y de penalización para la dinámica en tiempo real de sistemas multicuerpo.* PhD thesis, Universidade da Coruña, 2004.

[55] J. Garcia De Jalon, E. Alvarez, F. De Ribera, I. Rodriguez, and F. Funes, "A fast and simple semi-recursive formulation for multi-rigid-body systems," *Computational Methods in Applied Sciences*, vol. 2, pp. 1–23, 2005.

[56] F. J. Funes and J. García de Jalón, "An efficient dynamic formulation for solving rigid and flexible multibody systems based on semirecursive method and implicit integration," *Journal of Computational and Nonlinear Dynamics*, vol. 11, no. 5, 2016.

[57] Y. Pan, W. Dai, Y. Xiong, S. Xiang, and A. Mikkola, "Tree-topology-oriented modeling for the real-time simulation of sedan vehicle dynamics using independent coordinates and the rod-removal technique," *Mechanism and Machine Theory*, vol. 143, 2020.

[58] J. Cuadrado and D. Dopico, “A hybrid global-topological real-time formulation for multibody systems,” vol. 5 A, p. 115 – 121, 2003.

[59] J. Cuadrado, D. Dopico, M. Gonzalez, and M. Naya, “A combined penalty and recursive real-time formulation for multibody dynamics,” *Journal of Mechanical Design, Transactions of the ASME*, vol. 126, no. 4, p. 602 – 608, 2004.

[60] D. Dopico Dopico, Á. López Varela, and A. Luaces Fernández, “Augmented Lagrangian index-3 semi-recursive formulations with projections,” *Multibody System Dynamics*, 2020.

[61] S. Jaiswal, J. Rahikainen, Q. Khadim, J. Sopanen, and A. Mikkola, “Comparing double-step and penalty-based semirecursive formulations for hydraulically actuated multibody systems in a monolithic approach,” *Multibody System Dynamics*, vol. 52, no. 2, p. 169 – 191, 2021.

[62] J. Martins, P. Sturdza, and J. Alonso, “The complex-step derivative approximation,” *ACM Transactions on Mathematical Software*, vol. 29, no. 3, p. 245 – 262, 2003.

[63] T. Maly and L. R. Petzold, “Numerical methods and software for sensitivity analysis of differential-algebraic systems,” *Applied Numerical Mathematics*, vol. 20, no. 1, pp. 57–79, 1996.

[64] W. Feehery, J. Tolsma, and P. Barton, “Efficient sensitivity analysis of large-scale differential-algebraic systems,” *Applied Numerical Mathematics*, vol. 25, no. 1, p. 41 – 54, 1997.

[65] A. Callejo, S. Narayanan, J. García De Jalón, and B. Norris, “Performance of automatic differentiation tools in the dynamic simulation of multibody systems,” *Advances in Engineering Software*, vol. 73, p. 35 – 44, 2014.

[66] C. Bischof, G. Corliss, A. Griewank, P. Hovland, and A. Carle, “Adifor—generating derivative codes from fortran programs,” *Scientific Programming*, vol. 1, no. 1, p. 11 – 29, 1992.

[67] A. Griewank, D. Juedes, and J. Utke, “Algorithm 755: ADOL-C: A package for the automatic differentiation of algorithms written in c/c++,” *ACM Transactions on Mathematical Software*, vol. 22, no. 2, p. 131 – 167, 1996.

[68] S. Narayanan, B. Norris, and B. Winnicka, “ADIC2: Development of a component source transformation system for differentiating c and c++,” *Procedia Computer Science*, vol. 1, no. 1, pp. 1845–1853, 2010.

[69] A. Dürrbaum, W. Klier, and H. Hahn, “Comparison of automatic and symbolic differentiation in mathematical modeling and computer simulation of rigid-body systems,” *Multibody System Dynamics*, vol. 7, no. 4, p. 331 – 355, 2002.

[70] A. Callejo, J. García de Jalón, P. Luque, and D. A. Mántaras, "Sensitivity-based, multi-objective design of vehicle suspension systems," *Journal of Computational and Nonlinear Dynamics*, vol. 10, p. 031008, May 2015.

[71] A. Callejo and D. Dopico, "Direct sensitivity analysis of multibody systems: A vehicle dynamics benchmark," *Journal of Computational and Nonlinear Dynamics*, vol. 14, pp. 021004–021004–9, Jan. 2019.

[72] J. Ambrósio, M. Neto, and R. Leal, "Optimization of a complex flexible multibody systems with composite materials," *Multibody System Dynamics*, vol. 18, no. 2, p. 117 – 144, 2007.

[73] E. Haug and J. Arora, "Design sensitivity analysis of elastic mechanical systems," *Computer Methods in Applied Mechanics and Engineering*, vol. 15, no. 1, p. 35 – 62, 1978.

[74] J. Arora and E. Haug, "Methods of design sensitivity analysis in structural optimization," *AIAA Journal*, vol. 17, no. 9, pp. 970–974, 1979.

[75] E. Haug and B. Rousselet, "Design sensitivity analysis in structural mechanics. I. Static response variations," *Journal of Structural Mechanics*, vol. 8, no. 1, p. 17 – 41, 1980.

[76] E. Haug and B. Rousselet, "Design sensitivity analysis in structural mechanics. II. Eigenvalue variations," *Journal of Structural Mechanics*, vol. 8, no. 2, p. 161 – 186, 1980.

[77] B. Rousselet and E. Haug, "Design sensitivity analysis in structural mechanics. III. Effects of shape variation," *Journal of Structural Mechanics*, vol. 10, no. 3, p. 273 – 310, 1982.

[78] E. J. Haug, K. Neel, and P. Krishnaswami, "Design sensitivity analysis and optimization of dynamically driven systems.," vol. 9, p. 555 – 635, 1984.

[79] E. Haug, *Computer aided optimal design: structural and mechanical systems*, ch. Design sensitivity analysis of dynamic systems, pp. 705–755. No. 27 in NATO ASI series. Series F, Computer and systems sciences, Springer-Verlag, 1987.

[80] R. Wehage and E. Haug, "Generalized coordinate partitioning for dimension reduction in analysis of constrained dynamic systems," *Journal of Mechanical Design, Transactions of the ASME*, vol. 104, no. 1, p. 247 – 255, 1982.

[81] N. Mani and E. Haug, "Singular value decomposition for dynamic system design sensitivity analysis," *Engineering with Computers*, vol. 1, no. 2, p. 103 – 109, 1985.

[82] H. Ashrafiuon and N. Mani, "Analysis and optimal design of spatial mechanical systems," *Journal of Mechanical Design, Transactions of the ASME*, 1990.

[83] D. Bestle and J. Seybold, “Sensitivity analysis of constrained multibody systems,” *Archive of Applied Mechanics*, vol. 62, no. 3, p. 181 – 190, 1992.

[84] J. Dias and M. Pereira, “Sensitivity analysis of rigid-flexible multibody systems,” *Multibody System Dynamics*, vol. 1, pp. 303–322, 1997.

[85] E. Lund and N. Olhoff, “Shape design sensitivity analysis of eigenvalues using ”exact” numerical differentiation of finite element matrices,” *Structural Optimization*, vol. 8, no. 1, p. 52 – 59, 1994.

[86] J. Pagalday and A. Avello, “Optimization of multibody dynamics using object oriented programming and a mixed numerical-symbolic penalty formulation,” *Mechanism and Machine Theory*, vol. 32, no. 2, pp. 161–174, 1997.

[87] R. Serban and E. Haug, “Kinematic and kinetic derivatives in multibody system analysis,” *Mechanics of Structures and Machines*, vol. 26, no. 2, p. 145 – 173, 1998.

[88] X. Wang, E. J. Haug, and W. Pan, “Implicit numerical integration for design sensitivity analysis of rigid multibody systems,” *Mechanics Based Design of Structures and Machines*, vol. 33, no. 1, pp. 1–30, 2005.

[89] M. Schulz and H. Brauchli, “Two methods of sensitivity analysis for multibody systems with collisions,” *Mechanism and Machine Theory*, vol. 35, pp. 1345–1365, Oct. 2000.

[90] S. Li and L. Petzold, “Software and algorithms for sensitivity analysis of large-scale differential algebraic systems,” *Journal of Computational and Applied Mathematics*, vol. 125, pp. 131–145, 2000.

[91] Y. Cao, S. Li, and L. Petzold, “Adjoint sensitivity analysis for differential-algebraic equations: algorithms and software,” *Journal of Computational and Applied Mathematics*, vol. 149, no. 1, pp. 171 – 191, 2002.

[92] Y. Cao, S. Li, L. Petzold, and R. Serban, “Adjoint sensitivity analysis or differential-algebraic equations: The adjoint dae system and its numerical solution,” *SIAM Journal on Scientific Computing*, vol. 24, no. 3, pp. 1076–1089, 2003.

[93] A. Schaffer, “Stability of the adjoint differential-algebraic equation of the index-3 multibody system equation of motion,” *SIAM Journal on Scientific Computing*, vol. 26, no. 4, p. 1432 – 1448, 2005.

[94] A. S. Schaffer, *On the adjoint formulation of design sensitivity analysis of multibody dynamics.* PhD thesis, University of Iowa, 12 2005.

[95] A. Schaffer, “Stabilized index-1 differential-algebraic formulations for sensitivity analysis of multi-body dynamics,” *Proceedings of the Institution of Mechanical Engineers Part K- Journal of Multi-Body Dynamics*, vol. 220, pp. 141–156, SEP 2006.

[96] A. Sandu, “On the properties of Runge-Kutta discrete adjoints,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 3994 LNCS - IV, p. 550 – 557, 2006.

[97] T. Lauss, S. Oberpeilsteiner, W. Steiner, and K. Nachbagauer, “The discrete adjoint gradient computation for optimization problems in multibody dynamics,” *Journal of Computational and Nonlinear Dynamics*, vol. 12, no. 3, 2017.

[98] A. Callejo, V. Sonneville, and O. Bauchau, “Discrete adjoint method for the sensitivity analysis of flexible multibody systems,” *Journal of Computational and Nonlinear Dynamics*, vol. 14, no. 2, 2019.

[99] K. S. Anderson and Y. Hsu, “Analytical fully-recursive sensitivity analysis for multibody dynamic chain systems,” *Multibody System Dynamics*, vol. 8, pp. 1–27, 2002.

[100] K. Anderson and Y. Hsu, “Order-(n+m) direct differentiation determination of design sensitivity for constrained multibody dynamic systems,” *Structural and Multidisciplinary Optimization*, vol. 26, no. 3-4, pp. 171–182, 2004.

[101] K. D. Bhalerao, M. Poursina, and K. S. Anderson, “An efficient direct differentiation approach for sensitivity analysis of flexible multibody systems,” *Multibody System Dynamics*, vol. 23, no. 2, p. 121 – 140, 2010.

[102] M. Gutiérrez-López, A. Callejo, and J. García de Jalón, “Computation of independent sensitivities using Maggi’s formulation,” 05 2012.

[103] V. Sonneville and O. Brüls, “Sensitivity analysis for multibody systems formulated on a Lie group,” *Multibody System Dynamics*, vol. 31, no. 1, p. 47 – 67, 2014.

[104] S. Corner, A. Sandu, and C. Sandu, “Adjoint sensitivity analysis of hybrid multibody dynamical systems,” *Multibody System Dynamics*, vol. 49, no. 4, p. 395 – 420, 2020.

[105] R. Serban, “A parallel computational model for sensitivity analysis in optimization for robustness,” *Optimization Methods and Software*, vol. 24, no. 1, pp. 105–121, 2009.

[106] P. Maciag, P. Malczyk, and J. Fraczek, “Hamiltonian direct differentiation and adjoint approaches for multibody system sensitivity analysis,” *International Journal for Numerical Methods in Engineering*, vol. 121, no. 22, p. 5082 – 5100, 2020.

[107] D. Dopico, Y. Zhu, A. Sandu, and C. Sandu, "Direct and adjoint sensitivity analysis of ordinary differential equation multibody formulations," *Journal of Computational and Nonlinear Dynamics*, vol. 10, pp. 1–8, Sept. 2014.

[108] J. Nocedal and S. Wright, *Numerical Optimization.* Springer-Verlag New York, second ed., 2006.

[109] J. M. Pagalday, *Optimización del comportamiento dinámico de mecanismos.* PhD thesis, University of Navarre, 1994.

[110] J. S. Arora, ed., *Introduction to Optimum Design (Fourth Edition).* Boston: Academic Press, fourth edition ed., 2017.

[111] R. Fletcher, ed., *Practical methods of optimization.* Great Britain: John Wiley & Sons Ltd, second edition ed., 1987.

[112] J. Betts, ed., *Practical methods for optimal control using nonlinear programming.* Philadelphia: Society for Industrial and Applied Mathematics, first edition ed., 2001.

[113] R. Featherstone, *Rigid body dynamics algorithms.* Springer Science+Business Media, New York, USA, 2008.

[114] E. J. Haug, *Computer Aided Kinematics and Dynamics of Mechanical Systems: Basic Methods.* Allyn and Bacon series in engineering, Boston: Prentice Hall College Div, March 1989.

[115] J. Cuadrado, D. Dopico, M. Naya, and M. Gonzalez, "Penalty, semi-recursive and hybrid methods for mbs real-time dynamics in the context of structural integrators," *Multibody System Dynamics*, vol. 12, no. 2, pp. 117–132, 2004.

[116] J. I. Rodriguez, J. M. Jimenez, F. J. Funes, and J. G. de Jalón, "Recursive and residual algorithms for the efficient numerical integration of multi-body systems," *Multibody System Dynamics*, vol. 11, pp. 295–320, May 2004.

