ESCUELA POLITÉCNICA SUPERIOR

UNIVERSITY OF LA CORUÑA

# REAL–TIME METHODS IN FLEXIBLE MULTIBODY DYNAMICS

A thesis submitted for the degree of
Doctor Ingeniero Industrial

URBANO LUGRÍS ARMESTO

Ferrol, November 2008

ESCUELA POLITÉCNICA SUPERIOR

UNIVERSITY OF LA CORUÑA

# REAL–TIME METHODS IN FLEXIBLE MULTIBODY DYNAMICS

A thesis submitted for the degree of
Doctor Ingeniero Industrial

Urbano Lugrís Armesto

Advisor: Javier Cuadrado Aranda
Co–Advisor: Juana M. Mayo Núñez

**Ferrol, November 2008**

*A Pepe y Hortensia*

# AGRADECIMIENTOS

*En primer lugar, quisiera mostrar mi más sincero agradecimiento a los directores de esta tesis, Javier Cuadrado y Juana Mayo. Sus grandes ideas, junto con su estupenda labor de dirección, pueden considerarse el pilar fundamental de este trabajo. Especialmente, doy las gracias a Javier Cuadrado por haberme introducido en el apasionante mundo del* multibody.

*Quisiera también agradecer a la gente de la Universidad de Sevilla, en especial a Daniel García–Vallejo y José Luis Escalona, su inolvidable hospitalidad. Puedo afirmar que, gracias a ellos, me sentí como en casa durante los casi tres meses que duró mi estancia allí.*

*Y cómo no, recordar desde aquí el apoyo de todos los compañeros de trabajo en el Laboratorio de Ingeniería Mecánica de la Universidad de La Coruña: Alberto, Dani, Fran, Lolo, Miguel, Roland y Ruth. Su ayuda, así como el inmejorable ambiente de trabajo, han sido clave para la realización de esta tesis.*

*A Werner Schiehlen y Peter Eberhard, así como a los compañeros del* Institut, *les agradezco sinceramente el excelente trato recibido durante mi estancia en Stuttgart.*

*Y por último, a mis padres Urbano y Hortensia, y a mi hermana Victoria, les doy las gracias por la infinita paciencia de la que han hecho gala, sobre todo en las etapas finales de la tesis.*

## ABSTRACT

In many multibody system dynamics applications, in which the requirements of weight, operational speed, etc., are highly demanding, the deformation of certain elements of a mechanism can not be neglected. On the other hand, the currently available computational power makes the real–time simulation of flexible systems possible with standard workstations. The present work aims at developing formulations for flexible multibody dynamics meeting the efficiency, precision and robustness requirements of real–time applications.

The first chapter is a brief introduction to the existing developments, with the objective of situating the present work within the framework of flexible multibody dynamics.

In the second chapter, a new formulation for flexible multibody dynamics is presented. The method is an extension to the flexible case of an existing semi–recursive formulation for rigid systems in dependent relative coordinates. The flexible bodies are modeled by using the floating frame of reference approach with component mode synthesis, in order to meet the real–time requirements. Three different systems are simulated by using the new method, and the results are compared, in terms of efficiency and accuracy, to those obtained by means of a method in natural coordinates.

The third chapter addresses the calculation of the inertia terms by means of an alternative method, which does not depend on the size of the finite element model. The method uses the inertia shape integrals, which are a set of invariant integrals that can be obtained in a preprocessing stage, for achieving its objective. It is implemented in both the new formulation in relative coordinates, and the formulation in natural coordinates to which it is compared in the second chapter.

In the fourth chapter, three solutions for dealing with geometrically nonlinear problems are explored. The use of substructuring, a nonlinear stiffness matrix, and the inclusion of the axial foreshortening effect, are implemented and compared. Very good results are obtained when using the foreshortening method, capturing the geometric stiffening effect without problems where the linear method fails.

Finally, the conclusions extracted from the present work, along with the possible future developments, are presented in Chapter 5.

# RESUMEN

En numerosas aplicaciones de dinámica de sistemas multicuerpo, en las que los requerimientos de peso, velocidad de operación, etc. son muy exigentes, la deformación de ciertos elementos de un mecanismo no puede ser despreciada. Por otro lado, la potencia de los ordenadores actuales hace posible la simulación de sistemas flexibles en tiempo real con estaciones de trabajo estándar. El presente trabajo está enfocado al desarrollo de formulaciones para sistemas multicuerpo flexibles que satisfagan los requerimientos de eficiencia, precisión y robustez de las aplicaciones de tiempo real.

El primer capítulo es una breve introducción a los desarrollos existentes, con el objetivo de situar el presente trabajo en el marco de la dinámica de sistemas multicuerpo flexibles.

En el segundo capítulo, se presenta una nueva formulación para dinámica de sistemas multicuerpo flexibles. El método es una extensión de una formulación semi–recursiva en coordenadas relativas dependientes ya existente para sistemas rígidos. Los cuerpos flexibles son modelizados usando sistema de referencia flotante con síntesis de componentes, para cumplir con los requerimientos de tiempo real. Se simulan tres sistemas diferentes mediante el nuevo método, y los resultados son comparados con los obtenidos mediante un método en coordenadas naturales, en términos de eficiencia y precisión.

El tercer capítulo trata sobre el cálculo de los términos de inercia por medio de un método alternativo, que no depende del tamaño del modelo de elementos finitos. Para conseguir este objetivo, el método usa un conjunto de integrales de forma, que son constantes y se obtienen en una fase de preproceso. Se ha implementado tanto en la nueva formulación en coordenadas relativas como en la formulación en coordenadas naturales con la que se ha comparado en el segundo capítulo.

En el cuarto capítulo se exploran tres soluciones para abordar problemas que presentan no linealidad geométrica. Se han implementado y comparado el uso de subestructuras, matriz de rigidez no lineal, y la inclusión del efecto del *foreshortening* axial. El método del *foreshortening* ha dado muy buenos resultados, capturando el efecto de rigidización geométrica donde falla el método lineal.

Finalmente, las conclusiones extraídas del presente trabajo, junto con posibles desarrollos futuros, se presentan en el Capítulo 5.

# Contents

# Chapter 1

# Introduction

## 1.1 Motivation

The consideration of flexibility in the simulation of multibody systems has been increasingly acquiring interest since the early seventies. Especially in the aerospatial and robotics fields, the need for more lightweight and slender elements in mechanisms such as manipulator arms or deployable solar arrays, along with the obvious difficulties presented by the work with actual prototypes, increased the demand for simulation methods capable of including flexibility effects.

Nowadays, the simulation of flexible multibody systems is used in many engineering fields and applications, such as vehicle dynamics, biomechanics, manufacturing machines, serial and parallel robots, etc., and most commercial computer codes, such as ADAMS, RecurDyn, SAMCEF Mecano, or SIMPACK, along with many other academic ones, include capabilities for considering flexibility in their simulations. The increase of computing power experienced during the last years allows for including flexible bodies in applications where efficiency is of key importance, such as simulations with real–time requirements, optimization processes that require many evaluations of the simulation, etc.

Real–time requirements appear, among others, in the so–called *human–in–the–loop* and *hardware–in–the-loop* simulations, like virtual reality applications, driving simulators, simulation–driven controllers, etc., where the simulation is carried out in real–time, and the response to the inputs must be immediate. In these applications, the main concern is efficiency, but not forgetting accuracy and, more importantly, robustness. A compromise must be achieved, by combining these three factors in an adequate form. It is obvious that the efficiency must be such that the simulation is run in real–time or faster, and it depends on the combination of four factors: the type of coordinates and modelization, the dynamic formulation, the numerical integrator, and the computer implementation. In some cases, formulations that are theoretically less efficient than others since they perform more arithmetic operations, can achieve better performance if they are combined with the adequate computer implementation and numerical integrator.

In the Laboratory of Mechanical Engineering of the University of La Coruña, several works have been dedicated to the real-time simulation. On the one hand, a very efficient semirecursive formulation for the simulation of rigid multibody systems (Cuadrado et al., 2004a,b; Dopico, 2004) has shown very good capabilities for real-time applications. On the other hand, a method for the simulation of flexible systems in natural coordinates was tested and validated against experimental results (Cuadrado et al., 2004c; Gutiérrez, 2003). The objective of this thesis is the development of methods as efficient as possible for flexible multibody dynamics, in order to meet the requirements for real–time applications. In order to do that, a new formulation is defined, by extending the aforementioned semi–recursive formulation for rigid systems to the flexible case. In the new formulation, the modeling of flexible bodies is taken from the cited formulation in natural coordinates. The efficiency is then further improved by introducing an alternative method for calculating the variable inertia terms, which, as opposed to that used in the original formulation, does not depend on the size of the finite element models used. This method is introduced in both the new and the original formulation in natural coordinates. Finally, several methods for dealing with geometrically nonlinear problems such as beams undergoing large deflections are implemented, for extending the range of applicability of the formulations.

## 1.2 Background

As pointed out in the previous section, the simulation of flexible multibody systems has been under research for the last 30 years. In such a long time, many different methods have been developed, and the existing literature is currently very extensive. Several books, either fully or partially devoted to flexible multibody dynamics, have been already published, by authors like Amirouche (1992), Géradin and Cardona (2001) or Shabana (1998). Some review papers have also appeared along these years, such as that of Shabana (1997), Bremer (1999), or Wasfy and Noor (2003). The last one is a particularly comprehensive and well organized review of the different methods, having a total of 877 references. In that survey article, the formulations are classified according to the frame of reference chosen for the modeling of flexible bodies.

The elastic displacements, which are usually modeled by means of the finite element method, can be then referred to three different types of frame of reference, as shown in Figure 1.1, depending on the entity they are associated to.

- In the *floating frame of reference* formulations, FFR in what follows, the elastic displacements are measured in local coordinates, with respect to a floating frame attached to the body, which in turn undergoes the large amplitude or *reference* motion. In the figure, the local frame is that represented by the vectors **u** and **v**.

- The *inertial frame* or global formulations use the global inertial frame as the reference for the material positions of all the points of the mechanism, in such a way that there is no formal separation between the large amplitude rigid–body motion and the elastic displacements; therefore, the system variables are the absolute positions and orientations of all the finite element nodes.

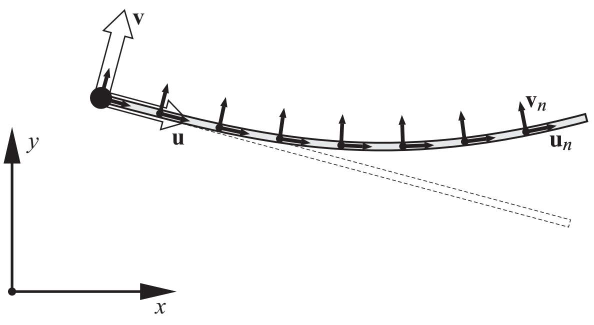


Figure 1.1: Inertial, floating and corotational frames of reference.

- The last type of formulations are the *corotational* ones, which define a corotational frame of reference, also known as convected frame, for each finite element, which is represented by the vectors $\mathbf{u}_n$ and $\mathbf{v}_n$ in the rightmost finite element of the figure.

Most of the currently used formulations fall into one of these three categories. Other methods, which do not clearly belong to any of them, are discussed at the end of this section.

### 1.2.1 Earlier developments in flexible multibody dynamics

#### Linear theory of elastodynamics

There exist several survey papers (Erdman and Sandor, 1972; Lowen and Chassapis, 1986; Lowen and Jandrasits, 1972) about the first approximated methods for flexible multibody dynamics, known as the *linear theory of elastodynamics*. In this approach, it is assumed that the elastic deformations have no significant effect on the global motion of the mechanism, so that the latter can be obtained from a preliminary rigid body simulation. The total motion is considered as the superposition of the elastic displacements to the global "nominal" motion, as it happens in FFR methods, although in this case they are not solved simultaneously. In order to obtain the elastic displacements, a dynamic analysis is carried out, assuming that the reference motion is that obtained from the rigid body simulation. This means that the elastic displacements are dynamically affected by the global motion, but the opposite effect, i.e. the influence of the deformations on the large amplitude motion, is completely neglected. Some of these methods (Sunada and Dubowsky, 1981, 1983) already introduced the use of component mode synthesis (Hurty, 1965) for reducing the number of unknowns.

#### Finite segment method

The finite segment method (Huston, 1981, 1991), can be considered as an application of the rigid body formalism to the simulation of flexible systems, more than as a flexible multibody formalism. The method consists of approximating a deformable body by a set of rigid bodies interconnected by force elements, in such a way that, provided that the discretization and the force elements are determined, the simulation can be carried out by using any existing method for rigid systems. This method has been successfully used for several applications, including vehicle crashworthiness tests (Nikravesh et al., 1983), or the simulation of flexible beams (Banerjee and Nagarajan, 1997; Zahariev, 2000). The most important problem it presents is, obviously, the definition of the discretization, along with the connection and parameters of the force elements, in such a way that the response of the equivalent flexible body is retained, especially in the case of complex geometries.

### 1.2.2 Floating frame of reference formulations

The floating frame of reference formulations, due to their high computational efficiency, are the most widely used in flexible multibody dynamics. In these methods, the motion of a flexible body is considered as the superposition of small elastic deformations to a large amplitude rigid–body motion. This reference motion is undergone by a local frame attached to the body, which can be represented, according to the specific formulation, by different types of rigid–body or reference coordinates. The elastic displacements are then considered within the local frame, after being linearized about the undeformed position. Due to this linearization, the use of these methods is, a priori, restricted to small–deformation problems, although some workarounds have been developed in order to extend their range of applicability.