[117] J. García de Jalón and E. Bayo, *Kinematic and dynamic simulation of multibody systems: The real-time challenge.* New York (USA): Springer-Verlag, 1994.

[118] J. Garcia De Jalon, A. Callejo, A. Hidalgo, and M. Gutierrez, "Efficient solution of Maggi's equations," *Proceedings of the ASME Design Engineering Technical Conference*, vol. 4, no. PARTS A AND B, pp. 115–124, 2011.

[119] E. Bayo and R. Ledesma, "Augmented Lagrangian and mass-orthogonal projection methods for constrained multibody dynamics," *Nonlinear Dynamics*, vol. 9, pp. 113–130, 02 1996.

[120] D. Dopico, F. González, J. Cuadrado, and J. Kövecses, "Determination of holonomic and nonholonomic constraint reactions in an index-3 augmented Lagrangian formulation with velocity and acceleration projections," *Journal of Computational and Nonlinear Dynamics*, vol. 9, p. 041006, July 2014.

[121] J. Cuadrado, R. Gutiérrez, M. Naya, and M. González, "Experimental validation of a flexible MBS dynamic formulation through comparison between measured and calculated stresses on a prototype car," *Multibody System Dynamics*, vol. 11, no. 2, pp. 147–166, 2004.

[122] R. Gutiérrez, *Cálculo de tensiones en componentes de sistemas móviles mediante dinámica de sistemas multicuerpo flexibles.* PhD thesis, Universidade da Coruña, 2003.

[123] U. Lugrís, *Real-time methods in flexible multibody dynamics.* PhD thesis, Universidade da Coruña, 2008.

[124] M. A. Naya, *Aplicación de la Dinámica Multicuerpo en Tiempo Real a la Simulación y el Control de Automóviles.* PhD thesis, Universidade da Coruña, 2007.

[125] E. Bayo, J. Garcia De Jalon, and M. A. Serna, "A modified Lagrangian formulation for the dynamic analysis of constrained mechanical systems," *Computer Methods in Applied Mechanics and Engineering*, vol. 71, no. 2, pp. 183–195, 1988.

[126] J. C. García Orden, "Energy considerations for the stabilization of constrained mechanical systems with velocity projection," *Nonlinear Dynamics*, vol. 60, no. 1-2, p. 49 – 62, 2010.

[127] J. Orden and D. Dopico, "On the stabilizing properties of energy-momentum integrators and coordinate projections for constrained mechanical systems," *Computational Methods in Applied Sciences*, vol. 4, pp. 49–67, 2007.

[128] T. Lauss, S. Oberpeilsteiner, W. Steiner, and K. Nachbagauer, "The discrete adjoint method for parameter identification in multibody system dynamics," *Multibody System Dynamics*, vol. 42, no. 4, pp. 397–410, 2018.

[129] J.-F. Collard, *Geometrical and Kinematic Optimization of Closed-Loop Multibody Systems.* PhD thesis, Université Catholique de Louvain, 2007.

[130] D. Liu and J. Nocedal, "On the limited memory bfgs method for large scale optimization," *Mathematical Programming*, vol. 45, no. 1-3, pp. 503–528, 1989.

[131] C. Zhu, R. Byrd, P. Lu, and J. Nocedal, "Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization," *ACM Transactions on Mathematical Software*, vol. 23, no. 4, pp. 550–560, 1997.

[132] J. Morales and J. Nocedal, “Remark on ”algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound constrained optimization”,” *ACM Transactions on Mathematical Software*, vol. 38, no. 1, 2011.

[133] Y. Zhu, D. Dopico, C. Sandu, and A. Sandu, “Dynamic response optimization of complex multibody systems in a penalty formulation using adjoint sensitivity,” *Journal of Computational and Nonlinear Dynamics*, vol. 10, pp. 1–9, May 2015.

[134] I. T. C. for Multibody Dynamics., “Library of computational benchmark problems. url: `http://www.iftomm-multibody.org/benchmark`.” `http://www.iftomm-multibody.org/benchmark`, 2014.

[135] R. Patorino, *Experimental validation of a multibody model for a vehicle prototype and its application to automotive state observers.* PhD thesis, 2012.

[136] E. Sanjurjo, *State observers based on detailed multibody models applied to an automobile.* PhD thesis, Universidade da Coruña, 2016.

[137] P. Flores and H. Lankarani, *Contact Force Models for Multibody Dynamics.* Springer International Publishing, 2016.

[138] P. Luque, D. Alvarez, and C. Vera, *Ingenieria del automovil. Sistemas y comportamiento dinamico.* Thomson, 2004.

[139] AENOR, “Vehículos de carretera. Turismos. Pista de pruebas para un cambio brusco de carril. Parte1: Doble cambio de carril. UNE 26514-1. Madrid: AENOR,” 2003.

[140] T. Guillespie, *Fundamentals of vehicle dynamics.* SAE International, USA, 1992.

[141] A. Schwab and J. Meijaard, “A review on bicycle dynamics and rider control,” *Vehicle System Dynamics*, vol. 51, 07 2013.

[142] *Rider Optimal Control Identification in Bicycling*, vol. Volume 3: Renewable Energy Systems; Robotics; Robust Control; Single Track Vehicle Dynamics and Control; Stochastic Models, Control and Algorithms in Robotics; Structure Dynamics and Smart Structures; Surgical Robotics; Tire and Suspension Systems Modeling; Vehicle Dynamics and Control; Vibration and Energy; Vibration Control of *Dynamic Systems and Control Conference*, 10 2012.

[143] E. Sanjurjo, M. Naya, J. Cuadrado, and A. Schwab, “Roll angle estimator based on angular rate measurements for bicycles,” *Vehicle System Dynamics*, vol. 57, no. 11, pp. 1705–1719, 2019.

[144] J. Meijaard, J. M. Papadopoulos, A. Ruina, and A. Schwab, “Linearized dynamics equations for the balance and steer of a bicycle: a benchmark and review,” *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 463, no. 2084, pp. 1955–1982, 2007.

[145] D. Dopico, A. Luaces, M. Saura, J. Cuadrado, and D. Vilela, "Simulating the anchor lifting maneuver of ships using contact detection techniques and continuous contact force models," *Multibody System Dynamics*, 2019.

# Appendix A

# Derivative of angular velocity with respect to positions of relative coordinates

Angular velocity is an omnipresent piece in joint coordinate formulations. It is involved in the recursive expressions of each joint, i.e. in $\dot{\mathbf{b}}_i^v$ and $\mathbf{d}_i^v$, in the determination of the generalized coordinates vector and in the vector of Cartesian velocities, namely $\mathbf{V}$, $\mathbf{Y}$ or $\mathbf{Z}$. Hence, it is present in the computation of a sensitivity analysis in the form of derivatives with respect to positions and velocities of relative coordinates.

Even though the derivative of this magnitude with respect to velocities of joint coordinates is straightforward using the recursive relations (2.204), the derivative with respect to positions is slightly more intricate, and it becomes even more complicated if the differentiation of full matrices $\mathbf{R}_i^v$ is intended to be avoided.

The easiest path to get the derivative of the angular velocity with respect to positions could be to compute it differentiating with respect to time the partial derivative of the angular velocity with respect to joint velocities, i.e.:

$$\left(\boldsymbol{\omega}_i\right)_{\hat{\mathbf{z}}} \neq \left(\dot{\boldsymbol{\omega}}_i\right)_{\hat{\dot{\mathbf{z}}}} = \frac{\mathrm{d}}{\mathrm{d}t}\left(\left(\boldsymbol{\omega}_i\right)_{\hat{\dot{\mathbf{z}}}}\right) = \begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix}\dot{\mathbf{R}}_i^v \tag{A.1}$$

However, as it is indicated in A.1, this relation is not true, since in this equation there are dependencies on $\boldsymbol{\omega}_i$ that are not being considered.

Let us consider the computation of the angular velocity of a body using the relative velocities of each joint of the kinematic chain, using (2.107), (2.113) and (2.117):

$$\boldsymbol{\omega}_i = \sum_{j=1}^{i} \boldsymbol{\omega}_j^{j-1} = \sum_{j=1}^{i} \mathbf{b}_j^{v2}\dot{\mathbf{z}}_j \tag{A.2}$$

being $\mathbf{b}_j^{v2}$ the angular part of the term $\mathbf{b}_j^v$, corresponding to its fourth to sixth rows, according to the notation introduced in 4.4.3, and $j$ represents each one of the previous bodies of body $i$ in its kinematic chain. Taking derivatives with respect to relative coordinates in positions:

$$\left(\boldsymbol{\omega}_i\right)_{\hat{\mathbf{z}}} = \sum_{j=1}^{i}\left(\boldsymbol{\omega}_j^{j-1}\right)_{\hat{\mathbf{z}}} = \sum_{j=1}^{i}\left(\mathbf{b}_j^{v2}\right)_{\hat{\mathbf{z}}}\dot{\mathbf{z}}_j \tag{A.3}$$

with:

$$\mathbf{b}_j^{v2} = \begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix}\mathbf{b}_j^{v} = \begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix}\mathbf{b}_j^{y} = \begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix}\mathbf{b}_j^{z} \tag{A.4}$$

Now, the derivative of the angular velocity is expressed in terms of derivatives of $\mathbf{b}_j^{v2}$, which have the known expressions presented in 4.4.3. However, the simplification can be pulled forward, obtaining a derivative of $\mathbf{b}_j^{v}$ dependent on $\mathbf{R}_i^{v}$ for all the joints. Returning to 4.4.3, the derivatives of $\mathbf{b}_j^{v}$ not expressed in terms of $\mathbf{R}_i^{v}$ are the ones related to the spherical and floating joints, since they have parts dependent on the rotation matrix, which is not a function of $\mathbf{R}_i^{v}$ or even the recursive terms $\mathbf{b}_i^{v}$. Nevertheless, there is a method to obtain the derivative of the rotation matrix involving $\mathbf{R}_i^{v}$ by means of identification of terms using two different schemes of differentiation.

First, let us consider a vector $\mathbf{v}_j$ belonging to body $i$. Its position can be obtained as:

$$\mathbf{v}_j = \mathbf{A}_i\bar{\mathbf{v}}_j^{i} \tag{A.5}$$

and its velocity as:

$$\dot{\mathbf{v}}_j = \tilde{\boldsymbol{\omega}}_i\mathbf{v}_j \tag{A.6}$$

Now, the derivative of the position of a vector can be computed, using A.5 as:

$$\frac{\partial \mathbf{v}_j}{\partial \mathbf{z}} = \frac{\partial \mathbf{A}_i}{\partial \mathbf{z}}\bar{\mathbf{v}}_j^{i} \tag{A.7}$$

However, the expression in velocities is also valid:

$$\frac{\partial \mathbf{v}_j}{\partial \mathbf{z}} = \frac{\partial \dot{\mathbf{v}}_j}{\partial \dot{\mathbf{z}}} = \begin{bmatrix}\mathbf{0} & -\tilde{\mathbf{v}}_j\end{bmatrix}\mathbf{R}_i^{v} \tag{A.8}$$

Using the properties of the skew symmetric tensor of a matrix, (A.8) can be expressed as:

$$\frac{\partial \mathbf{v}_j}{\partial \mathbf{z}} = \tilde{\mathbf{R}}_i^{v2}\mathbf{v}_j = \tilde{\mathbf{R}}_i^{v2}\mathbf{A}_i\bar{\mathbf{v}}_j^{i} \tag{A.9}$$

in which:

$$\mathbf{R}_i^{v2} = \begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix}\mathbf{R}_i \tag{A.10}$$

Identifying terms between (A.9) and (A.7), the following expression of the partial derivative of the rotation matrix is achieved:

$$\frac{\partial \mathbf{A}_i}{\partial \mathbf{z}} = \tilde{\mathbf{R}}_i^{v2}\mathbf{A}_i \tag{A.11}$$

Now, the angular part of $\mathbf{b}_i^{v}$ in spherical and floating joints, this is, $2\mathbf{E}$, can be differentiated with respect to the relative coordinates of the model as:

$$\begin{gathered}\frac{\partial\left(2\mathbf{E}_i\right)}{\partial \mathbf{z}} = 2\frac{\partial \mathbf{A}_{i-1}}{\partial \mathbf{z}}\bar{\mathbf{E}}_i + 2\mathbf{A}_{i-1}\frac{\partial \bar{\mathbf{E}}_i}{\partial \mathbf{z}} = 2\tilde{\mathbf{R}}_{i-1}^{v2}\mathbf{A}_{i-1}\bar{\mathbf{E}}_i + 2\mathbf{A}_{i-1}\frac{\partial \bar{\mathbf{E}}_i}{\partial \mathbf{z}} = \\ 2\tilde{\mathbf{R}}_{i-1}^{v2}\mathbf{E}_i + 2\mathbf{A}_{i-1}\frac{\partial \bar{\mathbf{E}}_i}{\partial \mathbf{z}} = 2\tilde{\mathbf{E}}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\mathbf{R}_{i-1} + 2\mathbf{A}_{i-1}\frac{\partial \bar{\mathbf{E}}_i}{\partial \mathbf{z}}\end{gathered} \tag{A.12}$$

Equation (A.12) contributes to the simplification of the derivative of the angular part of $\mathbf{b}_i^v$, allowing to express it generally as:

$$\left(\mathbf{b}_j^{v2}\right)_{\hat{\mathbf{z}}} = \begin{cases} \widetilde{\mathbf{b}_j^{v2}} \begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix} \mathbf{R}_{j-1} & \text{in case 1} \\ \\ \widetilde{\mathbf{b}_j^{v2}} \begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix} \mathbf{R}_{j-1} + 2\mathbf{A}_{j-1}\dfrac{\partial \bar{\mathbf{E}}_j}{\partial \mathbf{z}} & \text{in case 2} \end{cases} \tag{A.13}$$

in which case 1 groups revolute, prismatic, Cardan, cylindrical and planar joints, and case 2 gathers spherical and floating joints. Observe that if the joint considered has more than one coordinate, the terms $\widetilde{\mathbf{b}_j^{v2}}$ are tensors of dimension 3 by 3 by $n_v$, being $n_v$ the number of variables describing the motion of this joint.

The derivative of the angular velocity can be then recalculated using the previous relation:

$$\left(\boldsymbol{\omega}_i\right)_{\hat{\mathbf{z}}} = \sum_{j=1}^{i} \left(\widetilde{\mathbf{b}_j^{v2}} \begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix} \mathbf{R}_{j-1}\right) \dot{\mathbf{z}}_j + \sum_{k=1}^{i} \left(2\mathbf{A}_{i-1}\frac{\partial \bar{\mathbf{E}}_k}{\partial \mathbf{z}}\right) \dot{\mathbf{z}}_k \tag{A.14}$$

Now, since:

$$\mathbf{b}_j^{v2}\dot{\mathbf{z}}_j = \boldsymbol{\omega}_j^{j-1} \tag{A.15}$$

equation (A.14) can be transformed into:

$$\begin{aligned} \frac{\partial \boldsymbol{\omega}_i}{\partial \mathbf{z}} &= \sum_{j=1}^{i} \left(\tilde{\boldsymbol{\omega}}_j^{j-1} \begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix} \mathbf{R}_{j-1}\right) + \sum_{k=1}^{i} \left(2\mathbf{A}_{i-1}\frac{\partial \bar{\mathbf{E}}_k}{\partial \mathbf{z}}\right) \dot{\mathbf{z}}_k = \\ & \sum_{j=1}^{i} \left(\tilde{\boldsymbol{\omega}}_j - \tilde{\boldsymbol{\omega}}_{j-1}\right) \begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix} \mathbf{R}_{i-1} + \sum_{k=1}^{i} \left(2\mathbf{A}_{k-1}\frac{\partial \bar{\mathbf{E}}_k}{\partial \mathbf{z}}\right) \dot{\mathbf{z}}_k \end{aligned} \tag{A.16}$$

The recursive assembly of the matrix $\mathbf{R}_{j-1}$ using the relative recursive terms of the previous joints can be recalled to reformulate the previous expression:

$$\begin{aligned} \left(\boldsymbol{\omega}_i\right)_{\hat{\mathbf{z}}} &= \sum_{j=1}^{i} \left(\tilde{\boldsymbol{\omega}}_j - \tilde{\boldsymbol{\omega}}_{j-1}\right) \left(-\sum_{k=1}^{j} \mathbf{b}_k^{v2}\right) + \sum_{k=1}^{i} \left(2\mathbf{A}_{i-1}\frac{\partial \bar{\mathbf{E}}_k}{\partial \mathbf{z}}\right) \dot{\mathbf{z}}_k = \\ & \sum_{j=1}^{i} \left(\tilde{\boldsymbol{\omega}}_{j-1} - \tilde{\boldsymbol{\omega}}_i\right) \mathbf{b}_j^{v2} + \sum_{k=1}^{i} \left(2\mathbf{A}_{i-1}\frac{\partial \bar{\mathbf{E}}_k}{\partial \mathbf{z}}\right) \dot{\mathbf{z}}_k \end{aligned} \tag{A.17}$$

Now note that:

$$\sum_{j=1}^{i} \tilde{\boldsymbol{\omega}}_i \mathbf{b}_j^{v2} = \tilde{\boldsymbol{\omega}}_i \begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix} \mathbf{R}_i \tag{A.18a}$$

$$\sum_{j=1}^{i} \tilde{\boldsymbol{\omega}}_{j-1} \mathbf{b}_j^{v2} + \sum_{k=1}^{i} \left(2\mathbf{A}_{i-1}\frac{\partial \bar{\mathbf{E}}_k}{\partial \mathbf{z}}\right) \dot{\mathbf{z}}_k = \begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix} \dot{\mathbf{R}}_i \tag{A.18b}$$

Therefore, the derivative of the angular velocity with respect to the relative coordinates in positions can be finally formulated as:

$$\left(\boldsymbol{\omega}_i\right)_{\hat{\mathbf{z}}} = \begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix} \frac{\partial \mathbf{R}_i^v}{\partial \mathbf{z}} \dot{\mathbf{z}} = \begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix} \dot{\mathbf{R}}_i^v - \begin{bmatrix}\mathbf{0} & \tilde{\boldsymbol{\omega}}_i\end{bmatrix} \mathbf{R}_i^v \tag{A.19}$$

# Appendix B

# Math notes: skew symmetric matrix of a vector and skew symmetric tensor of a matrix

Let us consider 2 vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^3$ and a scalar $\alpha \in \mathbb{R}$. The skew symmetric matrix of a vector can be defined as a matrix $\tilde{\mathbf{u}} \in \mathbb{R}^{3\times3}$ such as:

$$\tilde{\mathbf{u}} = \begin{bmatrix} 0 & -\mathbf{u}_z & \mathbf{u}_y \\ \mathbf{u}_z & 0 & -\mathbf{u}_x \\ -\mathbf{u}_y & \mathbf{u}_x & 0 \end{bmatrix} \tag{B.1}$$

Additionally, let us consider a matrix $\mathbf{A} \in \mathbb{R}^{3\times n}$ . The skew-symmetric tensor of the matrix $\mathbf{A} \in \mathbb{R}^{3\times n}$ is defined as the tensor $\tilde{\mathbf{A}} \in \mathbb{R}^{3\times3\times n}$ composed of the skew-symmetric matrices of each column of the matrix $\mathbf{A}$.

$$\tilde{\mathbf{A}} = \begin{bmatrix} \tilde{\mathbf{A}}_1 & \tilde{\mathbf{A}}_2 & ... & \tilde{\mathbf{A}}_n \end{bmatrix} \tag{B.2}$$

where $\mathbf{A}_i$ represents the column $i$ of the matrix $\mathbf{A}$.

The skew symmetric matrix of a vector and the skew symmetric tensor of a matrix have the following properties:

1.
$$\tilde{\mathbf{u}}^{\mathrm{T}} = -\tilde{\mathbf{u}} \tag{B.3}$$

2.
$$\alpha\tilde{\mathbf{u}} = (\widetilde{\alpha\mathbf{u}}) \tag{B.4}$$

3.
$$\tilde{\mathbf{u}}\mathbf{u} = \mathbf{0} \tag{B.5}$$

4.
$$\tilde{\mathbf{u}}\mathbf{v} = -\tilde{\mathbf{v}}\mathbf{u} \tag{B.6}$$

5.
$$\tilde{\mathbf{u}}\tilde{\mathbf{v}} = (\tilde{\mathbf{v}}\tilde{\mathbf{u}})^{\mathrm{T}} \tag{B.7}$$

6.
$$\tilde{\mathbf{u}}\tilde{\mathbf{v}} = \tilde{\mathbf{v}}\tilde{\mathbf{u}} + \left(\widetilde{\tilde{\mathbf{u}}\mathbf{v}}\right) \tag{B.8}$$

7.
$$\left(\widetilde{\tilde{\mathbf{u}}\mathbf{v}}\right) = \mathbf{v}\mathbf{u}^{\mathrm{T}} - \mathbf{u}\mathbf{v}^{\mathrm{T}} = \tilde{\mathbf{u}}\tilde{\mathbf{v}} - \tilde{\mathbf{v}}\tilde{\mathbf{u}} \tag{B.9}$$

8.
$$\left(\widetilde{\mathbf{u}+\mathbf{v}}\right) = \tilde{\mathbf{u}} + \tilde{\mathbf{v}} \tag{B.10}$$

9.
$$\tilde{\mathbf{u}}\tilde{\mathbf{u}} = \mathbf{u}\mathbf{u}^{\mathrm{T}} - \mathbf{u}^{\mathrm{T}}\mathbf{u}\mathbf{I} \tag{B.11}$$

10.
$$\frac{\mathrm{d}\left(\tilde{\mathbf{u}}\mathbf{v}\right)}{\mathrm{d}t} = \tilde{\dot{\mathbf{u}}}\mathbf{v} + \tilde{\mathbf{u}}\dot{\mathbf{v}} \tag{B.12}$$

11.
$$\frac{\mathrm{d}\tilde{\mathbf{v}}}{\mathrm{d}\mathbf{x}} = \left(\widetilde{\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}\mathbf{x}}}\right) \tag{B.13}$$

12.
$$\tilde{\mathbf{M}} \otimes \mathbf{v} = -\tilde{\mathbf{v}}\mathbf{M} \tag{B.14}$$

13.
$$\tilde{\mathbf{M}} \otimes \tilde{\mathbf{v}} = \tilde{\mathbf{v}}\tilde{\mathbf{M}} + \left(\widetilde{\tilde{\mathbf{M}} \otimes \mathbf{v}}\right) \tag{B.15}$$

# Appendix C

# Derivatives of recursive kinematic relations for RTdyn0 and RTdyn1

## C.1 Expressions of $\left(\dot{\mathbf{b}}_i^y\right)_{\mathbf{z}}$ for RTdyn0

This section gathers the analytical expressions of the partial derivative of the recursive kinematic relation $\dot{\mathbf{b}}_i^y$ in the framework of the RTdyn0 accumulation, with the set of reference points coincident with the center of mass of each body. Final expressions are directly exposed without intermediate developments.

- **Revolute joint:**

$$\left(\dot{\mathbf{b}}_i^y\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{y1,1}} \\ \widetilde{\mathbf{b}_i^{y2,1}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \dot{\mathbf{R}}_i^y + \begin{bmatrix} \widetilde{\dot{\mathbf{b}}_i^{y1,1}} \\ \widetilde{\dot{\mathbf{b}}_i^{y2,1}} \end{bmatrix} \tag{C.1}$$

- **Prismatic joint:**

$$\left(\dot{\mathbf{b}}_i^y\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\dot{\mathbf{b}}_i^{y1,1}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^y + \begin{bmatrix} \widetilde{\mathbf{b}_i^{y1,1}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \dot{\mathbf{R}}_i^y \tag{C.2}$$

- **Cylindrical joint:**

$$\left(\dot{\mathbf{b}}_i^{y1}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\dot{\mathbf{b}}_i^{y1,1}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^y + \begin{bmatrix} \widetilde{\mathbf{b}_i^{y1,1}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \dot{\mathbf{R}}_i^y \tag{C.3}$$

$$\left(\dot{\mathbf{b}}_i^{y2}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\dot{\mathbf{b}}_i^{y1,2}} \\ \widetilde{\dot{\mathbf{b}}_i^{y2,2}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^y + \begin{bmatrix} \widetilde{\mathbf{b}_i^{y1,2}} \\ \widetilde{\mathbf{b}_i^{y2,2}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \dot{\mathbf{R}}_i^y \tag{C.4}$$

- **Cardan joint:**

$$\left(\dot{\mathbf{b}}_i^{y1}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix}\widetilde{\dot{\mathbf{b}}_i^{y1,1}} \\ \widetilde{\dot{\mathbf{b}}_i^{y2,1}}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\mathbf{R}_{i-1}^y + \begin{bmatrix}\widetilde{\mathbf{b}_i^{y1,1}} \\ \widetilde{\mathbf{b}_i^{y2,1}}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\dot{\mathbf{R}}_{i-1}^y$$
$$+ \begin{bmatrix}\dot{\tilde{\mathbf{w}}}_j\left(\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_j\right) + \tilde{\mathbf{w}}_j\left(\dot{\tilde{\mathbf{r}}}_G^i - \dot{\tilde{\mathbf{r}}}_j\right) \\ \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\mathbf{b}_i^{y0} + \begin{bmatrix}\tilde{\mathbf{w}}_j\left(\tilde{\mathbf{r}}_G^i - \tilde{\mathbf{r}}_j\right) \\ \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\dot{\mathbf{b}}_i^{y0} \tag{C.5}$$

$$\left(\dot{\mathbf{b}}_i^{y2}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix}\widetilde{\dot{\mathbf{b}}_i^{y1,2}} \\ \widetilde{\dot{\mathbf{b}}_i^{y2,2}}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\mathbf{R}_i^y + \begin{bmatrix}\widetilde{\mathbf{b}_i^{y1,2}} \\ \widetilde{\mathbf{b}_i^{y2,2}}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\dot{\mathbf{R}}_i^y \tag{C.6}$$

- **Spherical joint:**

$$\left(\dot{\mathbf{b}}_i^{y}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix}\left(\tilde{\boldsymbol{\omega}}_i\right)_{\hat{\mathbf{z}}} & \mathbf{0} \\ \mathbf{0} & \left(\tilde{\boldsymbol{\omega}}_i\right)_{\hat{\mathbf{z}}}\end{bmatrix}\mathbf{b}_i^y + \begin{bmatrix}\tilde{\boldsymbol{\omega}}_i & \mathbf{0} \\ \mathbf{0} & \tilde{\boldsymbol{\omega}}_i\end{bmatrix}\left(\mathbf{b}_i^y\right)_{\hat{\mathbf{z}}} \tag{C.7}$$

- **Floating joint:**

$$\left(\dot{\mathbf{b}}_i^{y}\right)_{\hat{\mathbf{z}}} = \mathbf{0}_{6\times 7\times n} \tag{C.8}$$