The underlying idea of the FFR methods is that the absolute position of any given point of a flexible body, $\mathbf{r}$, can be expressed as

$$\mathbf{r} = \mathbf{r}_0 + \mathbf{A}\bar{\mathbf{r}} = \mathbf{r}_0 + \mathbf{A}\left(\bar{\mathbf{r}}_u + \bar{\mathbf{r}}_f\right) \tag{1.1}$$

where $\mathbf{r}_0$ is the absolute position of the origin of the local frame, and $\mathbf{A}$ is a transformation matrix that defines its orientation. The local position $\bar{\mathbf{r}}$ of the point within the local frame is, in turn, the sum of the elastic displacement, $\bar{\mathbf{r}}_f$, to the undeformed position, $\bar{\mathbf{r}}_u$, which is obviously constant. The elastic displacement can be modeled by using a finite element mesh defined in the local frame. As pointed out before, the deformations are assumed to remain small, so that in order to reduce the number of system variables, they can be approximated by using the component mode synthesis technique (Hurty, 1965), which approximates the deformation field as a linear combination of assumed mode shapes,

$$\bar{\mathbf{r}}_f = \sum_{i=1}^{\infty} \mathbf{\Phi}_i \eta_i \approx \sum_{i=1}^{n} \mathbf{\Phi}_i \eta_i \tag{1.2}$$

In this equation, $\mathbf{\Phi}_i$ are Ritz vectors defining the mode shapes, and $\eta_i$ are the so–called modal amplitudes, also known as elastic coordinates, which are in general added to the set of problem variables. An additional advantage of the component mode synthesis, apart from the reduction of the problem size, is the elimination of the higher frequency modes, which are undesirable from the numerical integration point of view.

FFR methods, as opposed to the earlier methods based on the linear theory of elastodynamics, take into account the coupling between the reference motion and the deformation, since they integrate the reference and elastic coordinates simultaneously. One of their most important drawbacks is that, due to this fact, the inertia terms are highly nonlinear. Another typical problem of FFR formulations, as pointed out by Kane et al. (1987), is their inability to capture some nonlinear effects such as the geometric stiffening in rotating beams, due to the linearization of the elastic displacements. The most general technique for avoiding this limitation, without introducing any modification to the formulation, is the substructuring strategy proposed by Wu and Haug (1988), which consists of dividing the body into several substructures, being each of them a flexible body with its own floating frame of reference. The adjacent substructures are interconnected through the so–called *bracket joints*, in such a way that the full set of substructures behaves like a whole body. This approach allows for capturing the nonlinear effects, although at the cost of a considerable increase in the number of variables. Other methods use a nonlinear formulation of the elastic forces (Sharf, 1996), or take into account the effect at geometric level, by introducing into the kinematic model the effect of the *foreshortening*, i.e. the axial displacement produced by the transversal deflection (Mayo et al., 1995). This aspect, however, is the object of the fourth chapter of this thesis, so that it will not be further discussed here.

As has been previously pointed out, the floating frame of reference formulations characterize the motion of the local frame by using rigid–body coordinates. Three main types of coordinates are commonly used for describing the motion of rigid bodies, namely the reference point coordinates, the natural coordinates, and the relative coordinates; therefore, the FFR methods can be classified accordingly. The difference among the different types of coordinates is illustrated in Figure 1.2, where a planar two degree of freedom manipulator arm is shown.

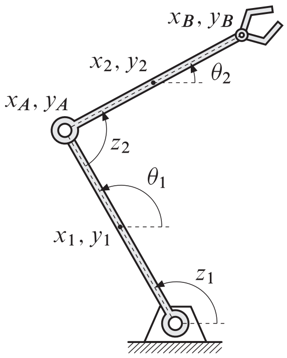


Figure 1.2: Reference point, natural and relative coordinates.

- The *reference point* coordinates model a rigid body by using the coordinates of one of its points, usually the center of gravity, for defining its translation, and at least three parameters, such as Bryant or Euler angles, for defining its orientation. Theoretically, this would allow for modeling the six degrees of freedom of a spatial rigid body, although, as it is well known, the representation of three–dimensional rotations by using only three parameters presents problems of singularity at certain positions, so that the common choice is the use of redundant parameters, such as the four Euler parameters, or $3 \times 3$ orthogonal rotation tensors. In the 2D manipulator arm shown in the figure, the reference point parameters would be the positions of the centers of gravity of the two arms $(x_1, y_1)$ and $(x_2, y_2)$, along with their orientation angles with respect to the global frame, $\theta_1$ and $\theta_2$. This makes a total of six parameters, to which four constraints must be applied for keeping the hinge points coincident.

- The *natural* coordinates (García de Jalón and Bayo, 1994) consist of points and unit vectors, expressed in Cartesian coordinates. In general, the points coincide with the positions of the kinematic pairs, and the unit vectors define their principal axes. This allows for modeling many kinematic pairs by sharing coordinates. Moreover, when using natural coordinates, the mass matrix of a rigid body can have a constant value, depending on the parameters chosen for its modeling. In the figure, the natural coordinates are the positions of the two revolute joints $(x_A, y_A)$ and $(x_B, y_B)$, thus making a total of four parameters. It is observed that the coordinates $x_A$ and $y_A$ are shared for both bodies, so that no algebraic constraint will be needed for the corresponding revolute joint. Only two constraints are needed, in this case for imposing the constant length of the links, thus reducing the dimension of the problem from 6 to 4.

- The *relative* coordinates are relative distances and angles between adjacent bodies. They usually form a minimal set of parameters for defining the position of a mechanism, which may be independent in the case of open–loop topologies. The relative coordinates relate the position of a body to that of its predecessor in the kinematic chain, which allows for implementing very efficient recursive algorithms for the solution of the equations of motion, although these methods are in general much more involved than the methods that use absolute coordinates. Many of these methods use symbolic algebra for obtaining the equations of motion, thus allowing for eliminating many repeated operations, leading to highly efficient methods. In the figure, the relative parameters are the angles $z_1$ and $z_2$, and, since the mechanism has two degrees of freedom, no additional constraints are required.

### FFR formulations in reference point coordinates

The method developed for the planar case by Song and Haug (1980) is the first FFR method that totally accounts for the inertia coupling between the reference motion and the local elastic displacements. The method is later generalized to the three–dimensional case and further developed by several authors (Shabana and Wehage (1983); Agrawal and Shabana (1985); Shabana (1985, 1998)). In these methods, the reference motion is parameterized by using a point and the four Euler parameters, and the elastic displacements are approximated by means of component mode synthesis. In Agrawal and Shabana (1986), the problem of how the floating frame should be attached to the body is addressed, introducing the concept of mean–axis frame. A mean–axis frame is not attached to a physical point of the body, and leads to the minimal coupling between the reference motion and the elastic deformation. The already mentioned substructuring method, proposed by Wu and Haug (1988) for addressing the nonlinearity problems, is based on this FFR method in reference point coordinates.

### FFR formulations in natural coordinates

One of the first flexible multibody implementations in natural coordinates is that introduced by Vukasovic (1990). According to the natural coordinates formalism, the frame of reference is modeled by a point and two orthogonal unit vectors, and the elastic displacements are approximated by means of a Craig–Bampton modal reduction (Craig and Bampton, 1968), in local coordinates. That formulation, however, is not fully consistent with the natural coordinates philosophy, since the coordinates that model the boundary points and unit vectors are those of their fictitious undeformed position, instead of the actual deformed position and orientation. This means that one of the main advantages of the natural coordinates, i.e. the possibility of sharing boundary entities to reduce the number of kinematic restrictions, is lost. Another drawback of this method is that it uses a velocity transformation technique for obtaining a system of equations in independent coordinates, and the calculation of that velocity transformation, which can be advantageous in rigid systems, is rather inefficient in the flexible case, due to the higher number of independent coordinates.

The method of Cuadrado (1993), also found in Cuadrado et al. (1996), addresses the two main drawbacks of this method. In this case, each flexible body is modeled by a point, three unit vectors, and a set of Craig–Bampton modes. The amplitudes of the static modes are eliminated, since they can be expressed as a function of the frame variables and the actual deformed boundary points and vectors. The coordinates of a body are then the frame parameters and the Cartesian coordinates of the boundaries, plus the dynamic modal amplitudes, which are independent. Moreover, the equations of motion are stated in dependent coordinates, by using a classical Lagrangian formulation with Baumgarte stabilization (Baumgarte, 1972). The inertia terms, i.e. the mass matrix and the velocity dependent inertia forces vector, are obtained by integrating the kinetic energy over the whole volume of the body, thus obtaining fully consistent inertia terms, but at the cost of a rather involved implementation, despite the use of invariant matrices (Shabana, 1991) to reduce the computation time.

In order to simplify the calculation of the inertia terms, Avello (1995) introduced a method based on the corotational inertia proposed by Cardona and Géradin (1988), which enables to obtain the mass matrix as a projection of the standard mass matrix of the finite element method into the coordinates of the body, although with some further simplifications. Avello assumed that the velocities could be interpolated in the global frame by means of the finite element interpolation functions, which is true in the case of isoparametric finite elements but, in the case of structural elements using infinitesimal rotations as nodal coordinates, such as beams, plates and shells, this is only an approximation. Another difference with respect to the method of Cuadrado is the explicit use of the static modal amplitudes as system variables. This work has been continued by Gutiérrez (2003), who introduced a nonlinear stiffness method to account for the geometric stiffening in rotating beams. This FFR formulation in natural coordinates has been validated against nonlinear finite element methods (Cuadrado et al., 2001), and also against experimental tests, by comparing the results to the actual measured stresses in the chassis of a prototype car (Cuadrado et al., 2004c). In this case, the same index–3 augmented Lagrangian formulation used in the present work is used for integrating the equations of motion.

### FFR formulations in relative coordinates

Prior to describing the FFR formulations in relative coordinates, the recursive methods for rigid body systems, on whose principles the flexible formulations rely, are briefly introduced here.

Traditionally, the efficiency of recursive methods has been associated to their computational complexity, understood as the number of floating point operations needed for obtaining the accelerations. This number of operations, for a given formulation, is a function of the number of elements of the mechanism, $n$, and the order of complexity of a formulation is defined as the exponent of $n$ in that function. Although some other methods with different orders of complexity exist, the majority of them can be considered as either semi–recursive $O\left(n^3\right)$ or fully–recursive $O\left(n\right)$ formulations.

The first family of formulations, those with an $O\left(n^3\right)$ complexity, were initially developed by Walker and Orin (1982) for open–loop robotic manipulators. The main idea of the original method is to take advantage of an inverse dynamics algorithm, more specifically the recursive Newton–Euler method (Featherstone, 1987), for the solution of the forward dynamics problem. The inverse dynamics problem consists of finding the forces and moments required at the actuators of a mechanism, $\boldsymbol{\tau}$, for obtaining a prescribed motion, according to the following equations of motion

$$\mathbf{M}(\mathbf{z})\ddot{\mathbf{z}} = \boldsymbol{\tau} - \mathbf{Q}(\mathbf{z}, \dot{\mathbf{z}}) \tag{1.3}$$

where the vector $\mathbf{Q}$ absorbs all the generalized forces not introduced by the actuators, such as the centrifugal and Coriolis forces, the gravitational forces, etc. Since the motion is prescribed, the relative positions $\mathbf{z}$, velocities $\dot{\mathbf{z}}$ and accelerations $\ddot{\mathbf{z}}$ are known a priori. The inverse dynamics algorithm obtains first the Cartesian positions, velocities and accelerations of all the elements of the mechanism, by performing a forward recursive analysis from the root to the leaves. Then, the forces and inertias are accumulated in the inverse direction, from the leaves to the root. During this process, the dynamic equilibrium is imposed at each kinematic pair, thus obtaining the unknown actuator forces. In order to solve the forward dynamics, the method proposed by Walker and Orin uses the inverse Newton–Euler method, by applying combinations of zero and unit values to the accelerations, for calculating the mass matrix $\mathbf{M}$ and the forces vector $\mathbf{Q}$. Once they have been determined, the accelerations are obtained by solving the resulting linear system in Eq. (1.3). The $O\left(n^3\right)$ complexity resides at this point, since the number of operations needed to solve a linear system grows with the cube of its dimension.

A more efficient revision of this method, referred to as *method 3* in the paper of Walker and Orin, and as *composite rigid–body method* in Featherstone (1987), uses an alternative recursive method, going from the leaves to the root, for calculating the mass matrix. First, the mass matrices of the individual bodies are computed in Cartesian coordinates, at the current position. These individual mass matrices are all referred to the global origin, instead of the center of gravity of each body, in order to facilitate their accumulation. The composite mass matrix at a joint $j$, for a given position of the mechanism, is the equivalent mass matrix that all the bodies placed between the joint and the end of the tree would have, if considered as a composite rigid body. This accumulated mass matrix can be obtained as the sum of the individual matrices of all the corresponding bodies, since all of them are calculated with respect to a common point. Finally, by using the recursive relations at the joints, the mass matrix in relative coordinates is assembled, which allows for stating the linear system that leads to the relative accelerations.

Featherstone (1983) introduces a fully–recursive $O(n)$ formulation, the *articulated inertia method*, which was proven to be more efficient than the semi–recursive ones for systems with more than ten elements (Featherstone, 1987). This method was further developed by Bae and Haug (1987), who use a different notation, and obtain the equations of motion from a variational approach. In the fully–recursive algorithms, the backward recursive accumulation of inertias and forces is performed by introducing the kinematic relations. The resulting mass matrices and forces vectors can then be used, in a second forward recursive analysis from the root to the leaves, for obtaining the accelerations at each joint. If an additional body is added to the mechanism, one more step is added to each recursive propagation, so that the number of operations grows according to an $O(n)$ law.

The same authors extended the fully–recursive method to closed–loop topologies (Bae and Haug, 1988). In order to apply the method to a closed–loop system, the closed loops are cut first, in order to obtain an open–loop version of the mechanism. Then, the reaction forces, which are introduced by means of Lagrange multipliers, are propagated as unknowns, along with the other forces, in the backward recursive inertia and force accumulation, thus keeping the fully–recursive $O(n)$ nature of the method.

In Jiménez (1993), a variation of the composite rigid–body method is presented, based on the notation introduced by Bae and Haug for the $O(n)$ formalism. In this case, the equations of motion in relative coordinates are obtained by means of a velocity transformation, although the underlying idea is the same as in the composite rigid–body method. This $O\left(n^3\right)$ method is generalized for closed–loop systems, by using two different solutions. The first one consists of stating the equations of motion in relative dependent coordinates, by using a penalty Lagrangian formulation (Bayo et al., 1988). The second solution performs a second velocity transformation (García de Jalón and Bayo, 1994), for obtaining a reduced set of independent coordinates, so that the equations of motion can be integrated as a standard ODE system. In the same work, a fully–recursive method, that uses the global origin as reference for the Cartesian mass matrices, as done in the composite rigid–body method, is also introduced. In Stelzle et al. (1995), the computational complexity of fully–recursive formulations is thoroughly analyzed and compared, concluding that the global origin is the reference point leading to the lowest complexity, if the dynamic terms are obtained in global coordinates.

Dopico (2004) performs a comparison between the formulation in natural coordinates, and the two $O\left(n^3\right)$ methods in relative coordinates proposed by Jiménez, by simulating several systems of different sizes. The results of the comparison indicate that the penalty semi–recursive method is more efficient than that in natural coordinates for large systems. On the other hand, the semi–recursive method in independent coordinates, in spite of integrating a smaller set of coordinates, is found to be less robust and efficient than its penalty–based counterpart. According to these results, the $O\left(n^3\right)$ method in dependent relative coordinates is chosen for being extended to the flexible case in this thesis.

One of the first FFR methods for flexible systems in relative coordinates is that described by Book (1984). The method, limited to the simulation of open–loop robots, shares some features with that of Sunada and Dubowsky, such as the Lagrangian formulation of the equations of motion, the use of the Denavit–Hartenberg parameterization (Hartenberg and Denavit, 1963) for the recursive relations, and the reduction of the finite element models by means of component mode synthesis. The method proposed by Book, however, fully accounts for the inertia coupling between the large amplitude and the elastic motion, instead of obtaining the former from a rigid body analysis. Each flexible link is modeled by means of two frames of reference, one placed at the input joint, which is the actual floating frame of reference of the body, and another frame at the output joint, whose displacement and rotation with respect to the body frame depend on the modal amplitudes. The recursive relations are as follows: from the parameters of the local frame of a body $i$, along with the modal amplitudes, the position of the output frame is determined. Then, the input frame of the next body $j$ is related the output frame of $i$ by means of a transformation matrix, which is a function of the relative coordinate between links $i$ and $j$. Thus, the orientation of the input frame of body $j$ is obtained as,

$$\mathbf{A}_j = \mathbf{A}_i \mathbf{A}_{fi} \mathbf{A}_r \tag{1.4}$$

where $\mathbf{A}_i$ and $\mathbf{A}_j$ are the absolute transformation matrices of the input frames of links $i$ and $j$ with respect to the inertial frame. The transformation $\mathbf{A}_{fi}$ is produced by the deformation of body $i$, and the resulting frame is transformed by $\mathbf{A}_r$, which is a function of the relative coordinates. In the formulation presented here, a similar approach is used, although it does not use the Denavit–Hartenberg formalism for the recursive relations, and, moreover, the floating frame does not necessarily have to be placed at the input joint. This method pertains to the family of $O\left(n^3\right)$ formulations, since it uses the recursive relations for obtaining the equations of motion in relative coordinates, and then solves a linear system for the accelerations.

The method proposed by Kim and Haug (1988, 1989) represents the natural extension to the flexible case of the method of Bae and Haug for rigid systems. In this case, the floating frame of the body is placed at the center of gravity, so that joint frames are required at both the input and output joints. The flexible bodies are modeled by using component mode synthesis, and the inertia terms are approximated by using a lumped mass approach. This formulation is also developed for its use in closed–loop systems, and uses the same $O(n)$ articulated inertia approach for the calculation of the accelerations.

Later, Shabana et al. (1992) and Wehage et al. (1992) introduced a new recursive formulation, which is only described for open–loop systems in the cited references. The kinematic modeling is the same as that used by Kim and Haug, although it presents several differences. On the one hand, the inertia terms are calculated by using consistent mass integrals, by means of the inertia shape integrals already used by the same authors in non–recursive algorithms. Due to the fact that the inertia properties of the elastic coordinates are constant, the second time derivatives of the modal amplitudes are eliminated by means of the Gauss–Jordan method prior to the recursive solution of the accelerations. This operation is efficiently performed since the block of the mass matrix corresponding to the elastic coordinates must be inverted only once. This is a recursive $O(n)$ formulation, although the equations of motion are stated from a Newton–Euler perspective. At the end of the second one of the cited papers, however, the modal elimination in an augmented Lagrangian approach is briefly commented.

The work of Znamenáček and Valášek (1998), described only for open–loop systems, is a reformulation of the $O(n)$ approach, based on the Gauss principle of least constraint. This method also performs the modal amplitudes elimination and uses the fully consistent inertia approach. In the cited work, two different orthogonalization methods, intended for optimizing the computational complexity, are introduced. Each one of the methods is optimized for bodies with either less or more than 17 mode shapes.

In Bae et al. (2001) a method for including flexible bodies into an existing rigid–body code is presented. The idea is to define modules that introduce the flexible bodies as virtual joints, in such a way that the core software is not modified.

The formulation recently introduced by Vampola and Valášek (2007) is very similar to that presented in this thesis, since it is an $O\left(n^3\right)$ method that relies on the ideas of the composite rigid body method of Walker and Orin, and accounts for open– and closed–loop topologies. This formulation is different in many aspects, however, since the equations of motion are stated, as in Znamenáček and Valášek (1998), by means of the Gauss principle. The inertia terms are obtained by using a lumped mass approach, and the number of nodes of the discretization appears in the total number of floating point operations, meaning that efficiency depends on the resolution of the mesh.

### 1.2.3 Inertial frame formulations

The formulations in absolute coordinates are derived from the finite element method, so that they use the nodal coordinates as system variables. The conventional finite element formulations (Bathe, 1995) do not allow for rigid body rotations without introducing elastic deformation, thus making them unsuitable for the simulation of flexible multibody systems. The absolute formulations introduce different modifications to the finite element method, aimed for obtaining finite elements that are invariant to rigid body rotations. In general, these formulations differ in the parameters they use for defining the orientation of the finite element nodes, being the positions defined by absolute Cartesian coordinates in all of them.

Most of the inertial frame formulations are developed for the modeling of flexible beams. The common approach is to define the position and orientation of a finite element node by its Cartesian coordinates, and its orientation, usually represented by a trihedron that remains perpendicular to the cross section. The formulation of Simó and Vu-Quoc (1986a,b,c,d) is one of the earliest applications of the inertial frame approach. In this formulation, developed first for two–dimensional beam elements and then generalized to the three–dimensional case, a reference trihedron is defined at every point of the beam, such that one of its directions is always orthogonal to the cross section. In order to interpolate its orientation, the Euler parameters defined at the nodes are used. In this formulation, as it happens to all the formulations based on the inertial frame approach, the mass matrix is a simple expression, being the elastic forces the most complicated term to evaluate.

A similar formulation is that proposed by Cardona and Géradin (1988), which introduces the *rotational vector* for the parameterization of the section orientations, instead of the Euler parameters. These authors developed a complete formulation, in the context of a general purpose multibody simulation software.

The method of Jonker (1989) shares some features with the FFR formulations. This method is intended for beam elements only, and uses the position and Euler parameters as nodal coordinates, being the intermediate points interpolated by means of cubic polynomials. The similarity with the substructuring FFR method of Wu and Haug (1988) is clear, since the finite elements can be seen as beams with only static deformation modes, and the nodal coordinates are shared between elements thus behaving like bracket joints.

An inertial frame formulation in natural coordinates, intended to be fully compatible with rigid or FFR methods based on the same coordinates, is presented by Avello (1990) and Avello et al. (1991). This method uses, as nodal parameters, the position of a point and three orthogonal unit vectors, all of them expressed in fully Cartesian coordinates, thus making a total of 12 variables per node. Six algebraic constraints must be then introduced for each trihedron, to keep its vectors orthonormal, as happened to the local frame vectors in FFR formulations. To avoid the interpolation of angles or Euler parameters, the orientation of the intermediate sections is obtained by interpolating the unit vectors themselves, thus obtaining as a result a constant mass matrix. This approach, however, is not completely exact, since the interpolated vectors are no longer unit and orthogonal, although the introduced error can be acceptable, converging to the exact solution as the mesh is refined.

A different approach is that presented by Shabana in the Absolute Nodal Coordinates Formulation (ANCF). This formulation uses positions of points and global slopes as nodal coordinates. In the earliest forms of the formulation, a corotational frame was defined at each element, which was then modeled in local coordinates as a two–dimensional Euler–Bernoulli beam (Escalona et al., 1998). In order to avoid the use of such local frames, Omar and Shabana (2001) reformulated the planar beam, calculating the elastic forces by using continuum mechanics, and then this approach was generalized to the three–dimensional beam element by Shabana and Yakoub (2001) and Yakoub and Shabana (2001). Later, Sopanen and Mikkola (2003) studied the elastic forces obtained by using this approach, finding that a beam element of this characteristics suffers of poor convergence due to shear locking problems. Several authors studied alternative workarounds, such as Dufva et al. (2005), who interpolate separately the rotation of the section due to deflection and shear forces, or Von Dombrowski (2002), who returns to the local frame approach and loses one of the main advantages of the ANCF, the constant mass matrix. The ANCF is not limited to beam elements, and several authors (Dmitrochenko and Pogorelov, 2003; Mikkola and Shabana, 2003) have also developed plate and shell elements. In García-Vallejo et al. (2003) and García-Vallejo (2006), a methodology for integrating flexible bodies modeled with ANCF into a rigid body formulation in natural coordinates is developed.

### 1.2.4 Corotational frame formulations

The first reference about this approach is the work of Belytschko and Hsieh (1973), who formulated it for planar beam and triangle elements. The idea has been further developed by many other authors, such as Crisfield (1997). These methods are intended for addressing the problems presented by the use of structural finite elements in the analysis of multibody systems. Structural elements, as opposed to the isoparametric ones, use infinitesimal rotations as nodal coordinates, and their interpolation functions do not correctly model the finite rigid–body rotations. However, if it is assumed that they are valid for small rotations, a local frame can be attached to every finite element, in such a way that the rotation between two consecutive configurations is correctly modeled in local coordinates. Two main families of these methods exist: the total Lagrangian (TL) methods, which calculate the deformations with respect to an undeformed reference position, and the updated Lagrangian (UL) ones, that refer them to the position in the last time–step. One common problem of these methods is that they do not model the exact rigid body dynamics. In order to overcome this problem, Shabana (1996) introduces a method that actually models the exact rigid body inertia, by using four different frames of reference simultaneously.

### 1.2.5 Other types of formulations

#### Discrete time transfer matrix method

The discrete time transfer matrix method, which has already been used in the structural mechanics field, was first adapted for its use in the simulation of rigid multibody systems by Rui et al. (2005). In this method, a state vector is defined at every connection point, containing its six position and orientation parameters, along with the internal force and moment vectors. The dynamic equations can be stated for an entity $j$, be it a body or a joint, then linearized, in order to obtain a position–dependent transfer matrix $\mathbf{U}_j$, that yields the state vector of the output point $\mathbf{z}_{jk}$ as a function of that at the input point $\mathbf{z}_{ij}$:

$$\mathbf{z}_{jk} = \mathbf{U}_j \mathbf{z}_{ij} \tag{1.5}$$

This recursive relation means that a transfer matrix between two given entities can be obtained as the product of all the transfer matrices existing between them. This allows for stating the equations of motion of a given body in terms of the transfer matrix between itself and the inertial frame of reference. Since, in the three–dimensional case, all these matrices are of dimension $13 \times 13$, the dimension of the problem does not depend on the number of bodies. There exists a different type of transfer matrix according to the entity it is associated to, and in the case of kinematic joints, the transfer matrix depends also on the type of the adjacent entities, so that a wide variety of transfer matrices need to be defined in order to cover all the possible configurations.

An application to flexible multibody dynamics is found in He et al. (2007), where a flexible beam is modeled by means of the finite segment method (Banerjee and Nagarajan, 1997). The discrete time transfer matrix method is later used by Rui et al. (2008) to compute the vibrational response of a multibody system including elastic beams, although no dynamic simulation is performed in that work.

### Global modal parameterization

The global modal parameterization (GMP), introduced by Brüls et al. (2007), consists of applying a configuration–dependent modal reduction to the whole system, at a given configuration range, instead of doing it on a per–body basis. The flexible bodies are modeled in the global inertial frame of reference, at a given position, by using the finite element method. The coordinates of the system, prior to the application of the reduction, are divided into the relative coordinates at the actuators, $\boldsymbol{\theta}$, the coordinates of other points where external forces might be applied, $\mathbf{q}^g$, and the remaining finite element internal degrees of freedom, $\mathbf{u}$.

For a given configuration, three sets of modes are defined. Each rigid–body mode is the result of applying a unit displacement to an actuator degree of freedom while keeping the remaining actuators fixed. The constraint modes are static deformation modes obtained by doing the same to the $\mathbf{q}^g$ coordinates, while keeping the actuators fixed. The remaining modes are obtained by fixing the rigid and constraint degrees of freedom, and obtaining the eigenmodes for the remaining coordinates $\mathbf{u}$. The method uses then a set of independent coordinates, which means that it is only valid within singularity–free regions of the subspace of possible configurations.

In practice, the method divides the domain of possible configurations into different subdomains within which a specific modal reduction is valid, so that only a finite number of mode sets will be needed. In order to assess the validity of the modal reduction within a configuration subdomain, the modal assurance criterion (MAC) is used. Moreover, a tracking strategy must be implemented in order to ensure the adequate identification of the dynamic modes according to their shape, since their frequencies can vary from one configuration to another.

As long as the number of degrees of freedom of a mechanism, i.e. the dimension of the possible configurations domain, grows, it is obvious that the amount of data needed for covering all the possible positions might increase exponentially. However, this method is primarily intended for control applications, where the trajectories are usually known in advance, so that the domain of possible configurations can be limited to cover only those trajectories.

**Recursive finite element formulations**

The concept of these formulation is a combination between the inertial frame approach and the recursive methods. Avello (1990) presented the idea of representing the coordinates of the finite element nodes of his formulation in a recursive way, although the method could become inefficient due to the need of calculating a velocity transformation for each node. The method of Bae (2005) is based on the same ideas, i.e. a finite element method in which the nodes are defined in relative coordinates. This approach is very similar to the FFR substructuring method, as used in relative coordinates by Kim and Haug (1988).