- **Planar joint:**

$$\left(\dot{\mathbf{b}}_i^{y1}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix}\widetilde{\dot{\mathbf{b}}_i^{y1,1}} \\ \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\mathbf{R}_{i-1}^y + \begin{bmatrix}\widetilde{\mathbf{b}_i^{y1,1}} \\ \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\dot{\mathbf{R}}_{i-1}^y \tag{C.9}$$

$$\left(\dot{\mathbf{b}}_i^{y2}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix}\widetilde{\dot{\mathbf{b}}_i^{y1,2}} \\ \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\mathbf{R}_{i-1}^y + \begin{bmatrix}\widetilde{\mathbf{b}_i^{y1,2}} \\ \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\dot{\mathbf{R}}_{i-1}^y \tag{C.10}$$

$$\left(\dot{\mathbf{b}}_i^{y3}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix}\mathbf{0} \\ \widetilde{\dot{\mathbf{b}}_i^{y2,3}}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\mathbf{R}_i^y + \begin{bmatrix}\mathbf{0} \\ \widetilde{\mathbf{b}_i^{y2,3}}\end{bmatrix}\begin{bmatrix}\mathbf{0} & -\mathbf{I}\end{bmatrix}\dot{\mathbf{R}}_i^y \tag{C.11}$$

## C.2 Expressions of $\left(\dot{\mathbf{b}}_i^{z}\right)_{\mathbf{z}}$ for RTdyn1

This section is devoted to present the derivative expressions of $\dot{\mathbf{b}}_i^z$ for RTdyn1 accumulations, i.e. for a selection of each reference point coincident with the global origin of coordinates at each time instant. The analytical derivative of the velocity of the reference point considered for the generation of expressions below is recalled here:

$$\frac{\partial \dot{\mathbf{r}}_0^i}{\partial \mathbf{z}} = \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\dot{\mathbf{R}}_i^z - \tilde{\boldsymbol{\omega}}_i\begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\mathbf{R}_i^z \tag{C.12}$$

- **Revolute joint**

$$\left(\dot{\mathbf{b}}_i^z\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\mathbf{b}_i^{z1,1}} \\ \widetilde{\mathbf{b}_i^{z2,1}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \dot{\mathbf{R}}_i^z + \begin{bmatrix} \widetilde{\dot{\mathbf{b}}_i^{z1,1}} \\ \widetilde{\dot{\mathbf{b}}_i^{z2,1}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^z + \begin{bmatrix} \dot{\tilde{\mathbf{u}}}_j \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} -\mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^z \\ + \begin{bmatrix} \tilde{\mathbf{u}}_j \\ \mathbf{0} \end{bmatrix} \tilde{\boldsymbol{\omega}}_i \begin{bmatrix} -\mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^z \tag{C.13}$$

- **Prismatic joint**

$$\left(\dot{\mathbf{b}}_i^z\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\dot{\mathbf{b}}_i^{z1,1}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^z + \begin{bmatrix} \widetilde{\mathbf{b}_i^{z1,1}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \dot{\mathbf{R}}_i^z \tag{C.14}$$

- **Cylindrical joint**

$$\left(\dot{\mathbf{b}}_i^{z1}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\dot{\mathbf{b}}_i^{z1,1}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^z + \begin{bmatrix} \widetilde{\mathbf{b}_i^{z1,1}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \dot{\mathbf{R}}_i^z \tag{C.15}$$

$$\left(\dot{\mathbf{b}}_i^{z2}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\dot{\mathbf{b}}_i^{z1,2}} \\ \widetilde{\dot{\mathbf{b}}_i^{z2,2}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^z + \begin{bmatrix} \widetilde{\mathbf{b}_i^{z1,2}} \\ \widetilde{\mathbf{b}_i^{z2,2}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \dot{\mathbf{R}}_i^z + \begin{bmatrix} \dot{\tilde{\mathbf{u}}}_j \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} -\mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^z \\ + \begin{bmatrix} \tilde{\mathbf{u}}_j \\ \mathbf{0} \end{bmatrix} \tilde{\boldsymbol{\omega}}_i \begin{bmatrix} -\mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^z \tag{C.16}$$

- **Cardan joint**

$$\left(\dot{\mathbf{b}}_i^{z1}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\dot{\mathbf{b}}_i^{z1,1}} \\ \widetilde{\dot{\mathbf{b}}_i^{z2,1}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_{i-1}^z + \begin{bmatrix} \widetilde{\mathbf{b}_i^{z1,1}} \\ \widetilde{\mathbf{b}_i^{z2,1}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \dot{\mathbf{R}}_{i-1}^z \\ + \begin{bmatrix} \dot{\tilde{\mathbf{w}}}_j \\ \mathbf{0} \end{bmatrix} \left(- \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^z\right) + \begin{bmatrix} \tilde{\mathbf{w}}_j \\ \mathbf{0} \end{bmatrix} \left(-\tilde{\boldsymbol{\omega}}_i \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^z\right) \\ + \begin{bmatrix} \dot{\tilde{\mathbf{w}}}_j \left(-\tilde{\mathbf{r}}_j\right) + \tilde{\mathbf{w}}_j \left(\dot{\tilde{\mathbf{r}}}_i - \dot{\tilde{\mathbf{r}}}_j\right) \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{b}_i^{z0} + \begin{bmatrix} \tilde{\mathbf{w}}_j \left(-\tilde{\mathbf{r}}_j\right) \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \dot{\mathbf{b}}_i^{z0} \tag{C.17}$$

$$\left(\dot{\mathbf{b}}_i^{z2}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\dot{\mathbf{b}}_i^{z1,2}} \\ \widetilde{\dot{\mathbf{b}}_i^{z2,2}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^z + \begin{bmatrix} \widetilde{\mathbf{b}_i^{z1,2}} \\ \widetilde{\mathbf{b}_i^{z2,2}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \dot{\mathbf{R}}_i^z + \begin{bmatrix} \dot{\tilde{\mathbf{w}}}_j \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} -\mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^z \\ + \begin{bmatrix} \tilde{\mathbf{w}}_j \\ \mathbf{0} \end{bmatrix} \tilde{\boldsymbol{\omega}}_i \begin{bmatrix} -\mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^z \tag{C.18}$$

- **Spherical joint**

$$\left(\dot{\mathbf{b}}_i^z\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \left(\tilde{\boldsymbol{\omega}}_i\right)_{\hat{\mathbf{z}}} & \mathbf{0} \\ \mathbf{0} & \left(\tilde{\boldsymbol{\omega}}_i\right)_{\hat{\mathbf{z}}} \end{bmatrix} \mathbf{b}_i^z + \begin{bmatrix} \tilde{\boldsymbol{\omega}}_i & \mathbf{0} \\ \mathbf{0} & \tilde{\boldsymbol{\omega}}_i \end{bmatrix} \left(\mathbf{b}_i^z\right)_{\hat{\mathbf{z}}} \tag{C.19}$$

- **Floating joint**

$$\left(\dot{\mathbf{b}}_i^z\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \left(\tilde{\boldsymbol{\omega}}_i\right)_{\hat{\mathbf{z}}} & \mathbf{0} \\ \mathbf{0} & \left(\tilde{\boldsymbol{\omega}}_i\right)_{\hat{\mathbf{z}}} \end{bmatrix} \mathbf{b}_i^z + \begin{bmatrix} \tilde{\boldsymbol{\omega}}_i & \mathbf{0} \\ \mathbf{0} & \tilde{\boldsymbol{\omega}}_i \end{bmatrix} \left(\mathbf{b}_i^z\right)_{\hat{\mathbf{z}}} \tag{C.20}$$

- **Planar joint**

$$\left(\dot{\mathbf{b}}_i^{z1}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\dot{\mathbf{b}}_i^{z1,1}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_{i-1}^z + \begin{bmatrix} \widetilde{\mathbf{b}_i^{z1,1}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \dot{\mathbf{R}}_{i-1}^z \tag{C.21}$$

$$\left(\dot{\mathbf{b}}_i^{z2}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\dot{\mathbf{b}}_i^{z1,2}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_{i-1}^z + \begin{bmatrix} \widetilde{\mathbf{b}_i^{z1,2}} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \dot{\mathbf{R}}_{i-1}^z \tag{C.22}$$

$$\begin{aligned} \left(\dot{\mathbf{b}}_i^{z3}\right)_{\hat{\mathbf{z}}} = \begin{bmatrix} \widetilde{\dot{\mathbf{b}}_i^{z1,3}} \\ \widetilde{\dot{\mathbf{b}}_i^{z2,3}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \mathbf{R}_i^z + \begin{bmatrix} \widetilde{\mathbf{b}_i^{z1,3}} \\ \widetilde{\mathbf{b}_i^{z2,3}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & -\mathbf{I} \end{bmatrix} \dot{\mathbf{R}}_i^z + \begin{bmatrix} \dot{\tilde{\mathbf{w}}}_j \\ \mathbf{0} \end{bmatrix} \left(- \begin{bmatrix} -\mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^z\right) \\ + \begin{bmatrix} \tilde{\mathbf{w}}_j \\ \mathbf{0} \end{bmatrix} \tilde{\boldsymbol{\omega}}_i \begin{bmatrix} -\mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{R}_i^z \end{aligned} \tag{C.23}$$

# Appendix D

# Semi-recursive index-1 DAE formulation

Index reduction is one of the well-known methods used to simplify the solution of systems of differential algebraic equations. Despite of the increase of robustness and the generality of the method, it has an important drawback in the generation of drift-off, which implies that small perturbations in accelerations cause a quadratic increase in the error of the constraints in positions.

Considering its properties, this formulation constitutes a suitable method for initializing the accelerations of any dynamic formulation or to compute certain instants of time, since the error generated by the drift-off in this case is not propagated nor accumulated.

The general expression of the classical Lagrange index-1 formulation (see [117]) in relative coordinates becomes:

$$\mathbf{M}^d\ddot{\mathbf{z}} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\lambda} = \mathbf{Q}^d \tag{D.1a}$$

$$\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\ddot{\mathbf{z}} = -\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\dot{\mathbf{z}} - \dot{\boldsymbol{\Phi}}_t = \mathbf{c} \tag{D.1b}$$

One of the possible solutions of (D.1) consists in using a fixed-point scheme in accelerations combined with a minimum norm solution of the system (if it involves redundant constraints).

## D.1 Direct sensitivity

Considering a set of $p$ parameters $\boldsymbol{\rho} \in \mathbb{R}^p$ conditioning the dynamic response of the system, the sensitivity analysis of the index-1 formulation can be obtained taking derivatives of (D.1) with respect to $\boldsymbol{\rho}$, according to [2]:

$$\mathbf{M}^d\ddot{\mathbf{z}}' + \mathbf{C}\dot{\mathbf{z}}' + \left(\mathbf{M}^d_{\hat{\mathbf{z}}}\ddot{\mathbf{z}} + \mathbf{K} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\lambda}\right)\mathbf{z}' + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\lambda}' = \mathbf{Q}^d_{\hat{\boldsymbol{\rho}}} - \mathbf{M}^d_{\hat{\boldsymbol{\rho}}}\ddot{\mathbf{z}} - \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}}\boldsymbol{\lambda} \tag{D.2a}$$

$$\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\ddot{\mathbf{z}}' = -2\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\dot{\mathbf{z}}' - \ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\mathbf{z}' - \ddot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}} \tag{D.2b}$$

Similarly to the dynamic solution proposed for the index-1 DAE, the sensitivities can be obtained using the same fixed point scheme with the sensitivities of the accelerations $\ddot{\mathbf{z}}'$ as main variables. The sensitivities of positions and velocities can be determined by means of a numerical integrator.

## D.2 Adjoint variable sensitivity

### D.2.1 Approach 1

The adjoint variable method has been previously applied to index-1 formulations in [2], but in the present section a slightly different approach is taken in order to avoid the computation of high order derivatives such as the second time derivative of the mass matrix $\ddot{\mathbf{M}}^d$ or the time derivative of the damping matrix $\dot{\mathbf{C}}$. These terms are avoided by means of the addition of a change of variables $\mathbf{v} = \dot{\mathbf{z}}$.

Let us consider the following Lagrangian:

$$\mathcal{L} = \boldsymbol{\psi} - \int_{t_0}^{t_F} \boldsymbol{\mu}_2^{\mathrm{T}} \left(\mathbf{M}\dot{\mathbf{v}} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \boldsymbol{\lambda} - \mathbf{Q}\right) \mathrm{d}t - \int_{t_0}^{t_F} \boldsymbol{\mu}_1^{\mathrm{T}} \left(\dot{\mathbf{z}} - \mathbf{v}\right) \mathrm{d}t - \int_{t_0}^{t_F} \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}} \left(\boldsymbol{\Phi}_{\hat{\mathbf{z}}} \dot{\mathbf{v}} - \mathbf{c}\right) \mathrm{d}t \tag{D.3}$$

Taking derivatives with respect to $\boldsymbol{\rho}$:

$$\begin{aligned} \mathcal{L}' = \boldsymbol{\psi}' - \int_{t_0}^{t_F} \boldsymbol{\mu}_1^{\mathrm{T}} \left(\dot{\mathbf{z}}' - \mathbf{v}'\right) \mathrm{d}t - \int_{t_0}^{t_F} \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}} \left(\boldsymbol{\Phi}_{\hat{\mathbf{z}}} \dot{\mathbf{v}}' + 2\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} \mathbf{v}' + \ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} \mathbf{z}' + \ddot{\boldsymbol{\Phi}}_{\hat{\rho}}\right) \mathrm{d}t \\ - \int_{t_0}^{t_F} \boldsymbol{\mu}_2^{\mathrm{T}} \left(\mathbf{M}^d \dot{\mathbf{v}}' + \mathbf{C}\mathbf{v}' + \left(\mathbf{M}_{\hat{\mathbf{z}}}^d \dot{\mathbf{v}} + \mathbf{K} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}} \boldsymbol{\lambda}\right) \mathbf{z}' + \boldsymbol{\Phi}_{\hat{\mathbf{z}}} \boldsymbol{\lambda}' - \mathbf{Q}_{\hat{\rho}}^d + \mathbf{M}_{\hat{\rho}}^d \ddot{\mathbf{z}} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\rho}}^{\mathrm{T}} \boldsymbol{\lambda}\right) \mathrm{d}t \end{aligned} \tag{D.4}$$

Regrouping terms:

$$\begin{aligned} \mathcal{L}' = &\int_{t_0}^{t_F} \left(\mathbf{g}_{\hat{\dot{\mathbf{z}}}} - \boldsymbol{\mu}_2^{\mathrm{T}} \mathbf{M}^d - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}} \boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right) \dot{\mathbf{v}}' \mathrm{d}t \\ &+ \int_{t_0}^{t_F} \left(\mathbf{g}_{\hat{\dot{\mathbf{z}}}} - \boldsymbol{\mu}_2^{\mathrm{T}} \mathbf{C} - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}} 2\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} + \boldsymbol{\mu}_1^{\mathrm{T}}\right) \mathbf{v}' \mathrm{d}t \\ &+ \int_{t_0}^{t_F} \left(-\boldsymbol{\mu}_1^{\mathrm{T}}\right) \dot{\mathbf{z}}' \mathrm{d}t \\ &+ \int_{t_0}^{t_F} \left(\mathbf{g}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}_2^{\mathrm{T}} \left(\mathbf{M}_{\hat{\mathbf{z}}}^d \dot{\mathbf{v}} + \mathbf{K} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}} \boldsymbol{\lambda}\right) - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}} \ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\right) \mathbf{z}' \mathrm{d}t \\ &+ \int_{t_0}^{t_F} \left(\mathbf{g}_{\boldsymbol{\lambda}} - \boldsymbol{\mu}_2^{\mathrm{T}} \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\right) \boldsymbol{\lambda}' \mathrm{d}t \\ &+ \int_{t_0}^{t_F} \left(\mathbf{g}_{\hat{\rho}} - \boldsymbol{\mu}_2^{\mathrm{T}} \left(\mathbf{M}_{\hat{\rho}}^d \ddot{\mathbf{z}} - \mathbf{Q}_{\hat{\rho}}^d + \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\rho}}^{\mathrm{T}} \boldsymbol{\lambda}\right) - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}} \ddot{\boldsymbol{\Phi}}_{\hat{\rho}}\right) \mathrm{d}t \end{aligned} \tag{D.5}$$

Assuming that $\dot{\mathbf{v}}$ is the temporal derivative of $\mathbf{v}$, and $\dot{\mathbf{z}}$ the time derivative of $\mathbf{z}$, an integration by parts can be applied to (D.5):

$$\begin{aligned}
\mathcal{L}' = &\left[\left(\mathbf{g}_{\hat{\dot{\mathbf{z}}}} - \boldsymbol{\mu}_2^{\mathrm{T}}\mathbf{M}^d - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right)\mathbf{v}'\right]_{t_0}^{t_F} \\
&- \int_{t_0}^{t_F}\left(\frac{\mathrm{d}\mathbf{g}_{\hat{\dot{\mathbf{z}}}}}{\mathrm{d}t} - \dot{\boldsymbol{\mu}}_2^{\mathrm{T}}\mathbf{M}^d - \boldsymbol{\mu}_2^{\mathrm{T}}\dot{\mathbf{M}}^d - \dot{\boldsymbol{\mu}}_{\boldsymbol{\Phi}}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\right)\mathbf{v}'\mathrm{d}t \\
&+ \int_{t_0}^{t_F}\left(\mathbf{g}_{\hat{\dot{\mathbf{z}}}} - \boldsymbol{\mu}_2^{\mathrm{T}}\mathbf{C} - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}2\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} + \boldsymbol{\mu}_1^{\mathrm{T}}\right)\mathbf{v}'\mathrm{d}t \\
&+ \left[\left(-\boldsymbol{\mu}_1^{\mathrm{T}}\right)\mathbf{z}'\right]_{t_0}^{t_F} \\
&- \int_{t_0}^{t_F}\left(-\dot{\boldsymbol{\mu}}_1^{\mathrm{T}}\right)\mathbf{z}'\mathrm{d}t \\
&+ \int_{t_0}^{t_F}\left(\mathbf{g}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}_2^{\mathrm{T}}\left(\mathbf{M}_{\hat{\mathbf{z}}}^d\dot{\mathbf{v}} + \mathbf{K} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\lambda}\right) - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\right)\mathbf{z}'\mathrm{d}t \\
&+ \int_{t_0}^{t_F}\left(\mathbf{g}_{\boldsymbol{\lambda}} - \boldsymbol{\mu}_2^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\right)\boldsymbol{\lambda}'\mathrm{d}t \\
&+ \int_{t_0}^{t_F}\left(\mathbf{g}_{\hat{\boldsymbol{\rho}}} - \boldsymbol{\mu}_2^{\mathrm{T}}\left(\mathbf{M}_{\hat{\boldsymbol{\rho}}}^d\ddot{\mathbf{z}} - \mathbf{Q}_{\hat{\boldsymbol{\rho}}}^d + \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}}\boldsymbol{\lambda}\right) - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\ddot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}}\right)\mathrm{d}t
\end{aligned} \tag{D.6}$$

Rearranging terms:

$$\begin{aligned}
\mathcal{L}' = &\left[\left(-\boldsymbol{\mu}_1^{\mathrm{T}}\right)\mathbf{z}'\right]_{t_0}^{t_F} + \left[\left(\mathbf{g}_{\hat{\dot{\mathbf{z}}}} - \boldsymbol{\mu}_2^{\mathrm{T}}\mathbf{M}^d - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right)\mathbf{v}'\right]_{t_0}^{t_F} \\
&+ \int_{t_0}^{t_F}\left(\mathbf{g}_{\hat{\dot{\mathbf{z}}}} - \frac{\mathrm{d}\mathbf{g}_{\hat{\ddot{\mathbf{z}}}}}{\mathrm{d}t} + \dot{\boldsymbol{\mu}}_2^{\mathrm{T}}\mathbf{M}^d + \boldsymbol{\mu}_2^{\mathrm{T}}\left(\dot{\mathbf{M}}^d - \mathbf{C}\right) + \dot{\boldsymbol{\mu}}_{\boldsymbol{\Phi}}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} + \boldsymbol{\mu}_1^{\mathrm{T}}\right)\mathbf{v}'\mathrm{d}t \\
&+ \int_{t_0}^{t_F}\left(\dot{\boldsymbol{\mu}}_1^{\mathrm{T}} + \mathbf{g}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}_2^{\mathrm{T}}\left(\mathbf{M}_{\hat{\mathbf{z}}}^d\dot{\mathbf{v}} + \mathbf{K} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\lambda}\right) - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\right)\mathbf{z}'\mathrm{d}t \\
&+ \int_{t_0}^{t_F}\left(\mathbf{g}_{\boldsymbol{\lambda}} - \boldsymbol{\mu}_2^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\right)\boldsymbol{\lambda}'\mathrm{d}t \\
&+ \int_{t_0}^{t_F}\left(\mathbf{g}_{\hat{\boldsymbol{\rho}}} - \boldsymbol{\mu}_2^{\mathrm{T}}\left(\mathbf{M}_{\hat{\boldsymbol{\rho}}}^d\ddot{\mathbf{z}} - \mathbf{Q}_{\hat{\boldsymbol{\rho}}}^d + \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}}\boldsymbol{\lambda}\right) - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\ddot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}}\right)\mathrm{d}t
\end{aligned} \tag{D.7}$$

The previous expressions are valid for any values of the adjoint variables, but the interest of this technique relies on the group of values that nullify the terms multiplying the sensitivities of the states, so their calculation can be avoided. Thus, the following set of adjoint equations is reached:

$$\mathbf{M}^{d\mathrm{T}}\dot{\boldsymbol{\mu}}_2 + \left(\dot{\mathbf{M}}^d - \mathbf{C}\right)^{\mathrm{T}}\boldsymbol{\mu}_2 + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\dot{\boldsymbol{\mu}}_{\boldsymbol{\Phi}} - \dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\mu}_{\boldsymbol{\Phi}} + \boldsymbol{\mu}_1 = -\mathbf{g}_{\hat{\dot{\mathbf{z}}}}^{\mathrm{T}} + \frac{\mathrm{d}\mathbf{g}_{\hat{\ddot{\mathbf{z}}}}^{\mathrm{T}}}{\mathrm{d}t} \tag{D.8a}$$

$$\dot{\boldsymbol{\mu}}_1 - \left(\mathbf{M}_{\hat{\mathbf{z}}}^d\dot{\mathbf{v}} + \mathbf{K} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\lambda}\right)^{\mathrm{T}}\boldsymbol{\mu}_2 - \ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\mu}_{\boldsymbol{\Phi}} = -\mathbf{g}_{\hat{\mathbf{z}}}^{\mathrm{T}} \tag{D.8b}$$

$$\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\boldsymbol{\mu}_2 = \mathbf{g}_{\boldsymbol{\lambda}}^{\mathrm{T}} \tag{D.8c}$$

$$\left[\boldsymbol{\mu}_1\right]^{t_F} = \mathbf{0} \tag{D.8d}$$

$$\left[\mathbf{M}^{d\mathrm{T}}\boldsymbol{\mu}_2 + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\mu}_{\boldsymbol{\Phi}}\right]^{t_F} = \left[\mathbf{g}_{\hat{\ddot{\mathbf{z}}}}^{\mathrm{T}}\right]^{t_F} \tag{D.8e}$$

The proposed solution for the initial conditions of the adjoint variables in $t_F$ is:

1. Initialize $\boldsymbol{\mu}_1$ by means of (D.8d).

2. Calculation of $\boldsymbol{\mu}_2$ and $\boldsymbol{\mu}_{\boldsymbol{\Phi}}$ from (D.8c) and (D.8e). These equations can be solved together using a minimum norm solution in order to handle redundant constraints, which is a requirement in the MBSLIM. One of the possible solutions could be the use of the Moore-Penrose generalized inverse, which allows the computation of the minimum norm solution of a system by means of a singular value decomposition, as presented in (5.92).

3. Determination of $\dot{\boldsymbol{\mu}}_1$ from (D.8b).

4. Evaluation of $\dot{\boldsymbol{\mu}}_2$ and $\dot{\boldsymbol{\mu}}_{\boldsymbol{\Phi}}$ from (D.8a) and the time derivative of (D.8c). In the following line, the resulting system is presented with the unknowns at the left side of the equal sign:

$$\mathbf{M}^{d\mathrm{T}}\dot{\boldsymbol{\mu}}_2 + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\dot{\boldsymbol{\mu}}_{\boldsymbol{\Phi}} = -\mathbf{g}_{\hat{\mathbf{z}}}^{\mathrm{T}} + \frac{\mathrm{d}\mathbf{g}_{\hat{\dot{\mathbf{z}}}}^{\mathrm{T}}}{\mathrm{d}t} - \left(\dot{\mathbf{M}}^d - \mathbf{C}\right)^{\mathrm{T}}\boldsymbol{\mu}_2 + \dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\mu}_{\boldsymbol{\Phi}} - \boldsymbol{\mu}_1 \tag{D.9a}$$

$$\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\dot{\boldsymbol{\mu}}_2 = \frac{\mathrm{d}\mathbf{g}_{\boldsymbol{\lambda}}^{\mathrm{T}}}{\mathrm{d}t} - \dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\boldsymbol{\mu}_2 \tag{D.9b}$$

The resulting system of equations is analog to the one described in step 2, and therefore the same scheme of solution can be applied.

The solution of the system composed of (D.8a), (D.8b) and (D.8c) for any time $t_i < t_F$ requires a backward integration in time to be computed.

The gradient of the objective function can be finally obtained using the values of the adjoint variables obtained for each time step and the partial derivatives of the dynamic terms with respect to the parameters:

$$\begin{aligned}\boldsymbol{\psi}' &= \left[\left(\boldsymbol{\mu}_1^{\mathrm{T}}\right)\mathbf{z}'\right]_{t_0} - \left[\left(\mathbf{g}_{\hat{\dot{\mathbf{z}}}} - \boldsymbol{\mu}_2^{\mathrm{T}}\mathbf{M}^d - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right)\dot{\mathbf{z}}'\right]_{t_0} \\ &+ \int_{t_0}^{t_F}\left(\mathbf{g}_{\hat{\boldsymbol{\rho}}} - \boldsymbol{\mu}_2^{\mathrm{T}}\left(\mathbf{M}_{\hat{\boldsymbol{\rho}}}^d\ddot{\mathbf{z}} - \mathbf{Q}_{\hat{\boldsymbol{\rho}}}^d + \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}}\boldsymbol{\lambda}\right) - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\ddot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}}\right)\mathrm{d}t\end{aligned} \tag{D.10}$$

It is convenient to remark that the computation of the sensitivities of the states in positions and velocities is required only once in the entire simulation, and exclusively at the initial point. The initial values of positions and velocities in the dynamic problem are usually computed by means of kinematic problems, so the assessment of the sensitivities of these problems are almost straightforward and do not involve dynamic terms such as derivatives of forces or masses.