## 1.3 Objectives

The objectives of this work are the following:

- Definition of a new FFR formulation for flexible multibody dynamics in relative coordinates. The formulation is based on the $O\left(n^3\right)$ penalty method described for rigid systems in Jiménez (1993) and Dopico (2004), and models the flexible bodies as done in Avello (1995) and Gutiérrez (2003). The new formulation must be totally compatible with the rigid body formulation it is based on, in order to allow for combining both rigid and flexible bodies. It must keep, as much as possible, the philosophy and simplicity of the original semi–recursive method. The geometry of the flexible elements must not be limited to any particular type, accepting flexible body data obtained from any external means, including commercial finite element software, experimental analysis, etc.

- Evaluation of the capabilities of the new formulation for real–time applications, by comparing it to the formulation in natural coordinates (Gutiérrez, 2003), in terms of efficiency and robustness. The comparison is performed for systems with different sizes, in order to verify if the advantages shown in rigid systems are kept when flexible bodies are included in the simulation, and how the number of flexible bodies may affect the results.

- Calculation of the inertia terms by means of the inertia shape integrals or invariants in both the FFR formulations in absolute (natural) and relative coordinates, and evaluation of the efficiency and range of applicability with respect to the projection method used in Avello (1995) and Gutiérrez (2003).

- Implementation of different methods for capturing the geometric stiffening effect, comparing the alternatives in terms of efficiency, accuracy and simplicity of implementation.

# Chapter 2

# Method in Relative Coordinates

## 2.1 Introduction

In this chapter, a new formulation for flexible multibody dynamics based on relative coordinates is introduced. The formulation is the result of combining a method in relative coordinates for rigid multibody systems, with a floating frame of reference formulation in natural coordinates. The main objective of this work is, apart from implementing flexibility into a semi–recursive formulation for rigid systems (Cuadrado et al., 2004a,b; Dopico, 2004), is to see whether the benefits obtained by using relative coordinates are kept in flexible systems or not.

The formulation in relative coordinates in which the present work is based is a reformulation, in terms of a velocity transformation, of the composite rigid–body inertia method, originally developed by Walker and Orin (1982). The velocity transformation uses the recursive relations defined by Bae and Haug (1987) for fully–recursive methods, although in this case the inertia terms are calculated with respect to the global origin, as in the original composite rigid–body inertia method, instead of the center of mass. The method performs a forward recursive analysis for calculating the positions and velocities, followed by a backward recursive accumulation of forces and inertias from the leaves to the root, and then obtains the accelerations by solving a linear system. According to this, the method pertains to the $O\left(n^3\right)$ family, i.e. methods whose computational complexity grows with the cube of the number of elements $n$.

The method is intended for both open– and closed–loop systems; in the latter case, the mechanism is transformed into an open–loop system with one or more open chains, by means of the cut–joint method. This means that the formulation uses a set of relative dependent coordinates, and in order to close the loops, the corresponding kinematic constraints must be imposed, in this case by means of an augmented Lagrangian formulation, which uses a set of kinematic constraints defined in natural coordinates (García de Jalón and Bayo, 1994). The method calculates the accumulated inertia and generalized forces corresponding to the relative coordinates, which are first obtained in an intermediate Cartesian coordinate system, then projected into the relative coordinates by means of a variable matrix. This matrix projection is performed in a very efficient recursive way, yielding the terms of the equations of motion expressed in the relative dependent coordinates. In order to perform the time integration, two solutions are explored in Cuadrado et al. (2004a), Dopico (2004). The first one consists of obtaining the equations of motion expressed in a subset of independent coordinates by using a second velocity transformation, whereas the second one consists of the direct integration of the DAE system in dependent coordinates, along with kinematic constraints, using an augmented Lagrangian formulation. The second approach has been chosen here, due to the good results obtained in the cited works. This formulation is efficient, robust, and relatively simple to implement.

The flexibility problem is addressed by means of the floating frame of reference (FFR) approach (Shabana, 1998). The flexible bodies are modeled as in a previously existing method in natural coordinates (Cuadrado et al., 2001, 2004c; Gutiérrez, 2003). In the cited method, each flexible body has a local frame of reference attached to it, which is defined by a point at the origin and three orthogonal unit vectors along the axes. This frame experiences the large amplitude motion, and deformations are added on local coordinates, by using component mode synthesis to reduce the model size. This method has been chosen because, due to how the elastic coordinates are defined, it integrates in a very convenient way into the formulation in relative coordinates. Some work has been previously done in this direction by Funes et al. (2004), but using the double velocity projection instead of the DAE integration.

The integration of the equations of motion is performed stating them as an index–3 DAE, with the positions as primary variables (Cuadrado et al., 1997), and then combining them with a numerical Newmark integrator (Newmark, 1959). This method needs to perform subsequent velocity and acceleration projections in order to fulfill the kinematic constraints at the velocity and acceleration levels, since the index–3 augmented Lagrangian formulation only enforces their fulfillment at position level (Cuadrado et al., 2000). The resulting method has a very good balance among accuracy, efficiency and robustness, and this behavior is kept in flexible systems as shown by the results obtained in this thesis.

In this chapter, the method in relative coordinates for flexible multibody systems will be thoroughly described. The first section explains the kinematic description of the system, which can be considered as divided into two parts: the modeling of the flexible bodies, and the kinematics of the open–loop system in relative coordinates. In the next section, the calculation of the inertia terms is addressed. This is followed by the description of the additional non–inertial forces that may appear in the system, including applied forces and moments, springs and dampers, and volume forces. Then, the problem of the kinematic constraints is addressed. Finally, the assembly and integration of the equations of motion is explained in detail, including the velocity projection needed to express all the previously described terms in the relative coordinates. Three examples are simulated in the results section, including a planar double four–bar mechanism, a vehicle suspension, and a full vehicle, using both the formulation in natural coordinates, and that in relative coordinates described in this chapter. Finally, some conclusions and criteria of use are extracted from the obtained results.

## 2.2 Modeling and kinematics of the flexible body

According to the FFR formalism, the motion of a body is studied as the superposition of two components: a large amplitude motion, undergone by a local floating frame, and small local elastic displacements with respect to that frame. This separation begins at the kinematic modeling stage, so both components of the motion, although not independent, will be studied separately: first, how the elastic body is modeled within the local frame and, then, how this local frame moves in the global frame and how the flexibility affects the global motion.

The modeling of flexible bodies is essentially the same for all FFR formulations: each flexible body is attached to a floating frame of reference, which undergoes the large rigid body displacements and rotations, being the total motion the result of adding the elastic deformation, obtained in local coordinates with respect to the floating frame, to the motion of the frame itself. Following the partition introduced by Shabana (1991), the set of generalized coordinates needed to describe the motion of a flexible body can be considered as divided into two subsets: the reference coordinates, $\mathbf{q}_t$ and $\mathbf{q}_\theta$, which represent the position and orientation, respectively, of the floating frame of reference, and the elastic coordinates, $\mathbf{q}_f$, also noted as $\mathbf{y}$, which model the local elastic deformation. The former depend on the type of rigid body coordinates used for modeling the large amplitude motion, whereas the latter in most cases consist of the modal amplitudes of a Rayleigh–Ritz reduction.

The proposed formulation in relative coordinates for flexible bodies is based on a previously existing one, developed for rigid body dynamics (Cuadrado et al., 2001, 2004a; Dopico, 2004). In this formulation, the dependent set of relative coordinates is named $\mathbf{z}$, to which only the elastic coordinates must be added in order to include flexible bodies. This means that the reference coordinates are the relative ones and, since it is difficult to obtain the inertia terms directly expressed in these coordinates, an intermediate Cartesian coordinate set $\mathbf{Z}$ is used for this purpose. These inertia terms are subsequently projected into the relative coordinates in order to build the equations of motion, and this projection is performed in an optimal way due to the recursive relations established between neighbor bodies. Therefore, it can be said that two different sets of reference coordinates exist for each body, whereas the elastic coordinates are common to both the relative and the Cartesian sets, since they represent the deformation with respect to the same floating frame, no matter which set of coordinates is used to model its motion.

### 2.2.1 Floating frame of reference in natural coordinates

In the FFR formulation in natural coordinates (Cuadrado et al., 2004c), the local frame of each flexible body is represented by the absolute position of its origin $\mathbf{r}_0$, and by three orthogonal unit vectors $\mathbf{u}$, $\mathbf{v}$ and $\mathbf{w}$, also expressed in the inertial frame of reference, which define its orientation, making a total of 12 variables. The frame behaves as a rigid body with six degrees of freedom, so that six kinematic constraints are needed for keeping the three vectors orthonormal (García de Jalón and Bayo, 1994). These vectors can be considered as the three columns of a rotation matrix **A**, which allows for transforming a vector in local coordinates into its global coordinates counterpart. In what follows, all vectors expressed in local coordinates are noted with a bar on top. According to this, the absolute position **r** of an arbitrary point $P$ of a deformed body, as seen in Figure 2.1, is defined as follows:

$$\mathbf{r} = \mathbf{r}_0 + \mathbf{A}\bar{\mathbf{r}} = \mathbf{r}_0 + \mathbf{A}\left(\bar{\mathbf{r}}_u + \bar{\mathbf{r}}_f\right) \tag{2.1}$$

where $\bar{\mathbf{r}}$ is the deformed position in local coordinates, which is in turn equal to the sum of the undeformed position $\bar{\mathbf{r}}_u$ plus the elastic displacement $\bar{\mathbf{r}}_f$. The elastic displacement is a continuum field, characterized by a reduced set of elastic coordinates, which depend on how the flexible body is modeled in the local frame.

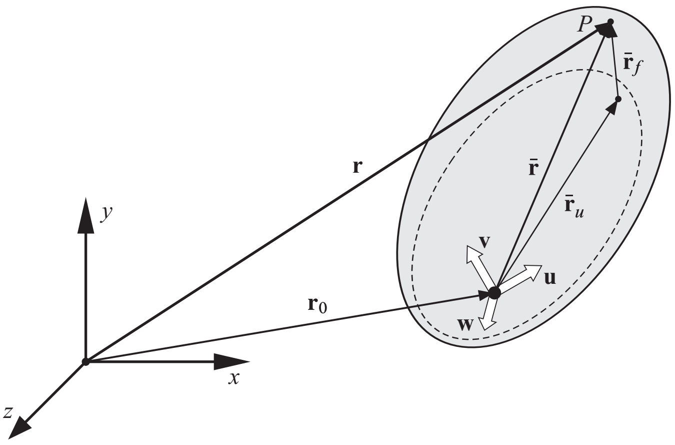


Figure 2.1: General flexible body.

The frame of reference can be connected to the flexible body by establishing different conditions, known in multibody dynamics as *reference conditions*. One common approach is the so called *tangent frame*, which is rigidly attached to a material point of the body. An opposite philosophy is the *mean–axis* frame, which is chosen in such a way that the mean elastic displacements with respect to it are minimized; this is also known as the Tisserand or Buckens frame (Agrawal and Shabana, 1986; Schwertassek et al., 1999b), and unlike the tangent frame, it is an actual floating frame of reference, not attached to any point of the body. The choice of the type of frame will condition how the elastic displacements are obtained and, in many cases, the results will also be affected (Escalona et al., 2002; Shabana, 1995). According to the reference conditions used, the motion is split into large amplitude motion and local deformation in different ways; in theory, the total motion should be the same no matter which type of frame is used, although in practice this is not true, due to the use of a reduced set of mode shapes.

The motion of the elastic body is completely defined by the reference and elastic coordinates but, in general, more points and unit vectors are needed to model the kinematic constraints of the mechanism. Following the natural coordinates formalism, these points and unit vectors are system variables, and they can be shared between neighbor bodies, which is a very simple method for introducing constraints. In Figure 2.2 this is illustrated through an example. A 3D beam with the local frame placed at one end, with the **u** vector aligned to its undeformed neutral axis, is connected to the adjacent body by means of a revolute joint placed at the opposite end, represented by shared entities: a point $P$ and a unit vector $\mathbf{v}_2$. Since the motion of the body is already

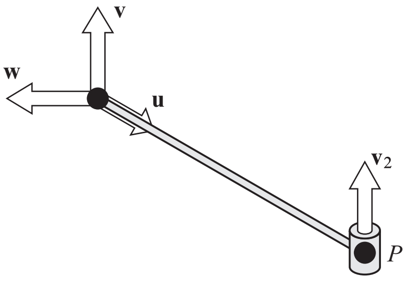


Figure 2.2: 3D beam with boundary point and vector at the tip.

represented by the reference and elastic coordinates, the additional point and vector can be considered as coordinates in excess, so that they are massless, and they must be connected to the elastic coordinates by means of kinematic constraints (Cuadrado et al., 2004c). In the method in relative coordinates, the position and orientation of the revolute joint are also needed for defining the recursive kinematic relations but, as will be seen later, they are not system variables, so no kinematic constraints are needed, which is one of the advantages of the formulation.

### 2.2.2 Component mode synthesis

As previously stated, the elastic displacement $\bar{\mathbf{r}}_f$ is a continuum field, which makes the use of an approximation technique necessary in order to make its computation practical. The most common method used for this purpose in FFR formulations is the Rayleigh–Ritz method, due to the fact that the elastic deformation is obtained in local coordinates with respect to a local frame, so that it can be linearized around the undeformed position for small deformations. The Ritz method allows to represent the deformed state of a solid with a minimum set of variables, by approximating it as a linear combination of constant deformation modes, whose coefficients are the so called modal amplitudes or elastic coordinates. The deformation modes, also known as mode shapes, are deformed configurations that can be obtained according to different criteria, in order to allow for modeling the displacement field to the required accuracy, while keeping their number as low as possible. Several types of mode shapes can be used, according to the criteria they are based on. Natural vibration modes or eigenmodes are calculated in order to keep the natural frequencies of the system. Other types of modes, such as the Krylov subspaces (Lehner and Eberhard, 2006), alone or combined with Gramian matrices (Lehner and Eberhard, 2007), are intended to keep the dynamic response within a frequency range. Static modes are obtained as the deformed shape under certain displacement or loading conditions. In the present work, only static modes and eigenmodes are used, although any constant deformed shapes can be used as long as they fulfill the reference conditions. There exist several ways to obtain the mode shapes of a flexible body. In simple structural elements such as beams, plates or shells, there may exist an analytical solution for many of them. In case no analytical functions are available, the mode shapes can be obtained by means of the finite element method, or even by performing an experimental modal analysis. These deformation modes are calculated in the local frame of reference, and they depend on the type of frame, i.e. the type of reference conditions.