### D.2.2 Approach 2

One of the biggest drawbacks of the computation of the adjoint system of an index-1 classical Lagrange formulation with the approach described in D.2.1 is related to the requirement of computation of the time derivative $\frac{\mathrm{d}\mathbf{g}_{\hat{\dot{\mathbf{z}}}}^{\mathrm{T}}}{\mathrm{d}t}$ which involve the jerks $\dddot{\mathbf{z}}$. This issue is specially important because it has to be computed at each time step.

In order avoid this derivative, the acceleration of the system can be explicitly expressed in terms of positions, velocities and Lagrange multipliers, as it is explained in [2].

Accordingly, (D.2), can be reformulated in a matrix form as:

$$\begin{bmatrix}\mathbf{M}^d & \mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \\ \mathbf{\Phi}_{\hat{\mathbf{z}}} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\ddot{\mathbf{z}}' \\ \boldsymbol{\lambda}'\end{bmatrix} = \begin{bmatrix}-\mathbf{M}_{\hat{\mathbf{z}}}^d\ddot{\mathbf{z}} - \mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\lambda} - \mathbf{K} \\ -\ddot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}\end{bmatrix}\mathbf{z}' + \begin{bmatrix}-\mathbf{C} \\ -2\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}\end{bmatrix}\dot{\mathbf{z}}' + \begin{bmatrix}\mathbf{Q}_{\hat{\boldsymbol{\rho}}}^d - \mathbf{M}_{\hat{\boldsymbol{\rho}}}^d\ddot{\mathbf{z}} - \mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}}\boldsymbol{\lambda} \\ -\ddot{\mathbf{\Phi}}_{\hat{\boldsymbol{\rho}}}\end{bmatrix} \tag{D.11}$$

The leading matrix is the same as the one of the index-1 formulation, with does not have an inverse if there are redundant constraints, but it has a generalized inverse in all the cases. Denoting the generalized inverse matrix with the superscript †, (D.11) can be rewritten as:

$$\begin{bmatrix}\ddot{\mathbf{z}}' \\ \boldsymbol{\lambda}'\end{bmatrix} = \begin{bmatrix}\mathbf{M}^d & \mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \\ \mathbf{\Phi}_{\hat{\mathbf{z}}} & \mathbf{0}\end{bmatrix}^{\dagger}\left(\begin{bmatrix}-\mathbf{M}_{\hat{\mathbf{z}}}^d\ddot{\mathbf{z}} - \mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\lambda} - \mathbf{K} \\ -\ddot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}\end{bmatrix}\mathbf{z}' + \begin{bmatrix}-\mathbf{C} \\ -2\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}\end{bmatrix}\dot{\mathbf{z}}' + \begin{bmatrix}\mathbf{Q}_{\hat{\boldsymbol{\rho}}}^d - \mathbf{M}_{\hat{\boldsymbol{\rho}}}^d\ddot{\mathbf{z}} - \mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}}\boldsymbol{\lambda} \\ -\ddot{\mathbf{\Phi}}_{\hat{\boldsymbol{\rho}}}\end{bmatrix}\right) \tag{D.12}$$

which can be reformulated as:

$$\begin{aligned}\ddot{\mathbf{z}}' &= \ddot{\mathbf{z}}_{\mathbf{z}}\mathbf{z}' + \ddot{\mathbf{z}}_{\dot{\mathbf{z}}}\dot{\mathbf{z}}' + \ddot{\mathbf{z}}_{\boldsymbol{\rho}} \\ \boldsymbol{\lambda}' &= \boldsymbol{\lambda}_{\mathbf{z}}\mathbf{z}' + \boldsymbol{\lambda}_{\dot{\mathbf{z}}}\dot{\mathbf{z}}' + \boldsymbol{\lambda}_{\boldsymbol{\rho}}\end{aligned} \tag{D.13}$$

with

$$\begin{bmatrix}\ddot{\mathbf{z}}_{\mathbf{z}} \\ \boldsymbol{\lambda}_{\mathbf{z}}\end{bmatrix} = \begin{bmatrix}\mathbf{M}^d & \mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \\ \mathbf{\Phi}_{\hat{\mathbf{z}}} & \mathbf{0}\end{bmatrix}^{\dagger}\begin{bmatrix}-\mathbf{M}_{\hat{\mathbf{z}}}^d\ddot{\mathbf{z}} - \mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\lambda} - \mathbf{K} \\ -\ddot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}\end{bmatrix} \tag{D.14a}$$

$$\begin{bmatrix}\ddot{\mathbf{z}}_{\dot{\mathbf{z}}} \\ \boldsymbol{\lambda}_{\dot{\mathbf{z}}}\end{bmatrix} = \begin{bmatrix}\mathbf{M}^d & \mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \\ \mathbf{\Phi}_{\hat{\mathbf{z}}} & \mathbf{0}\end{bmatrix}^{\dagger}\begin{bmatrix}-\mathbf{C} \\ -2\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}\end{bmatrix} \tag{D.14b}$$

$$\begin{bmatrix}\ddot{\mathbf{z}}_{\boldsymbol{\rho}} \\ \boldsymbol{\lambda}_{\boldsymbol{\rho}}\end{bmatrix} = \begin{bmatrix}\mathbf{M}^d & \mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}} \\ \mathbf{\Phi}_{\hat{\mathbf{z}}} & \mathbf{0}\end{bmatrix}^{\dagger}\begin{bmatrix}\mathbf{Q}_{\hat{\boldsymbol{\rho}}}^d - \mathbf{M}_{\hat{\boldsymbol{\rho}}}^d\ddot{\mathbf{z}} - \mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}}\boldsymbol{\lambda} \\ -\ddot{\mathbf{\Phi}}_{\hat{\boldsymbol{\rho}}}\end{bmatrix} \tag{D.14c}$$

Now, the gradient of the Lagrangian (D.3) can be expressed as:

$$\begin{aligned}\mathcal{L}' = &\int_{t_0}^{t_F}\left(\mathbf{g}_{\hat{\dot{\mathbf{z}}}} - \boldsymbol{\mu}_2^{\mathrm{T}}\mathbf{M}^d - \boldsymbol{\mu}_{\mathbf{\Phi}}^{\mathrm{T}}\mathbf{\Phi}_{\hat{\mathbf{z}}}\right)\dot{\mathbf{v}}'\mathrm{d}t \\ &+\int_{t_0}^{t_F}\left(\mathbf{g}_{\hat{\dot{\mathbf{z}}}} - \boldsymbol{\mu}_2^{\mathrm{T}}\mathbf{C} - \boldsymbol{\mu}_{\mathbf{\Phi}}^{\mathrm{T}}2\dot{\mathbf{\Phi}}_{\hat{\mathbf{z}}} + \boldsymbol{\mu}_1^{\mathrm{T}}\right)\mathbf{v}'\mathrm{d}t \\ &+\int_{t_0}^{t_F}\left(-\boldsymbol{\mu}_1^{\mathrm{T}}\right)\dot{\mathbf{z}}'\mathrm{d}t \\ &+\int_{t_0}^{t_F}\left(\mathbf{g}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}_2^{\mathrm{T}}\left(\mathbf{M}_{\hat{\mathbf{z}}}^d\dot{\mathbf{v}} + \mathbf{K} + \mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\lambda}\right) - \boldsymbol{\mu}_{\mathbf{\Phi}}^{\mathrm{T}}\ddot{\mathbf{\Phi}}_{\hat{\mathbf{z}}}\right)\mathbf{z}'\mathrm{d}t \\ &+\int_{t_0}^{t_F}\left(\mathbf{g}_{\boldsymbol{\lambda}} - \boldsymbol{\mu}_2^{\mathrm{T}}\mathbf{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\right)\boldsymbol{\lambda}'\mathrm{d}t \\ &+\int_{t_0}^{t_F}\left(\mathbf{g}_{\hat{\boldsymbol{\rho}}} - \boldsymbol{\mu}_2^{\mathrm{T}}\left(\mathbf{M}_{\hat{\boldsymbol{\rho}}}^d\ddot{\mathbf{z}} - \mathbf{Q}_{\hat{\boldsymbol{\rho}}}^d + \mathbf{\Phi}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}^{\mathrm{T}}\boldsymbol{\lambda}\right) - \boldsymbol{\mu}_{\mathbf{\Phi}}^{\mathrm{T}}\ddot{\mathbf{\Phi}}_{\hat{\boldsymbol{\rho}}}\right)\mathrm{d}t\end{aligned} \tag{D.15}$$

The objective of this new approach is to eliminate the time derivatives of the objective function, so equations (D.13) are substituted exclusively in the terms involving any derivative of $\mathbf{g}$. Thus, substituting (D.13) in (D.16) (remember that $\dot{\mathbf{v}}' = \ddot{\mathbf{z}}'$):

$$\begin{aligned}
\mathcal{L}' = \int_{t_0}^{t_F} \left(-\boldsymbol{\mu}_2^{\mathrm{T}}\mathbf{M}^d - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right)\dot{\mathbf{v}}'\mathrm{d}t \\
+ \int_{t_0}^{t_F} \left(\mathbf{g}_{\hat{\dot{\mathbf{z}}}} + \mathbf{g}_{\hat{\ddot{\mathbf{z}}}}\ddot{\mathbf{z}}_{\dot{\mathbf{z}}} + \mathbf{g}_{\boldsymbol{\lambda}}\boldsymbol{\lambda}_{\dot{\mathbf{z}}} - \boldsymbol{\mu}_2^{\mathrm{T}}\mathbf{C} - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}2\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} + \boldsymbol{\mu}_1^{\mathrm{T}}\right)\mathbf{v}'\mathrm{d}t \\
+ \int_{t_0}^{t_F} \left(-\boldsymbol{\mu}_1^{\mathrm{T}}\right)\dot{\mathbf{z}}'\mathrm{d}t \\
+ \int_{t_0}^{t_F} \left(\mathbf{g}_{\hat{\mathbf{z}}} + \mathbf{g}_{\hat{\ddot{\mathbf{z}}}}\ddot{\mathbf{z}}_{\mathbf{z}} + \mathbf{g}_{\boldsymbol{\lambda}}\boldsymbol{\lambda}_{\mathbf{z}} - \boldsymbol{\mu}_2^{\mathrm{T}}\left(\mathbf{M}^d_{\hat{\mathbf{z}}}\dot{\mathbf{v}} + \mathbf{K} + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}\boldsymbol{\lambda}\right) - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\right)\mathbf{z}'\mathrm{d}t \\
+ \int_{t_0}^{t_F} \left(-\boldsymbol{\mu}_2^{\mathrm{T}}\boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\right)\boldsymbol{\lambda}'\mathrm{d}t \\
+ \int_{t_0}^{t_F} \left(\mathbf{g}_{\hat{\boldsymbol{\rho}}} + \mathbf{g}_{\hat{\ddot{\mathbf{z}}}}\ddot{\mathbf{z}}_{\boldsymbol{\rho}} + \mathbf{g}_{\boldsymbol{\lambda}}\boldsymbol{\lambda}_{\boldsymbol{\rho}} - \boldsymbol{\mu}_2^{\mathrm{T}}\left(\mathbf{M}^d_{\hat{\boldsymbol{\rho}}}\ddot{\mathbf{z}} - \mathbf{Q}^d_{\hat{\boldsymbol{\rho}}} + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}\boldsymbol{\lambda}\right) - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\ddot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}}\right)\mathrm{d}t
\end{aligned} \tag{D.16}$$

Now, integrating by parts:

$$\begin{aligned}
\mathcal{L}' = \left[\left(-\boldsymbol{\mu}_2^{\mathrm{T}}\mathbf{M}^d - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\right)\mathbf{v}'\right]_{t_0}^{t_F} \\
- \int_{t_0}^{t_F} \left(-\dot{\boldsymbol{\mu}}_2^{\mathrm{T}}\mathbf{M}^d - \boldsymbol{\mu}_2^{\mathrm{T}}\dot{\mathbf{M}}^d - \dot{\boldsymbol{\mu}}_{\boldsymbol{\Phi}}^{\mathrm{T}}\boldsymbol{\Phi}_{\hat{\mathbf{z}}} - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\right)\mathbf{v}'\mathrm{d}t \\
+ \int_{t_0}^{t_F} \left(\mathbf{g}_{\hat{\dot{\mathbf{z}}}} + \mathbf{g}_{\hat{\ddot{\mathbf{z}}}}\ddot{\mathbf{z}}_{\dot{\mathbf{z}}} + \mathbf{g}_{\boldsymbol{\lambda}}\boldsymbol{\lambda}_{\dot{\mathbf{z}}} - \boldsymbol{\mu}_2^{\mathrm{T}}\mathbf{C} - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}2\dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}} + \boldsymbol{\mu}_1^{\mathrm{T}}\right)\mathbf{v}'\mathrm{d}t \\
+ \left[\left(-\boldsymbol{\mu}_1^{\mathrm{T}}\right)\mathbf{z}'\right]_{t_0}^{t_F} - \int_{t_0}^{t_F} \left(-\dot{\boldsymbol{\mu}}_1^{\mathrm{T}}\right)\mathbf{z}'\mathrm{d}t \\
+ \int_{t_0}^{t_F} \left(\mathbf{g}_{\hat{\mathbf{z}}} + \mathbf{g}_{\hat{\ddot{\mathbf{z}}}}\ddot{\mathbf{z}}_{\mathbf{z}} + \mathbf{g}_{\boldsymbol{\lambda}}\boldsymbol{\lambda}_{\mathbf{z}} - \boldsymbol{\mu}_2^{\mathrm{T}}\left(\mathbf{M}^d_{\hat{\mathbf{z}}}\dot{\mathbf{v}} + \mathbf{K} + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}\boldsymbol{\lambda}\right) - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}\right)\mathbf{z}'\mathrm{d}t \\
+ \int_{t_0}^{t_F} \left(-\boldsymbol{\mu}_2^{\mathrm{T}}\boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}}\right)\boldsymbol{\lambda}'\mathrm{d}t \\
+ \int_{t_0}^{t_F} \left(\mathbf{g}_{\hat{\boldsymbol{\rho}}} - \boldsymbol{\mu}_2^{\mathrm{T}}\left(\mathbf{M}^d_{\hat{\boldsymbol{\rho}}}\ddot{\mathbf{z}} - \mathbf{Q}^d_{\hat{\boldsymbol{\rho}}} + \boldsymbol{\Phi}^{\mathrm{T}}_{\hat{\mathbf{z}}\hat{\boldsymbol{\rho}}}\boldsymbol{\lambda}\right) - \boldsymbol{\mu}_{\boldsymbol{\Phi}}^{\mathrm{T}}\ddot{\boldsymbol{\Phi}}_{\hat{\boldsymbol{\rho}}}\right)\mathrm{d}t
\end{aligned} \tag{D.17}$$