In the present work, a Craig–Bampton reduction (Craig and Bampton, 1968) is used in combination with a tangent frame (Schwertassek et al., 1999b), although the modeling is easily generalizable for different types of floating frames and reduction methods. The Craig–Bampton reduction is especially designed for the modeling of interconnected bodies, through the use of static and dynamic modes. According to this method, the local elastic displacement can be approximated by means of a linear combination of $n_s$ static modes and $n_d$ dynamic modes, where each static mode $\mathbf{\Phi}_i$ is the displacement field that results from applying a unit displacement or rotation to a boundary degree of freedom, while keeping the remaining ones fixed, and the dynamic modes $\mathbf{\Psi}_j$ are normal eigenmodes calculated in a fixed undeformed interface configuration

$$\bar{\mathbf{r}}_f = \sum_{i=1}^{n_s} \mathbf{\Phi}_i \eta_i + \sum_{j=1}^{n_d} \mathbf{\Psi}_j \xi_j \tag{2.2}$$

where $\eta_i$ and $\xi_j$ are the static and dynamic modal amplitudes respectively, which are added as new coordinates of the multibody system. This expression can be written in matrix form

$$\bar{\mathbf{r}}_f = \begin{bmatrix} \mathbf{\Phi}_1 & \cdots & \mathbf{\Phi}_{ns} & \mathbf{\Psi}_1 & \cdots & \mathbf{\Psi}_{nd} \end{bmatrix} \begin{Bmatrix} \eta_1 \\ \vdots \\ \eta_{ns} \\ \xi_1 \\ \vdots \\ \xi_{nd} \end{Bmatrix} = \mathbf{X}\mathbf{y} \tag{2.3}$$

being $\mathbf{X}$ a matrix formed by the modes as columns, and $\mathbf{y}$ a vector containing all the modal amplitudes of the body, i.e. the elastic coordinates vector. This combination of static and fixed–interface dynamic modes can conform a set of quasi–comparison functions (Meirovitch and Kwak, 1990; Schwertassek et al., 1999b). This allows them for achieving better convergence than a set of admissible functions, such as normal eigenmodes, due to a better fulfillment of the dynamic boundary conditions. Moreover, the use of Craig–Bampton modes simplifies the kinematic constraints that link the elastic coordinates to the frame coordinates.

The static and dynamic modes are calculated within the local frame, thus obtaining the mode shapes expressed in local coordinates. In case a tangent frame is used, the body is considered to be clamped to the frame origin, which may or may not coincide with a kinematic pair, although, in order to reduce the number of coordinates of the system, it is more practical to use a kinematic pair to locate a tangent frame. By doing this, if a body has $n$ connection points, only $n - 1$ of these points are defining static modes. In Figure 2.3 a 3D beam is shown, with a tangent frame located at one end, and the six static modes defined by the degrees of freedom at the opposite end: unit axial translation along the **u** axis, unit bending translations along the **v** and **w** axes, and three unit rotations: torsion about the **u** axis, and bending due to unit rotation about the **v** and **w** axes. The dynamic modes are obtained as natural vibration modes,

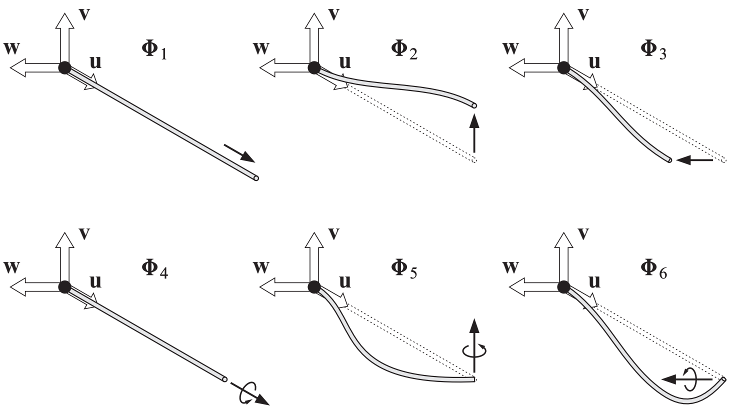


Figure 2.3: Static modes defined at the tip of a cantilever 3D beam.

with all the boundaries fixed. In the example of the beam, they can be obtained as the eigenmodes of a beam clamped at both ends. In Figure 2.4 two bending dynamic modes are shown, where it can be observed that there is neither rotation nor translation at any of the boundaries. This means that if a beam is deformed according to any of these modes, the boundaries will not be affected, so only the static modal amplitudes are used when the position of the kinematic pairs is calculated.

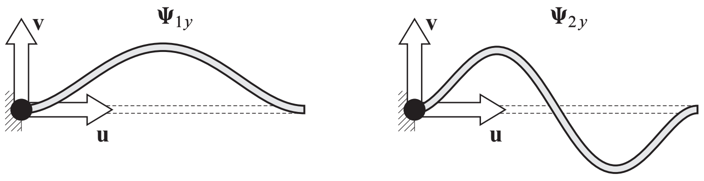


Figure 2.4: Dynamic modes in the local $xy$ plane of a cantilever beam.

The Craig–Bampton modes can be systematically calculated from the finite element mass and stiffness matrices, by imposing the adequate displacements to the boundary degrees of freedom, and solving the static and dynamic problems for the remaining internal displacements. In case a tangent frame is used, the rows and columns corresponding to the degrees of freedom of the clamped node(s) are eliminated, for being equal to zero. The remaining degrees of freedom can be divided into boundary $b$ and internal $i$, so that the finite element mass and stiffness matrices, along with the mode shapes matrix $\mathbf{X}^*$, can be partitioned into blocks accordingly,

$$\mathbf{M}^* = \begin{bmatrix} \mathbf{M}^*_{bb} & \mathbf{M}^*_{bi} \\ \mathbf{M}^*_{ib} & \mathbf{M}^*_{ii} \end{bmatrix}; \quad \mathbf{K}^* = \begin{bmatrix} \mathbf{K}^*_{bb} & \mathbf{K}^*_{bi} \\ \mathbf{K}^*_{ib} & \mathbf{K}^*_{ii} \end{bmatrix}; \quad \mathbf{X}^* = \begin{bmatrix} \mathbf{X}^*_{bs} & \mathbf{X}^*_{bd} \\ \mathbf{X}^*_{is} & \mathbf{X}^*_{id} \end{bmatrix} \tag{2.4}$$

The asterisk indicates nodal values, i.e. referred to a finite element discretization. In the mode shapes matrix $\mathbf{X}^*$, the column subindices $s$ and $d$ stand for static and dynamic modes, for the sake of clarity. The displacements of the boundary nodes are imposed in all the mode shapes, so that the blocks referred to boundary degrees of freedom, $\mathbf{X}^*_{bs}$ and $\mathbf{X}^*_{bd}$, can be substituted by their actual values. By definition, each static mode has a unit displacement at the corresponding DOF while keeping the remaining boundaries fixed, which means that the $\mathbf{X}^*_{bs}$ block is nothing but an identity matrix. Analogously, the dynamic modes have no boundary displacements, which is the same as stating that $\mathbf{X}^*_{bd}$ is equal to zero.

In order to calculate the static modes, a static equilibrium problem must be solved. The elastic equilibrium equation, which leads to the displacements of the internal nodes, can be written as a linear system with multiple right hand sides

$$\mathbf{K}^*\mathbf{X}^*_s = \mathbf{F}^* \tag{2.5}$$

The forces that would be needed to apply to the boundaries $\mathbf{F}^*_b$ are unknowns, whereas there are no applied forces to the internal nodes. Therefore, the system can be written in partitioned form, after substituting the known values of forces and displacements

$$\begin{bmatrix} \mathbf{K}^*_{bb} & \mathbf{K}^*_{bi} \\ \mathbf{K}^*_{ib} & \mathbf{K}^*_{ii} \end{bmatrix} \begin{bmatrix} \mathbf{I} \\ \mathbf{X}^*_{is} \end{bmatrix} = \begin{bmatrix} \mathbf{F}^*_b \\ 0 \end{bmatrix} \tag{2.6}$$

The solution of the equations corresponding to the internal nodes yields the internal displacements, so that the static modes can be directly obtained

$$\mathbf{K}^*_{ib} + \mathbf{K}^*_{ii}\mathbf{X}^*_{is} = 0 \implies \mathbf{X}^*_s = \begin{bmatrix} \mathbf{I} \\ -\left(\mathbf{K}^*_{ii}\right)^{-1}\mathbf{K}^*_{ib} \end{bmatrix} \tag{2.7}$$

The internal displacements of the dynamic modes, $\mathbf{X}^*_{id}$, which are the only unknowns they have, are obtained as a solution of a generalized eigenvalues problem stated only for the internal nodes

$$\left(\mathbf{K}^*_{ii} - \Omega^2\mathbf{M}^*_{ii}\right)\mathbf{X}^*_{id} = 0 \implies \mathbf{X}^*_d = \begin{bmatrix} 0 \\ \mathbf{X}^*_{id} \end{bmatrix} \tag{2.8}$$

The type of kinematic pairs which connect the flexible body to the rest of the system affect the boundary conditions used for the calculation of the mode shapes. In

Figure 2.3, the static modes are calculated by imposing three translations and three rotations to the tip of the beam, in order to account for the deformation produced by forces or moments in the corresponding directions. In most cases, the connection to the adjacent body is such that in one or more of these directions, degrees of freedom exist, which means that no force or moment will be transmitted to the body. The static modes are then not strictly obtained as described before. Each degree of freedom added to a boundary kinematic pair, will eliminate the corresponding static mode, and the finite element DOF in that direction becomes an internal one. In order to illustrate this, the example shown in Figure 2.2 is used. If the beam has a revolute joint at the tip, with its axis oriented in the **v** direction, the static mode $\mathbf{\Phi}_5$ shown in Figure 2.3 no longer exists, since no moment in that direction is transmitted at the joint. Accordingly, $\mathbf{\Phi}_3$ is calculated with the rotation in the **v** direction allowed, taking the shape shown in Figure 2.5. The same happens to the dynamic modes. In Figure 2.6, the first dynamic

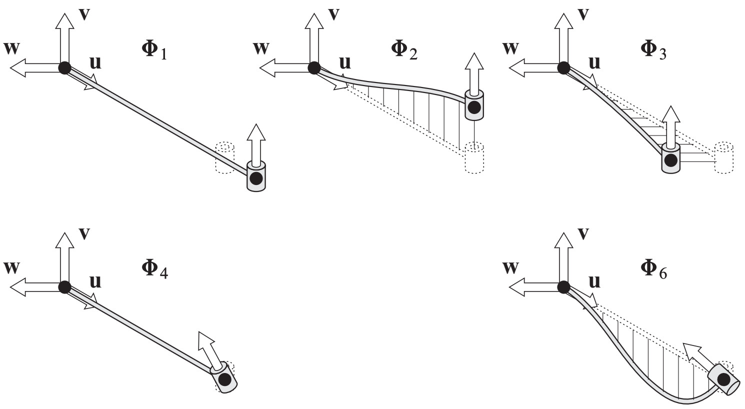


Figure 2.5: Static modes at the tip of a beam with revolute joint.

modes in both the **v** and **w** directions are shown, where it can be observed that the mode in the **w** direction has the rotation along the revolute axis allowed. The two static modes $\mathbf{\Phi}_2$ and $\mathbf{\Phi}_3$ are also shown, in order to see more clearly the difference between them. In short, the modes are calculated considering the beam as clamped in

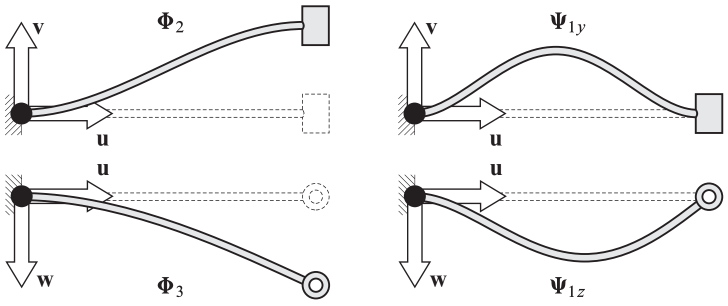


Figure 2.6: Static bending mode with fixed and free tip rotation.

the **uv** plane, while the conditions used for the **uw** plane are clamped–pinned. This is easily generalizable to any kind of kinematic joint: each degree of freedom in the multibody model eliminates one static mode from the elastic body, and makes the corresponding finite element DOF become internal.