Rearranging terms and making null the expressions multiplying the sensitivities of the states and the Lagrange multipliers, the following set of adjoint equations is reached:

$$\mathbf{M}^{d\mathrm{T}}\dot{\boldsymbol{\mu}}_2 + \left(\dot{\mathbf{M}}^d - \mathbf{C}\right)^{\mathrm{T}}\boldsymbol{\mu}_2 + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\dot{\boldsymbol{\mu}}_{\boldsymbol{\Phi}} - \dot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\mu}_{\boldsymbol{\Phi}} + \boldsymbol{\mu}_1 = -\left(\mathbf{g}_{\hat{\dot{\mathbf{z}}}}^{\mathrm{T}} + \mathbf{g}_{\hat{\ddot{\mathbf{z}}}}\ddot{\mathbf{z}}_{\dot{\mathbf{z}}} + \mathbf{g}_{\boldsymbol{\lambda}}\boldsymbol{\lambda}_{\dot{\mathbf{z}}}\right)^{\mathrm{T}} \tag{D.18a}$$

$$\dot{\boldsymbol{\mu}}_1 - \left(\mathbf{M}_{\hat{\mathbf{z}}}^d\dot{\mathbf{v}} + \mathbf{K} + \boldsymbol{\Phi}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\lambda}\right)^{\mathrm{T}}\boldsymbol{\mu}_2 - \ddot{\boldsymbol{\Phi}}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\mu}_{\boldsymbol{\Phi}} = -\left(\mathbf{g}_{\hat{\mathbf{z}}}^{\mathrm{T}} + \mathbf{g}_{\hat{\ddot{\mathbf{z}}}}\ddot{\mathbf{z}}_{\mathbf{z}} + \mathbf{g}_{\boldsymbol{\lambda}}\boldsymbol{\lambda}_{\mathbf{z}}\right)^{\mathrm{T}} \tag{D.18b}$$

$$\boldsymbol{\Phi}_{\hat{\mathbf{z}}}\boldsymbol{\mu}_2 = \mathbf{0} \tag{D.18c}$$

$$\left[\boldsymbol{\mu}_1\right]^{t_F} = \mathbf{0} \tag{D.18d}$$

$$\left[\mathbf{M}^{d\mathrm{T}}\boldsymbol{\mu}_2 + \boldsymbol{\Phi}_{\hat{\mathbf{z}}}^{\mathrm{T}}\boldsymbol{\mu}_{\boldsymbol{\Phi}}\right]^{t_F} = \left[\mathbf{g}_{\hat{\dot{\mathbf{z}}}}^{\mathrm{T}}\right]^{t_F} \tag{D.18e}$$

Observe that the resulting adjoint system of equations generated by this approach and the one presented in D.2.1 are equivalent and can be solved with the same scheme of computation. The main advantage of the new set of equations obtained relies on the avoidance of high order temporal derivatives of the objective function, being the computation of new terms such as $\ddot{\mathbf{z}}_{\mathbf{z}}$ or $\boldsymbol{\lambda}_{\mathbf{z}}$ its more remarkable disadvantage.

# Appendix E

# List of publications

This thesis has been funded by MINECO by means of the doctoral research contract BES-2017-080727, co-financed by the European Union through the ESF program and associated to the project DPI2016-81005-P.

The work developed during the present thesis has been exposed in different conference presentations, and has lead to one conference paper and one journal paper. It should be remarked that there are two more journal papers under development related to some topics discussed in chapter 5. The papers and journal communications are listed below:

**Journal papers**

D. Dopico Dopico, A. López Varela, A. Luaces Fernández. Augmented Lagrangian index-3 semi-recursive formulations with projections. *Multibody System Dynamics*, 2020. doi: 10.1007/s11044-020-09771-9.

**Journal papers under development**

Discrete adjoint variable method for the sensitivity analysis of ALI3-P formulations.

Direct sensitivity analysis of semi-recursive ALI3-P formulations.

**Conference papers**

D. Dopico Dopico, A. López Varela, A. Luaces Fernández. Two general index-3 semi-recursive formulations for the dynamics of multibody systems. In A. Kecskeméthy, F. Geu Flores, editors, *Multibody Dynamics 2019*, 401–408. Springer International Publishing, 2019.

**Conference communications**

A. López Varela, D. Dopico Dopico, A. Luaces Fernández. Direct sensitivity analysis of multibody systems modeled with relative coordinates using an augmented Lagrangian formulation with projections. In *Proceeding of the ASME 2020 International*

*Design Engineering Technical Conferences and Computers and Information in Engineering Conference - 16th International Conference on Multibody Systems, Nonlinear Dynamics and Control.* Saint Louis, USA (virtual conference), 2020.

A. López Varela, A. Luaces Fernández, D. Dopico Dopico. Adjoint sensitivity analysis of multibody systems modeled with joint coordinates using an augmented Lagrangian formulation with projections. In *Proceeding of the ASME 2021 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference - 17th International Conference on Multibody Systems, Nonlinear Dynamics and Control.* Saint Louis, USA (virtual conference), 2021.

A. López Varela, A. Luaces Fernández, D. Dopico Dopico. Discrete adjoint approach for the sensitivity analysis of an augmented Lagrangian index-3 formulation with projections. In *10th ECCOMAS Thematic Conference on Multibody Dynamics.* Budapest, Hungary (virtual conference), 2021.

D. Dopico Dopico, A. López Varela, A. Luaces Fernández. Optimization of a three wheeled tilting vehicle. In *10th ECCOMAS Thematic Conference on Multibody Dynamics.* Budapest, Hungary (virtual conference), 2021.

D. Dopico Dopico, A. López Varela, A. Luaces Fernández. Steering optimal design of a three wheeled tilting vehicle. In *1st International Conference on Machine Design.* Porto, Portugal, 2021.

D. Dopico Dopico, A. López Varela, A. Luaces Fernández, E. Sanjurjo Maroño. Kinematic and dynamic optimization of the steering of a tilting tricycle. In *IUTAM Symposium on Optimal Design and Control of Multibody Systems.* Hamburg, Germany, 2022.

A. López Varela, D. Dopico Dopico, A. Luaces Fernández. Discrete adjoint variable method applied to semi-recursive augmented Lagrangian index-3 formulations with projections. In *IUTAM Symposium on Optimal Design and Control of Multibody Systems.* Hamburg, Germany, 2022.

**Submitted conference communications**

A. López Varela, D. Dopico Dopico, A. Luaces Fernández. Sensitivity analysis of semi-recursive Augmented Lagrangian formulations with projections. In *6th Joint International Conference on Multibody System Dynamics and 10th Asian Conference on Multibody Dynamics.* New Delhi, India, 2022.

A. Luaces Fernández, A. López Varela, A. Verulkar, C. Sandu, A. Sandu, D. Dopico Dopico. Optimal design and control of the steering of a tilting tricycle. In *6th Joint International Conference on Multibody System Dynamics and 10th Asian Conference on Multibody Dynamics.* New Delhi, India, 2022.

# Appendix F

# Resumo estendido

A análise de sensibilidade é unha ferramenta extraordinariamente útil na optimización de sistemas multicorpo, tanto a nivel de deseño como a nivel de operación ou control. A sensibilidade dunha determinada función con respecto duns parámetros ofrece unha medida do efecto ou impacto deses parámetros na función, o cal permite obter, entre outras cousas, unha medida de avance dun optimizador baseado en gradiente.

Existen múltiples métodos de diferenciación para acadar a sensibilidade dun determinado sistema de ecuacións. A opción máis sinxela e inmediata abrangue os métodos numéricos de diferencias finitas, que se fundamentan na perturbación da función obxectivo. Estes métodos, aínda que simples, teñen un problema intrínseco; o chamado "dilema de tamaño de paso". Para perturbacións moi grandes, a aproximación da derivada mediante diferencias finitas incorpora grandes erros, mentres que perturbacións moi reducidas engaden erros numéricos relativos á resta de magnitudes moi semellantes. Este inconveniente pode ser evadido mediante o uso do método de diferencias finitas en variable complexa, de forma que é posible empregar perturbacións extremadamente pequenas sen incorporar erros á derivada. Sen embargo, este último método implica ampliar calquera cálculo para soportar variables complexas.

Outro método de diferenciación en auxe hoxe en día é a diferenciación automática. Esta técnica baséase na utilización de librerías e ferramentas matemáticas que son quen de calcular a derivada de calquera función a través das derivadas analíticas das operacións elementais que a compoñen mediante a aplicación da regra da cadea. O único que require esta técnica é unha definición particular de cada unha das funcións involucradas no cálculo xeral da función obxectivo, de forma que as librerías externas saiban interpretar os cálculos requiridos pola función principal. O aumento de librerías ou paquetes software dedicados a este tipo de tarefas fixo que nos últimos anos moitos investigadores optasen por esta opción nos seus cálculos de sensibilidade. A diferenciación automática permite unha sensibilidade con alta precisión cun reducido coste a nivel de implementación, pero existen outros métodos que seguen sendo os máis eficientes a nivel de tempo de computación: os métodos analíticos.

A diferenciación analítica fundaméntase no cálculo e ensamblaxe das expresións analíticas de cada dependencia da función obxectivo. Esta técnica require un enorme esforzo de implementación, pero como contrapartida tamén soe conducir ós cálculos máis eficientes, xa que o estudo analítico das derivadas permite simplificar cálculos e evitar repeticións.

Dentro das técnicas para a análise de sensibilidade, pódense distinguir métodos de diferenciación directa e métodos de variable adxunta. Os primeiros xeran un problema expresado en función das sensibilidades das variables do problema orixinal, mentres que o método adxunto permite reformular a análise de sensibilidade para evitar ter que calcular esas sensibilidades das variables primarias, que poden ser computacionalmente moi custosas para un elevado número de parámetros.

Volvendo a sistemas multicorpo, a aplicación de técnicas de análise de sensibilidade pode facerse a nivel de cinemática ou de dinámica. A cinemática de calquera sistema multicorpo ven determinada por unha serie de ecuacións alxébricas a nivel de relacións recursivas ou a nivel de restricións. A dinámica, sen embargo, ven descrita por ecuacións diferenciais alxébricas (máis coñecidas polas súas siglas en inglés DAE ou differential algebraic equations), e a súa análise de sensibilidade é máis complexa, especialmente se se empregan métodos analíticos. No presente traballo estúdase a aplicación de técnicas de sensibilidade analíticas á dinámica de sistemas multicorpo modelada mediante coordenadas de par ou relativas.

En xeral, pódense distinguir tres tipos de coordenadas amplamente despregadas na comunidade multicorpo: coordenadas de punto de referencia, coordenadas naturais (ou completamente Cartesianas), e coordenadas relativas. As coordenadas de punto de referencia constitúen quizais unha das formas máis directas e sistemáticas de modelar un sistema multicorpo. Neste caso, o movemento de cada corpo ven descrito pola posición e orientación dun punto, que frecuentemente acostuma a coincidir co centro de masas. Un dos problemas deste tipo de coordenadas reside na imposibilidade de definir unha orientación mediante tres parámetros sen singularidades. Moitos autores describen a orientación mediante tres variables pero empregan métodos para manexar esas configuracións singulares. Outros empregan conxuntos de parámetros redundantes, sendo o máis destacado o conxunto de parámetros de Euler, que consiste en catro parámetros condicionados por unha restrición de norma unitaria. A maiores, tamén existen novos estudos fundamentados na teoría de grupos de Lie, que permiten utilizar parametrizacións locais de forma que non se acadan configuracións singulares dentro dun paso de tempo.

O segundo método de modelado consiste no emprego de coordenadas puramente Cartesianas, é dicir, coordenadas de puntos e vectores. Con esta definición supérase o problema de parametrización das orientacións das coordenadas de punto de referencia, pero esta non é a única vantaxe. Aínda que para a definición do mecanismo é preciso empregar un elevado número de puntos e vectores (12 coordenadas cartesianas para un sólido 3D), as relacións cinemáticas de compartición de puntos e vectores poden ser establecidas de forma automática e as restricións resultantes son, como moito, cuadráticas. Este tipo de coordenadas producen sistemas de ecuacións moi dispersos, pero mediante o uso de solvers e álxebra dispersa pódense conseguir solucións moi eficientes da dinámica multicorpo.