### Arbitrarily oriented boundaries

In the general case, the degrees of freedom of a joint are not aligned to the principal directions of the local frame of reference. In order to calculate the mode shapes correctly, a local frame of reference must be defined at the joint $i$, located at point $P$, defined by a local orientation matrix $\bar{\mathbf{A}}_{ui}^{P}$, where the $u$ subindex stands for *undeformed*. The degrees of freedom of the joint must be defined along this joint frame prior to eliminating the corresponding static modes and solving for the internal nodes. As an example, the calculation of the mode shapes of a 2D cantilever beam with a slider joint whose direction is rotated an angle $\theta$ with respect to the **v** vector of the frame of reference, as shown in Figure 2.7, is described. The local frame of reference,

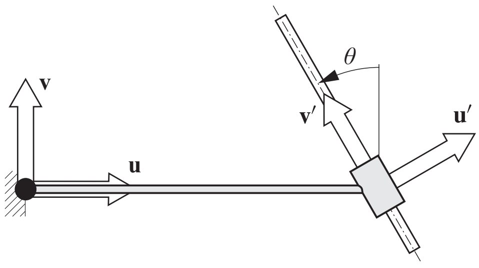


Figure 2.7: 2D cantilever beam with arbitrarily oriented slider joint.

fixed to the kinematic joint, is defined by vectors $\mathbf{u}'$ and $\mathbf{v}'$, which are rotated an angle $\theta$ with respect to the original frame of reference. There exists a degree of freedom in the $\mathbf{v}'$ direction, so that the beam must have motion allowed in that direction when calculating both the static and the dynamic mode shapes. In order to do that, the degrees of freedom of the corresponding node can be rotated by means of a transformation matrix $\mathbf{T}$. Let $\mathbf{M}^*$, $\mathbf{K}^*$ and $\mathbf{X}^*$ be the mass, stiffness and mode shapes matrices obtained with all the degrees of freedom aligned to the local frame. The transformed matrices $\mathbf{M}_r^*$, $\mathbf{K}_r^*$ and $\mathbf{X}_r^*$, with the end node rotated to the $\mathbf{u}'\mathbf{v}'$ base, are obtained as

$$\mathbf{M}_r^* = \mathbf{T}^{\mathsf{T}}\mathbf{M}^*\mathbf{T}; \quad \mathbf{K}_r^* = \mathbf{T}^{\mathsf{T}}\mathbf{K}^*\mathbf{T}; \quad \mathbf{X}^* = \mathbf{T}\mathbf{X}_r^* \tag{2.9}$$

The transformation matrix $\mathbf{T}$ is an identity matrix, containing $\bar{\mathbf{A}}_{ui}^{P}$ at the diagonal block corresponding to the affected node. If structural finite elements are used, the finite element nodes have six coordinates, so that two $\bar{\mathbf{A}}_{ui}^{P}$ blocks must be placed. According to the new orientation of the degrees of freedom, there will still be three static modes: two unit displacements in the $\mathbf{u}'$ and $\mathbf{v}'$ directions, and a unit in–plane rotation. However, the row and column corresponding to the displacement along $\mathbf{v}'$ can now be moved into the set of internal degrees of freedom, thus calculating the modes in the normal way.

**Considerations about isoparametric finite elements**

In case that isoparametric finite elements are used, with only translation degrees of freedom at their nodes, some precautions should be taken when calculating the mode shapes. The problem comes from the fact that a node with no rotations cannot be clamped, but only pinned, thus still allowing rigid body rotations that might render the system matrix singular. For instance, a beam can be totally clamped if the six degrees of freedom are locked at a specific node, but a solid discretized into 4–node tetrahedrons cannot, since a node has only three degrees of freedom. Even if six degrees of freedom are locked, for example by fixing two nodes, there could still exist rotation about the axis defined by them. In general, when only translations are used as nodal degrees of freedom, it is better to apply the reference conditions to lines in the plane case, or surfaces in the three–dimensional case, in order to avoid undesired rigid body modes.

## 2.2.3 Kinematics of boundary points and joint frames

Provided that the position and velocity of a flexible body, i.e. the position and orientation of its local floating frame of reference, are known, the first step to be carried out in order to apply the kinematic relations is the calculation of the position and velocity of its boundary points and their corresponding joint frames. Each boundary can undergo a maximum of three translations and three rotations, which can be immediately obtained from the amplitudes of its static modes if they are defined as unit displacements, however in the general case one or more of these possible motions will not appear due to the joint degrees of freedom, as has been pointed out when the modal reduction has been described. Only in case a bracket joint is considered, such as those used in substructuring techniques, the six static modes will appear. These amplitudes can be grouped into a vector $\boldsymbol{\eta}^P$ that contains the translational $\boldsymbol{\eta}_t^P$ and rotational $\boldsymbol{\eta}_\theta^P$ modal amplitudes, which will be defined in the frame of reference associated to the joint.

The local position $\bar{\mathbf{r}}^P$ of a boundary point $P$ is the sum of its undeformed position $\bar{\mathbf{r}}_u^P$ in local coordinates, plus the elastic displacement $\bar{\mathbf{r}}_f^P$. The local elastic displacement is approximated by a modal superposition, which, if evaluated at the specific boundary node $P$, results to be equal to $\bar{\mathbf{A}}_u^P \boldsymbol{\eta}_t^P$, as long as the static modes are defined as unit displacements along the principal directions of the undeformed joint frame, namely $\mathbf{u}'$ and $\mathbf{v}'$ in Figure 2.8. According to this, the local position of $P$ can be obtained as

$$\bar{\mathbf{r}}^P = \bar{\mathbf{r}}_u^P + \bar{\mathbf{A}}_u^P \boldsymbol{\eta}_t^p = \bar{\mathbf{r}}_u^P + \bar{\mathbf{A}}_u^P \begin{Bmatrix} \eta_1 \\ \eta_2 \\ \eta_3 \end{Bmatrix} \tag{2.10}$$

where $\bar{\mathbf{A}}_u^P$ is the orientation matrix of the joint in local coordinates, already used for

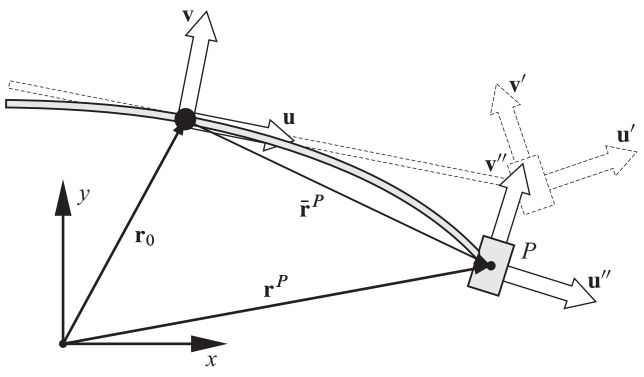


Figure 2.8: Deformed joint frame of reference.

obtaining the **T** matrix in Eq. (2.9). This matrix contains, in the 2D example of Figure 2.8, the $\mathbf{u}'$ and $\mathbf{v}'$ vectors as columns, expressed in the **u**, **v** base; the choice of a two–dimensional representation is made in order to simplify the figures, but the mathematical expressions will be all developed for a 3D general case, so there is no loss of generality. The position of point $P$ in global coordinates will finally be

$$\mathbf{r}^P = \mathbf{r}_0 + \mathbf{A}\bar{\mathbf{r}}^P = \mathbf{r}_0 + \mathbf{A}\left(\bar{\mathbf{r}}_u^P + \bar{\mathbf{A}}_u^P \boldsymbol{\eta}_t^p\right) \tag{2.11}$$

As shown in Figure 2.8, the final orientation of a joint frame in local coordinates, $\bar{\mathbf{A}}^P$, represented by vectors $\mathbf{u}''$ and $\mathbf{v}''$, is obtained after two transformations. The first transformation is defined by the constant matrix $\bar{\mathbf{A}}_u^P$, and, as has been previously pointed out, it is only needed in case the joint axis $\mathbf{v}'$ is not parallel to any of the principal axes of the local frame of the body. Then, a second transformation is applied to calculate the deformed orientation of the joint, and it is defined by an infinitesimal rotation matrix $\mathbf{I}_3 + \tilde{\boldsymbol{\eta}}_\theta^P$, which is itself expressed in the joint frame since the rotations are considered about the joint axes. The matrix $\tilde{\boldsymbol{\eta}}_\theta^P$ is the skew–symmetric matrix associated to the vector of rotation modal amplitudes $\boldsymbol{\eta}_\theta^P$, which are assumed to be small, so that $\cos\eta \approx 1$ and $\sin\eta \approx \eta$,

$$\bar{\mathbf{A}}^P = \bar{\mathbf{A}}_u^P\left(\mathbf{I}_3 + \tilde{\boldsymbol{\eta}}_\theta^P\right) = \bar{\mathbf{A}}_u^P \begin{bmatrix} 1 & -\eta_6 & \eta_5 \\ \eta_6 & 1 & -\eta_4 \\ -\eta_5 & \eta_4 & 1 \end{bmatrix} \tag{2.12}$$

By rotating this deformed local system into the local frame of the body, the absolute orientation of the joint frame results

$$\mathbf{A}^P = \mathbf{A}\bar{\mathbf{A}}^P = \mathbf{A}\bar{\mathbf{A}}_u^P\left(\mathbf{I}_3 + \tilde{\boldsymbol{\eta}}_\theta^P\right) \tag{2.13}$$

By using all the previous expressions, the absolute position $\mathbf{r}^P$ and orientation $\mathbf{A}^P$ of a joint frame can be derived from the variables of the body it is attached to. When performing the forward position analysis, the opposite situation may arise: the data of an input joint is known, and the objective is to calculate the parameters of the corresponding body. In this case, the position of the local frame of the body is obtained by following the inverse procedure. First, the orientation is obtained by performing the inverse transformation of Eq. (2.13),

$$\mathbf{A} = \mathbf{A}^P \bar{\mathbf{A}}^{P\top} = \mathbf{A}^P \left(\mathbf{I}_3 - \tilde{\boldsymbol{\eta}}_\theta^P\right) \mathbf{A}_u^{P\top} \tag{2.14}$$

and this allows for obtaining the position of the origin of the local frame, by isolating it in Eq. (2.11),

$$\mathbf{r}_0 = \mathbf{r}^P - \mathbf{A}\left(\bar{\mathbf{r}}_u^P + \bar{\mathbf{A}}_u^P \boldsymbol{\eta}_t^p\right) \tag{2.15}$$

The need for evaluating these inverse relations can be avoided if the local frame of the body is placed at its input joint, as a tangent frame. In such a case, the local frame of the input joint directly coincides with the body reference frame, thus simplifying the kinematic relations. Additionally, the use of this kind of frame reduces the number of static modes, due to the fact that the boundaries at the input joint are fixed by definition. However, the use of a tangent frame can have some drawbacks, which have been studied by many authors such as Shabana (1995), Schwertassek et al. (1999a,b), or Escalona et al. (2002). Firstly, it leads to larger elastic displacements with respect to the undeformed mean axis, as can be appreciated in Figure 2.9. On the left, a deformed

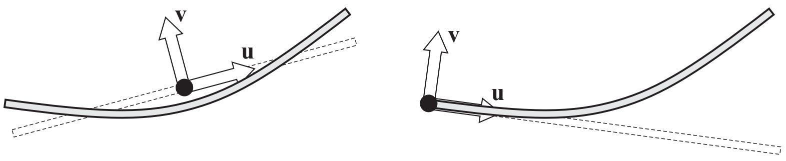


Figure 2.9: Buckens versus tangent reference frame.

beam with a Buckens frame is shown, where it can be seen that the mean elastic displacement is smaller than on the right, where a tangent frame is attached to the same beam. This also means that, when a tangent frame is used, there will exist a stronger coupling between the reference and elastic motions, thus reducing the sparsity of the mass matrix. Another drawback of the tangent frame can be the loss of symmetry, which can introduce spurious stresses if the number of deformation modes is low.

At this point, the position and orientation of any joint frame can be calculated from those of the corresponding body, and vice–versa. The remaining step needed to complete the forward position analysis is the calculation of the parameters of an input frame from those of the output frame of the preceding body. These relations, since they are established between joint frames, which are considered as locally rigid, can be derived from rigid body mechanics.

## 2.3 Recursive kinematics in open–loop systems

### 2.3.1 Opening of closed loops

As already pointed out at the introduction of this chapter, the formulation here proposed uses relative coordinates for modeling the large amplitude or *reference* motion. These coordinates are defined as relative displacements and rotations between a body and its predecessor in the kinematic chain, meaning that the position of a body is not completely determined by its own coordinates, but those of the preceding one are also required. According to this, in order to obtain the absolute position of a body $j$, it must be first calculated for all the preceding bodies, by performing a forward loop from the first body of the mechanism (root or base body) to body $j$. This is done through *recursive relations*, that express the position of a body $j$ as a function of that of its predecessor $i$, and the relative coordinates between them. In order to perform a velocity or acceleration analysis, i.e. calculate the velocities and accelerations of the bodies from the first and second time derivatives of the relative coordinates, the corresponding recursive relations must be also established. In order to define the relative coordinates and establish the recursive relations, the closed loops must be cut to obtain a open–loop or spanning–tree mechanism. This open–loop mechanism is characterized by a set of relative dependent coordinates, to which a set of kinematic constraints are later applied in order to enforce the closure of the cut joints.

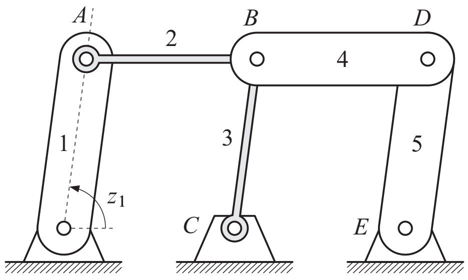


Figure 2.10: Closed–loop mechanism.

An example mechanism will be used along this chapter to illustrate the formulation. The mechanism, shown in Figure 2.10, consists of a planar double four–bar mechanism, in which only bodies 2 and 3 are flexible. The rigid version of this mechanism has one degree of freedom, represented by the $z_1$ angle. In case a body is considered as flexible, more degrees of freedom are added, since each deformation mode, let it be static or dynamic, adds one degree of freedom to the system. In what follows, the degrees of freedom corresponding to kinematic pairs, which are associated to the rigid body or large amplitude motion, will be referred to as *reference* degrees of freedom, as opposed to the *flexible* degrees of freedom due to elasticity of the bodies. One possible open–loop configuration for the example mechanism is shown in Figure 2.11, obtained as a result of cutting the joints at $C$ and $E$.

This open–loop mechanism has five reference degrees of freedom, plus the even-

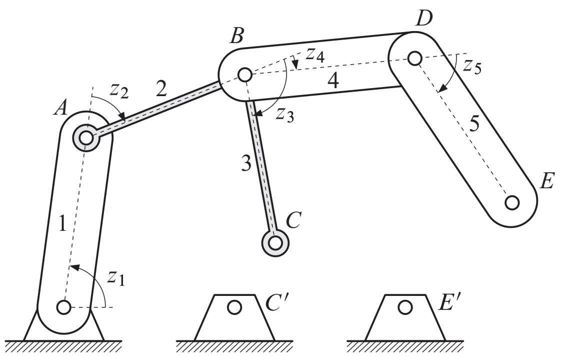


Figure 2.11: Open–loop mechanism.

tual modal amplitudes, so that four kinematic constraints must be added in order to link points $C$ and $E$ to their respective ground attachments, thus eliminating four of the five reference degrees of freedom.