A terceira das ramas do modelado en sistemas multicorpo consiste no uso das coordenadas dos pares cinemáticos como variables do modelo. Este sistema constitúe a maneira máis natural de modelar un sistema multicorpo, especialmente en cadeas abertas. Neste tipo de modelos, os graos de liberdade son (habitualmente) directamente identificables coas coordenadas de par en cadeas abertas, o cal implica que se consegue unha parametrización do modelo en coordenadas mínimas. En cadeas pechadas, esta identificación de coordenadas de par e graos de liberdade xa non se cumpre, e é preciso eliminar un dos pares para converter a cadea pechada nunha cadea aberta. Para manter a topoloxía de cadea pechada, é necesario introducir unha restrición cinemática que garanta o movemento relativo do par eliminado. Deste xeito, unha cadea pechada pode expresarse en termos dunha cadea aberta suxeita a unha serie de restricións cinemáticas. Os modelos de coordenadas relativas habitúan a xerar modelos reducidos que é posible resolver de forma moi eficiente. Porén, non é posible determinar que estes modelos xeren sempre as solucións mais rápidas, senón que a eficiencia de cada modelización depende do mecanismo e da manobra executada. Por este motivo, é extremadamente útil ter dispoñible nunha mesma librería multicorpo a posibilidade de resolver sistemas mecánicos a través de diferentes modelos.

Neste respecto, no Laboratorio de Ingeniería Mecánica da Universidade da Coruña desenvolveuse unha librería de propósito xeral para a simulación da cinemática, dinámica e sensibilidade (entre outros problemas) de modelos multicorpo chamada MBSLIM. Esta librería foi orixinalmente deseñada en coordenadas naturais debido ás súas grandes vantaxes en canto á definición de modelos, propiedades dinámicas (matriz de masas constante), ou simplicidade das restricións xeradas. Recentemente xurdiu a posibilidade de ampliar esta librería para coordenadas relativas, de maneira que fose posible resolver determinados modelos de forma máis eficiente. Parte do traballo desenvolto durante a presente tese consistiu na implementación destes modelos na librería multicorpo MBSLIM, tanto a nivel de creación de modelo como cinemática, dinámica e sensibilidade. Cabe destacar que esta programación supuxo a maior parte dos esforzos desta tese para poder obter un código xeral, robusto e eficiente.

No capítulo 2 da tese, preséntase unha revisión das relacións cinemáticas para sete dos tipos de pares máis habituais no modelado de sistemas multicorpo en 3 dimensións. Cada par estúdase en termos de cinemática a nivel de posición, e tamén se xeran unha serie de relacións recursivas a nivel de velocidade que suporán a base dos métodos de cálculo tanto cinemático como dinámico. As relacións recursivas expresan a velocidade e aceleración lineal dun punto de referencia dun corpo e a velocidade e aceleración angulares dese corpo en relación coas mesmas magnitudes do corpo anterior.

En tanto as magnitudes lineais son relativas a un determinado punto de referencia, as relacións recursivas de cada par involucrarán o punto de referencia usado, o cal implica que se obterán expresións diferentes para cada selección. Por este motivo, desenvolvéronse unha serie de expresións xerais válidas para calquera punto de referencia, e despois particularizáronse para dous conxuntos de puntos de referencia amplamente empregados en métodos recursivos, que se corresponden cos centro de masa de cada corpo (RTdyn0) ou ben cos puntos coincidentes coa orixe global de coordenadas en cada instante (RTdyn1).

A cinemática dos modelos de coordenadas relativas de cadea aberta pode ser directamente calculada gracias ás relacións cinemáticas e recursivas de cada tipo de par, pero a dinámica precisa máis pasos. En primeiro lugar, é necesario determinar que tipo de método se pretende utilizar para resolver a dinámica, xa que métodos semi-recursivos ou métodos completamente recursivos requiren pasos e ensamblaxes completamente diferentes. En ambos métodos, o primeiro paso consiste en determinar a matriz de masas e o vector de forzas xeneralizado para cada corpo expresado en función do punto de referencia seleccionado. O seguinte paso consiste na acumulación, que en métodos semi-recursivos leva á construción dunha matriz de masas cadrada do tamaño do vector de coordenadas relativas e a un vector de forzas coa mesma magnitude. O cálculo totalmente recursivo é diferente a este respecto, xa que non require ensamblar grandes matrices ou vectores, senón acumular matrices de masas e vectores de forzas xeneralizados entre corpos consecutivos partindo dos extremos da árbore cinemática ata a base do mecanismo.

No capítulo 3, as expresións de cadea aberta desenvoltas no capítulo 2 son combinadas con restricións para poder resolver problemas de cadea pechada. Os clásicos problemas cinemáticos de posición inicial, desprazamentos finitos, velocidade e aceleración son particularizados para o caso de coordenadas relativas engadindo unha nova característica non recollida polo momento en ningún outro dos textos revisados polo autor: a posibilidade de empregar graos de liberdade sen estar contidos no vector de coordenadas dependentes. Esta propiedade é retomada en formulacións dinámicas en coordenadas independentes tamén estudadas nesta tese.

A primeira das formulacións dinámicas estudada e implementada na librería consiste na formulación Matriz R semi-recursiva, que recolle os desenvolvementos de cadea aberta do capítulo 2 e os combina cun esquema de proxección das coordenadas relativas no manifold de coordenadas independentes. Este proceso require a resolución dos problemas cinemáticos de posición e velocidade unha vez por iteración, o cal pode supoñer un grande esforzo computacional. Outra posible imposición das restricións en sistemas de cadea pechada consiste no uso dun esquema de Lagrangiano aumentado de índice 3 con proxeccións (máis coñecido polas súas siglas en inglés ALI3-P ou Augmented Lagrangian Index-3 formulation with projections). Este esquema combina a imposición do vector de restricións a nivel de posición cunha proxección en velocidades e outra en aceleracións para garantir un determinado nivel de cumprimento das restricións a niveles de velocidade e aceleración. Este é un esquema: moi robusto, xa que permite a definición de restricións redundantes ou matrices de masas singulares (con coordenadas sen masa asociada); é preciso, gracias ó bo cumprimento de restricións a niveles de posición, velocidade e aceleración; e é moi eficiente, gracias á resolución mediante un esquema de Lagrangiano aumentado e grazas á resolución de proxeccións coa mesma matriz de proxección, que permite factorizar a matriz do sistema só unha vez por paso de tempo.

A maiores, tamén se estudou a aplicación de restricións a formulacións totalmente recursivas, pero debido á súa elevada complexidade e ó seu comportamento francamente pouco robusto, decidiuse non incluír estes desenvolvementos no presente documento de tese. Aínda así, o traballo con estas formulacións non foi en van, xa que foi posible empregar acumulacións totalmente recursivas no cálculo de sensibilidades semi-recursivas.

A contribución máis importante do presente traballo atópase no capítulo 4, dedicado ó cálculo analítico de tódolos termos involucrados na análise de sensibilidade de formulacións recursivas de cadea aberta. As formulacións recursivas, tanto semi-recursivas como completamente recursivas, carrexan unha serie de inconvenientes que fixeron que ata o momento non se estudasen de forma detallada a nivel de sensibilidade analítica. O maior problema é relativo á cantidade de operacións intermedias necesarias para obter a dinámica, incluíndo cálculo de relacións recursivas, ensamblaxe de termos a nivel de sólido, etc. É obvio que unha longa concatenación de produtos e operacións na dinámica implica unha explosión a nivel de sensibilidade, que pode facer que a análise de sensibilidade sexa altamente ineficiente. No capítulo 4 preséntanse tódalas ecuacións e relacións necesarias para poder executar unha análise de sensibilidade de sistemas de cadea aberta modelados en coordenadas relativas. Tódalas expresións finais foron estudadas para mellorar a eficiencia e para simplificar o proceso de implementación. Cabe destacar que tódalas expresións foron orixinalmente referidas a un conxunto de puntos de referencia arbitrario e logo foron particularizadas para os dous conxuntos máis usados, é dicir, para RTdyn0 e RTdyn1.

Un dos desenvolvementos máis destacables presentados nesta tese consiste no cálculo e ensamblaxe eficiente das derivadas relativas a puntos e vectores. Mediante unha simplificación de termos acadouse unha definición de derivadas que non depende do punto de referencia seleccionado, o cal unifica as expresións de RTdyn0 e RTdyn1, facilitando o proceso de implementación. A maiores, tamén se presentan nesta tese os cálculos precisos para obter as derivadas parciais con respecto de calquera tipo de parámetro, facendo especial fincapé no caso de coordenadas locais de puntos que afectan á topoloxía do sistema.

No capítulo 5, a sensibilidade dos problemas cinemáticos de posición, velocidade e aceleración son estudados empregando tanto o método de diferenciación directa como o método de variable adxunta. Estes cálculos, ademais de ter interese en sí, tamén se inclúen como parte dos desenvolvementos requiridos na sensibilidade de formulacións dinámicas con restricións.

A sensibilidade de Matriz R semi-recursiva tamén é estudada mediante os métodos de diferenciación directa e variable adxunta continua, prestando especial atención á posibilidade de empregar como coordenadas independentes un conxunto de variables que non estean incluídas no vector de coordenadas relativas dependentes. Tanto a aplicación do método de diferenciación directa como adxunta están fundamentados en traballos previos, pero neste caso amplíase a xeneralidade dos métodos da bibliografía mediante a posibilidade de ter coordenadas independentes fora do vector de coordenadas dependentes.

A análise de sensibilidade da formulación ALI3-P semi-recursiva é significativamente máis complexa que a de Matriz R, en parte debido ás iteracións dos multiplicadores de Lagrange, á aplicación de ecuacións alxébricas adicionais (proxeccións) e á definición das ecuacións do movemento mediante ecuacións diferenciais alxébricas de índice 3. Se ben a aplicación do método de diferenciación directa é máis ou menos inmediato, o proceso para a obtención das ecuacións de variable adxunta é realmente intricado. Por este motivo se desenvolveron dous esquemas en paralelo, un empregando o método continuo de variable adxunta e outro o discreto.

No método continuo de variable adxunta considéranse as ecuacións do movemento como expresións continuas, e derívanse baixo esta asunción aplicando integración por partes no tempo. Isto implica que se xeran unha serie de condicións no tempo final da simulación dinámica que determinan a inicialización das variables adxuntas, convertendo este cálculo de valores iniciais nun proceso realmente complexo. Adicionalmente, a integración por partes implica a aparición de novas derivadas temporais que non son necesarias na dinámica e poden ser computacionalmente custosas. No método discreto, pola contra, o Lagrangiano adxunto constrúese a partir das ecuacións da dinámica discretizada, o cal implica que as ecuacións do integrador empregado na dinámica determinarán as expresións adxuntas finais. A pesar deste contratempo, o proceso de inicialización das variables adxuntas é inmediato e non son precisas máis derivadas que as requiridas no método de diferenciación directa.

A aplicación de ambos métodos á formulación ALI3-P semi-recursiva levouse a cabo de forma satisfactoria, se ben se comprobou empiricamente que o método continuo ten xeralmente un peor comportamento que o discreto tanto a nivel de precisión como a nivel de coste computacional.

No capítulo 6 descríbense brevemente os esforzos de implementación necesarios para programar os métodos introducidos na presente tese na librería MBSLIM. Non se pretende que este capítulo sexa unha guía de programación nin un listado detallado dos procedementos de creación de modelos, composición de sistemas e resolución de ecuacións, senón unha descrición das liñas mestras seguidas durante o proceso de implementación. Tamén se inclúen neste capítulo desenvolvementos intermedios para maximizar a eficiencia dos métodos dinámicos e de sensibilidade, e coméntase o proceso seguido para a validación da implementación de cada expresión dinámica e de sensibilidade.

O capítulo 7 manifesta a xeneralidade, corrección, exactitude e eficiencia dos métodos dinámicos e de análise de sensibilidade descritos ó longo do documento mediante a avaliación do seu desempeño en experimentos numéricos con cinco mecanismos: un mecanismo de cinco barras, un biela manivela espacial, un vehículo tipo buggy, unha bicicleta e un sistema composto por unha cadea e unha áncora. Os dous primeiros mecanismos son exemplos tradicionalmente usados como banco de probas que teñen solucións dinámicas coñecidas, e utilízanse para comprobar a exactitude e eficiencia tanto das formulacións dinámicas semi-recursivas como das formulacións de sensibilidade semi-recursiva. No terceiro e cuarto modelo próbanse as formulacións dinámicas e de sensibilidade en exemplos máis complexos, e inclúen a aplicación dos cálculos de sensibilidade na resolución dun problema de deseño óptimo e doutro problema de control óptimo. A optimización de mecanismos é un dos fins últimos da análise de sensibilidade, e a este respecto, comprobouse a utilidade dos métodos de sensibilidade descritos. O último experimento serve como demostración das limitacións dos métodos semi-recursivos, e manifesta a vantaxe de poder contar con librerías que soporten varios sistemas de modelado en diversas coordenadas.

Cabe destacar que tódolos resultados dinámicos e de sensibilidade se compararon cos obtidos con formulacións globais en coordenadas naturais, e obtivéronse extraordinarios niveles de converxencia a pesar da diferencia de modelos (naturais e relativas), diferencia de formulacións (ALI3-P e Matriz R), diferencia na selección de puntos de referencia (RTdyn0 e RTdyn1), diferencia de métodos de sensibilidade (adxunta e directa) e mesmo diferencia de filosofías de discretización (métodos adxuntos continuo e discreto). Os resultados obtidos deses experimentos numéricos validan tanto os desenvolvementos teóricos presentados ó longo da tese como a implementación na librería multicorpo de propósito xeral MBSLIM.

O traballo desenvolto nesta tese abre as portas a outros traballos futuros que non foron realizados polo momento debido á falta de tempo e/ou á magnitude dos mesmos. Entre eles, podemos destacar o desenvolvemento de formulacións totalmente recursivas con restricións, o estudo e implementación de algoritmos de optimización baseados en gradiente, a paralelización das ecuacións da dinámica e da sensibilidade, ou a extensión dos presentes métodos a sólidos flexibles.