Having an open–loop structure enables to perform the position and velocity analyses in a recursive forward loop, and then to accumulate forces and inertias from the leaves to the root in a recursive backward loop, which would not be possible in a closed–loop mechanism. In the open–loop version of the system, each body $j$ is connected to its predecessor in the kinematic chain by a joint $j$, which will be considered as its *input* joint, and can be followed, if it is not the end of the tree, by one or more bodies connected through *output* joints. In the example, body 2 has an input joint with a relative angle $z_2$, and two output joints 3 and 4, which are in turn the input joints of bodies 3 and 4 respectively, placed at the same point $B$. In order to make the kinematic relationships clearer, bodies are numbered from the root to the leaves, in such a way that if body $i$ precedes body $j$ in the kinematic chain, immediately or not, it never can happen that $i > j$; the same convention is applied to the points where static modes are defined.

The choice of which joints to cut is not unique, and can affect both the accuracy of the solution and the complexity of the resulting model. The accuracy can be affected especially in long branches, because of the numerical round–off error accumulated from the root to the leaves, and the solution in Figure 2.11 is not optimal from this point of view, since the position of body 5 depends on four coordinates; cutting the two joints at $B$ would lead to three shorter branches, meaning that the resulting system is less prone to numerical error. In what regards the complexity of the model, although not seen in the example, the type of joints being cut is of great importance; in the general 3D case, for instance, cutting a spherical joint would eliminate three relative coordinates, asking for three additional kinematic constraints, whereas doing so to a cylindrical joint would eliminate one coordinate and add five constraints, which is obviously a worse choice.

In this case, five relative coordinates along with the modal amplitudes are needed to correctly position all the bodies of the mechanism. In natural coordinates, if the ground attachments are not considered as system variables, the system would have six coordinates, i.e. the positions of $A$, $B$ and $D$, so that using relative coordinates would only reduce the size of the problem in one variable. In this example, therefore, no efficiency improvement is expectable in the rigid case from the use of relative coordinates, since the reduction of the problem size does not compensate for the higher complexity of the formulation, but as will be seen later, improvement appears as the number of variables is increased.

### 2.3.2 Modeling of the kinematic joints

There exist two basic types of kinematic joints, on which all the others are based: translation along a straight line, and rotation about an axis. In both cases, the relative motion is characterized by a principal axis, which can be represented by a point $P$ and a unit vector $\mathbf{v}$, expressed in global Cartesian coordinates. As pointed out when addressing the modeling of flexible bodies, as long as the axis of a joint is not parallel to any of the principal directions of the local frame, a joint frame must be defined at its position, in such a way that one of its three principal directions is coincident to that of the joint. In the formulation in natural coordinates, the unit vector is directly used for defining the kinematic pair and its three components are system variables, but in relative coordinates a complete frame is needed in order to correctly position the next body in the chain, although the joint frame is used as an intermediate reference and does not add any variable to the system.

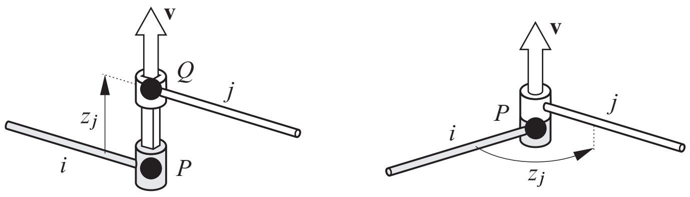


Figure 2.12: Basic translational and revolute joints.

These two basic joints are shown in Figure 2.12. The translational joint, which can be seen in the left side of the figure, consists of a translation along an axis that passes through point $P$, and whose direction is defined by vector $\mathbf{v}$, in such a way that point $Q$ of body $j$ is placed at a distance from $P$ equal to $z_j$ along the $\mathbf{v}$ axis. The revolute joint, shown in the right side, is a rotation about an axis, also defined by a point and a unit vector, so that the orientation of the joint frame, considered as pertaining to body $j$, is obtained after applying a rotation of value $z_j$ about the revolute axis $\mathbf{v}$.

These are the basic one degree of freedom joints, but, in the general case, other types of joints with more than one degree of freedom can appear, such as cylindrical, spherical, universal, etc. These joints can be considered as a combination of translations and rotations, being each of them undergone by intermediate virtual bodies with zero mass and length. In Figure 2.13, it can be seen an example of how a cylindrical joint, connecting two bodies $i$ and $k$, is derived from a translational and a revolute joint. This is a two–degree–of–freedom joint, in which the translation and the rotation share the same axis. Therefore, it can be modeled by introducing an intermediate

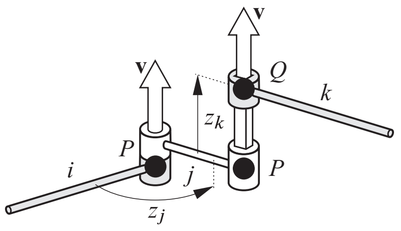


Figure 2.13: Cylindrical joint as a combination of a translational and a revolute joint.

zero–length body $j$ between the two actual bodies, in such a way that the rotation is defined between bodies $i$ and $j$, and the translation is afterwards applied between bodies $j$ and $k$. This is easily generalizable to other types of kinematic joints, and it can even be used to model the relative motion between two bodies that are not interconnected, or the motion of a free body with respect to the inertial frame of reference, by means of the so called *floating joint* or *six–degree–of–freedom joint* (Wittenburg, 1977). A floating joint can be considered as a combination of three translational plus three revolute joints, each of them acting between two bodies, so that five intermediate virtual massless bodies should be added.

When a simulation is carried out in relative coordinates, the first step is the solution, from a known set of relative coordinates and velocities, of the position and velocity problems in a forward recursive loop. This is achieved by applying recursive position and velocity relations, going from the root to the leaves, in order to obtain the position and velocity of all the bodies and kinematic pairs. These positions and velocities are expressed in natural coordinates, as points and unit vectors, which are only used as an intermediate coordinate set. Two kinds of recursive relations can be considered at this point: the internal relations between points of a body, characterized by the flexible relative coordinates, and the relative motion at the joints, modeled by the reference coordinates.

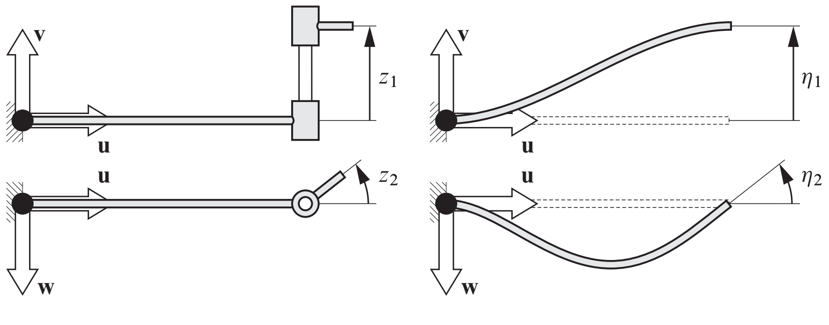


Figure 2.14: Analogy between relative coordinates and static modal amplitudes.

The Craig–Bampton reduction is very well suited for using in combination with this formulation. The dynamic modes are calculated with all the boundaries fixed, which means that they do not affect their relative positions, and this implies that they do not affect the kinematic relations. And the static modes are defined as basic translations and rotations, which affect only one position or orientation parameter, so that they behave exactly like basic kinematic joints. This is illustrated with an example in Figure 2.14, where a translational and a revolute joint are shown on the left, in this case with their axes coincident with those of the local frame of the body. On the right, the analogy to the translation and rotation static modes is shown. If the modes are defined as unit displacements or rotations, the value of the modal amplitude will be the actual value of the displacement or rotation at that point, in the local coordinates of the joint, which is very convenient for establishing the recursive relations.

## 2.3.3 Kinematic relations for the forward position analysis

The forward position analysis consists of obtaining, from a given set of relative coordinates, the Cartesian positions of all the relevant points and unit vectors of the system, i.e. all those implied in kinematic pairs or constraints, force elements, etc. This is achieved by following a recursive procedure, going from the root or base body to the leaves, in such a way that the position of a body is needed to obtain that of the following one in the chain.

The position relations are different depending on the type of kinematic joint. However, differences only exist in the relation between the local frame at an output joint and the corresponding input frame of the next body, which is where the kinematic joint is actually defined. In order to avoid excessive repetition, the common characteristics of both joints are to be described first, being the specific relations addressed later. A generic joint can be defined as a relative motion between two consecutive bodies $i$ and $j$, in such a way that the relative coordinate $z_j$ acts between an output point $P$ of the first body, and an input point $Q$ of the second one. This relative motion can be either a translation, as shown in Figure 2.15, or a rotation as in Figure 2.16. The sequence for obtaining the position and orientation of body $j$ from those of body $i$, regardless of the type of joint, is the following:

- The deformed local position $\bar{\mathbf{r}}_i^P$ and orientation $\bar{\mathbf{A}}_i^P$ of a joint frame within body $i$ are obtained by means of Eqs. (2.10) and (2.12).

- They are substituted, along with the body position $\mathbf{r}_i$ and rotation matrix $\mathbf{A}_i$, into Eqs. (2.11) and (2.13), in order to obtain their absolute counterparts $\mathbf{r}_i^P$ and $\mathbf{A}_i^P$.

- The relative displacement or rotation $z_j$ yields the absolute parameters of the input frame of body $j$, namely $\mathbf{r}_j^Q$ and $\mathbf{A}_j^Q$. This is the only step that is specific to the type of kinematic joint.

- The values of $\bar{\mathbf{r}}_j^Q$ and $\bar{\mathbf{A}}_j^Q$ are obtained as done for body $i$ in the first step.

- By applying the inverse relations (2.14) and (2.15), the absolute parameters of the floating frame of the second body, $\mathbf{r}_j$ and $\mathbf{A}_j$, can be finally determined.

The only undetermined step is the third one, and it will be detailed next, for both the translational and the revolute joints.

### Translational joint

A 2D translational joint is shown in Figure 2.15, acting between two bodies $i$ and $j$. All the development, as done when addressing the elastic coordinates, will be carried out for a general spatial joint, so that there is no loss of generality due to the use of 2D figures.

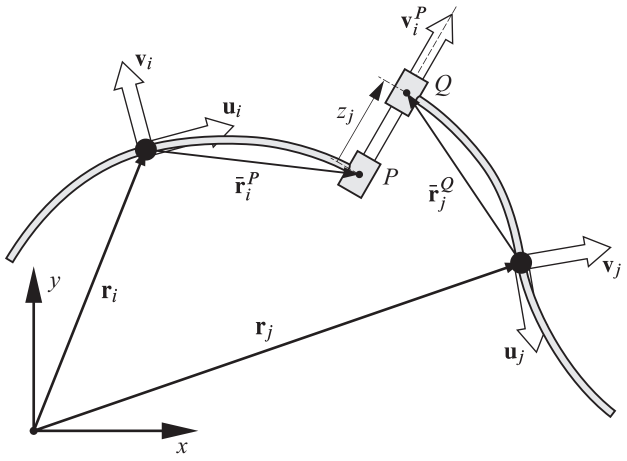


Figure 2.15: Planar translational joint.

In the case of a translational joint, the relative motion between both frames is determined by the value of the relative coordinate $z_j$, which represents a translation of point $Q$ with respect to $P$ along a direction defined by a unit vector $\mathbf{v}_i^P$, so that the position of $Q$ is obtained as

$$\mathbf{r}_j^Q = \mathbf{r}_i^P + z_j \mathbf{v}_i^P \tag{2.16}$$

If the joint frame at $P$ is defined, as suggested in the component mode synthesis section, such that the relative translation axis coincides with one of its three principal directions, the unit vector $\mathbf{v}_i^P$ will be directly the corresponding column in $\mathbf{A}_i^P$. In what regards the orientation of the frames, the translational joint does not introduce any relative rotation, so that

$$\mathbf{A}_j^Q = \mathbf{A}_i^P \tag{2.17}$$

thus completely defining the position and orientation of the frame at $Q$, from that at $P$ and the relative coordinate $z_j$.

### Revolute joint

A planar revolute joint can be seen in Figure 2.16. In this case, points $P$ and $Q$ are coincident, and the axis that defines the rotation is the vector $\mathbf{w}_i^P$, perpendicular to the plane of the figure.

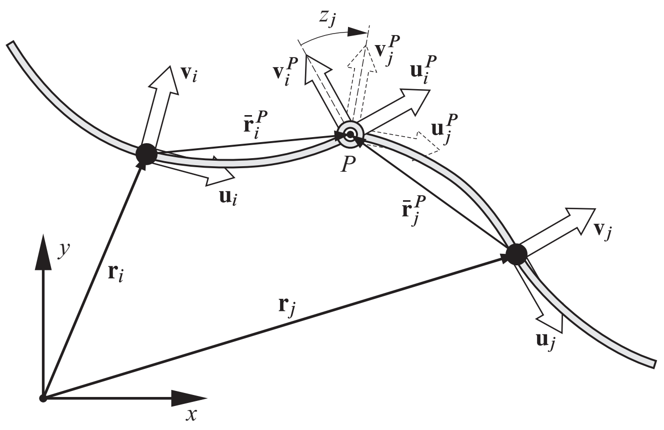


Figure 2.16: Planar revolute joint.

Being the point $P$ the same for both bodies, the only thing that must be determined is the orientation of the input frame at point $P$, considered as pertaining to body $j$. It can be obtained by means of a generic rotation matrix $\mathbf{A}_\mathbf{w}\left(z_j\right)$,

$$\mathbf{A}_j^P = \mathbf{A}_i^P \mathbf{A}_\mathbf{w}\left(z_j\right) \tag{2.18}$$

A finite rotation $\phi$ about a generic unit vector $\mathbf{u}$, defined in the local coordinates of the joint frame at $P$, can be obtained by means of the Rodrigues' formula (Wittenburg, 1977),

$$\mathbf{A}_\mathbf{u}(\phi) = \mathbf{I}_3 + \tilde{\mathbf{u}}\sin\phi + \tilde{\mathbf{u}}\tilde{\mathbf{u}}\,(1-\cos\phi) \tag{2.19}$$

In the example, since the vector $\mathbf{w}_i^P$, expressed in the local frame at $P$, is the third vector of the canonical base, the rotation matrix results,

$$\mathbf{A}_\mathbf{w}\left(z_j\right) = \begin{bmatrix} \cos z_j & -\sin z_j & 0 \\ \sin z_j & \cos z_j & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.20}$$

When a static mode corresponding to a revolute joint is removed from the finite element model of a flexible body, an important side effect can appear. Due to the lack of a static mode defining a rotation about the joint axis, the joint frame obtained after the deformation will not reflect the actual orientation of the joint. This can be seen in Figure 2.17, where a revolute joint with the rotation mode eliminated from bodies $i$ and $j$ is shown. As can be seen in the figure, after the first body is deformed, point $P$ suffers a displacement $\boldsymbol{\eta}_{ti}^P$, but the joint frame remains parallel to the body frame, and the same happens to the second body. Since the joint frames are no longer perpendicular to the neutral axis of the beams, the angle $z_j$ will not be the actual angle rotated at the joint. In some cases, when the joint is allowed free motion, it can be justified to eliminate the mode for the sake of efficiency, but in case the angle value is

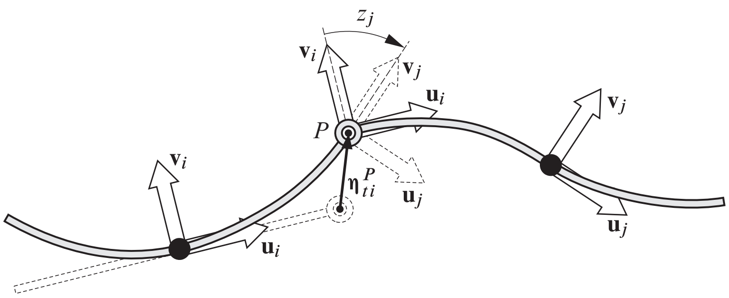


Figure 2.17: Effect of eliminating rotation modes at revolute joints.

required because of control demand, or due to the presence of an actuator, a rotational spring, etc., the rotation mode can not be eliminated.

**Fixed rigid–body rotations**

Apart from the possible rotations introduced by revolute joints, a fixed relative rotation might exist between two consecutive frame joints. For instance, in the case of a revolute joint, both frames can be defined to be coincident for $\mathbf{z}_j = 0$, but there can also exist a fixed offset for that value of the coordinate. Analogously, in the translational joint shown in Figure 2.15, the $\mathbf{u}_j^Q$ and $\mathbf{w}_j^Q$ vectors might be rotated a fixed amount $\phi$ about the $\mathbf{v}_i^P$ direction. Any of these possibilities can be considered by means of a constant rotation matrix, which can be obtained in the general case by using the Rodrigues' formula. In such a case, the total rotation at a translational joint shown in Eq. (2.17) becomes

$$\mathbf{A}_j^Q = \mathbf{A}_i^P \mathbf{A}_{\mathbf{v}} (\phi) \tag{2.21}$$

and the same happens to Eq. (2.18) for a revolute joint

$$\mathbf{A}_j^P = \mathbf{A}_i^P \mathbf{A}_{\mathbf{w}} (\phi) \, \mathbf{A}_{\mathbf{w}} \left(z_j\right) \tag{2.22}$$

Another case in which these fixed rotations might be needed is when the direction of the joint does not correspond to the same axis in both frames, due to differences in how the local frames of the finite element models are chosen. For example, the rotation axis can be defined as the $\mathbf{u}_i^P$ vector of the first body, but coinciding with the $\mathbf{v}_j^Q$ vector of the second one. In such a case, the necessary $\pi$ or $\pm\pi/2$ rotations must be also performed at this point.

## 2.3.4 Recursive relations for velocities and accelerations

When relative coordinates are used, the calculation of the inertia parameters associated to them is anything but straightforward. The solution, in the general case, is to first calculate them in terms of an intermediate set of Cartesian coordinates $\mathbf{Z}$, defined at velocity level for each body, in such a way that they can be afterwards projected into the relative coordinates $\mathbf{z}$ by means of a velocity transformation. In the present subsection, recursive relations between the Cartesian velocities of consecutive bodies are to be established, thus enabling to define the velocity transformation in an efficient way. In order to calculate the time derivative of the velocity transformation, needed for obtaining the velocity dependent inertia forces, these relations are also obtained at acceleration level. Firstly, the recursive relations for rigid bodies are derived for the joint frames, and then they will be extended to include the effect of static modal amplitudes.

The intermediate Cartesian coordinates can be divided into reference velocities $\mathbf{Z}_r$, representing the motion of the floating frame of reference, and elastic velocities $\mathbf{Z}_f$, which will be nothing but the derivatives of the modal amplitudes $\dot{\mathbf{y}}$. The reference section is in turn divided into translational and angular velocities, $\mathbf{Z}_t$ and $\mathbf{Z}_\theta$. The translational reference velocity $\mathbf{Z}_t$ used here, as pointed out in the introduction to this chapter, is chosen following the approach of Jiménez (1993), i.e. the velocity of the point of the body which instantly coincides with the origin of the global frame of reference, considering the point as rigidly attached to the local frame of the body. The angular velocity $\mathbf{Z}_\theta$ is represented by the instant angular velocity vector $\boldsymbol{\omega}$, so that the complete set of Cartesian velocities that characterize the velocity field of a flexible body is

$$\mathbf{Z} = \begin{Bmatrix} \mathbf{Z}_t \\ \mathbf{Z}_\theta \\ \mathbf{Z}_f \end{Bmatrix} = \begin{Bmatrix} \dot{\mathbf{s}} \\ \boldsymbol{\omega} \\ \dot{\mathbf{y}} \end{Bmatrix} \tag{2.23}$$

The use of these coordinates features some advantages when deriving the final equations of motion in relative coordinates. On the one hand, the rotary inertias of all the bodies are obtained with respect to the same point, so that the Cartesian mass matrices can be directly accumulated without any further transformation. On the other hand, the recursive relations derived for these coordinates are simpler than those obtained when the center of mass is used as the point of reference. The only drawback of using the global origin is the increased complexity of the resulting inertia terms in Cartesian coordinates.

The recursive relations consist of expressing the velocity of a body $j$, more specifically its reference part $\mathbf{Z}_{rj}$, as a function of that of the preceding body, $\mathbf{Z}_{ri}$, and the relative velocity produced at the joint between them. The relative coordinate at the joint $j$ is named $z_j$, and depending on the joint type, it will be the translation along or the rotation about the direction defined by the joint vector. In what follows, without any loss of generality, the vector defining the axis of the joint will be generically named $\mathbf{v}_i^P$. The objective is to find a relation between the velocities of two consecutive bodies of the form

$$\mathbf{Z}_{rj} = \mathbf{Z}_{ri} + \mathbf{b}_j \dot{z}_j \tag{2.24}$$

where $\mathbf{b}_j$ depends only on the type and position parameters of the joint, namely $\mathbf{r}_j^P$

and $\mathbf{v}_j^P$. In order to establish recursive relations for the accelerations as well, these velocity relations can be differentiated with respect to time

$$\dot{\mathbf{Z}}_{rj} = \dot{\mathbf{Z}}_{ri} + \mathbf{b}_j \ddot{z}_j + \dot{\mathbf{b}}_j \dot{z}_j = \dot{\mathbf{Z}}_{ri} + \mathbf{b}_j \ddot{z}_j + \mathbf{d}_j \tag{2.25}$$

where the term $\mathbf{d}_j$ depends on the relative velocity $\dot{z}_j$, and on the position and velocity parameters of the joint frame. The calculation of these recursive terms is performed in a two–stage process. First, a forward position analysis is carried out, in order to calculate all the $\mathbf{b}_j$ terms, which depend only on the position. Once they are obtained, a velocity analysis will enable to construct the $\mathbf{d}_j$ terms. These two steps can be carried out in parallel: from the position parameters of a body, those of the output joint are obtained, which in turn allow for calculating the corresponding $\mathbf{b}$ term; then, the velocity of the joint frame is obtained, finally leading to the $\mathbf{d}$ term, before moving on to the next body in the kinematic chain. If a tree structure with multiple branches is present, the forward analysis of the different branches can be carried out in parallel processes.

These recursive relations, as opposed to what happens in rigid systems, are not actually defined between bodies, but between reference frames, which are themselves considered as rigid bodies. There will exist, then, two types of such relations: the relation between two consecutive joint frames, i.e. an actual kinematic joint, and the relative motion between the frame of a body and that of a boundary point, produced by the elastic deformation. The expressions of the terms of the recursive relations between bodies, i.e. those appearing at the joints, will be first derived. The relative elastic displacements and rotations are to be addressed later, since they are based upon the same kinematic relations defined for the joints.

In order to derive these recursive relations, it is useful to express the velocity $\dot{\mathbf{r}}$ and acceleration $\ddot{\mathbf{r}}$ of the origin of a frame of reference, let it be that of a body or a joint, in terms of $\mathbf{Z}_r$ and $\dot{\mathbf{Z}}_r$:

$$\dot{\mathbf{r}} = \dot{\mathbf{s}} + \boldsymbol{\omega} \times \mathbf{r} \tag{2.26}$$

$$\ddot{\mathbf{r}} = \ddot{\mathbf{s}} + \dot{\boldsymbol{\omega}} \times \mathbf{r} + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}) \tag{2.27}$$

**Translational joint**

The relative translational velocity introduced between two joint frames by a translational joint is obtained by differentiating Eq (2.16), being the angular velocities unmodified

$$\dot{\mathbf{r}}_j^Q = \dot{\mathbf{r}}_i^P + \dot{z}_j \mathbf{v}_i^P + \boldsymbol{\omega}_i^P \times z_j \mathbf{v}_i^P \tag{2.28}$$

$$\boldsymbol{\omega}_j^Q = \boldsymbol{\omega}_i^P \tag{2.29}$$

If these equations are combined with the expressions of the absolute velocities $\dot{\mathbf{r}}_i^P$ and $\dot{\mathbf{r}}_j^Q$ in terms of the $\mathbf{Z}_r$ velocities of both frames, by means of Eq. (2.26), the following relation between the reference translational velocities can be derived:

$$\dot{\mathbf{s}}_j^Q = \dot{\mathbf{s}}_i^P + \mathbf{v}_i^P \dot{z}_j \tag{2.30}$$

This equation, along with the equality between the angular velocities, can be written in vector form, leading to an expression of the form shown in Eq. (2.24), thus yielding the $\mathbf{b}_j$ term for this type of joint

$$\begin{Bmatrix} \dot{\mathbf{s}}_j \\ \boldsymbol{\omega}_j^Q \end{Bmatrix} = \begin{Bmatrix} \dot{\mathbf{s}}_i^P \\ \boldsymbol{\omega}_i^P \end{Bmatrix} + \begin{Bmatrix} \mathbf{v}_i^P \\ 0 \end{Bmatrix} \dot{z}_j \implies \mathbf{b}_j = \begin{Bmatrix} \mathbf{v}_i^P \\ 0 \end{Bmatrix} \tag{2.31}$$

The same reasoning can be applied to the accelerations, in order to obtain the $\mathbf{d}_j$ term. First, the velocity relations in terms of the absolute positions and velocities of the joints are differentiated with respect to time,

$$\ddot{\mathbf{r}}_j^Q = \ddot{\mathbf{r}}_i^P + \ddot{z}_j \mathbf{v}_i^P + \dot{\boldsymbol{\omega}}_i^P \times z_j \mathbf{v}_i^P + \boldsymbol{\omega}_i^P \times \boldsymbol{\omega}_i^P \times z_j \mathbf{v}_i^P + 2\boldsymbol{\omega}_i^P \times \dot{z}_j \mathbf{v}_i^P \tag{2.32}$$

$$\dot{\boldsymbol{\omega}}_j^Q = \dot{\boldsymbol{\omega}}_i^P \tag{2.33}$$

The translational accelerations $\ddot{\mathbf{r}}_j^Q$ and $\ddot{\mathbf{r}}_i^P$ can be expressed in terms of $\mathbf{Z}$ and $\dot{\mathbf{Z}}$, in this case by using Eq. (2.27). By combining all these equations, it is not difficult to find that

$$\ddot{\mathbf{s}}_j^Q = \ddot{\mathbf{s}}_i^P + \mathbf{v}_i^P \ddot{z}_j + 2\tilde{\boldsymbol{\omega}}_i^P \mathbf{v}_i^P \dot{z}_j \tag{2.34}$$

If this expression is again written in vector form, along with the equality between the angular accelerations, the result is an expression of the form of Eq. (2.25) so that, by analogy, the form of the $\mathbf{d}_j$ term for a translational joint can be finally obtained

$$\begin{Bmatrix} \ddot{\mathbf{s}}_j^Q \\ \dot{\boldsymbol{\omega}}_j^Q \end{Bmatrix} = \begin{Bmatrix} \ddot{\mathbf{s}}_i^P \\ \dot{\boldsymbol{\omega}}_i^P \end{Bmatrix} + \begin{Bmatrix} \mathbf{v}_i^P \\ 0 \end{Bmatrix} \ddot{z}_j + \begin{Bmatrix} 2\tilde{\boldsymbol{\omega}}_i^P \mathbf{v}_i^P \dot{z}_j \\ 0 \end{Bmatrix} \implies \mathbf{d}_j = \begin{Bmatrix} 2\tilde{\boldsymbol{\omega}}_i^P \mathbf{v}_i^P \dot{z}_j \\ 0 \end{Bmatrix} \tag{2.35}$$

**Revolute joint**

The velocities of two consecutive frames connected by a revolute joint $j$, defined by a vector $\mathbf{v}_i^P$, are related as follows:

$$\dot{\mathbf{r}}_j^P = \dot{\mathbf{r}}_i^P \tag{2.36}$$

$$\boldsymbol{\omega}_j^Q = \boldsymbol{\omega}_i^P + \dot{z}_j \mathbf{v}_i^P \tag{2.37}$$

It they are combined with the velocities expressed in terms of the $\dot{\mathbf{s}}$ and $\boldsymbol{\omega}$ velocities, as done in the translational joint, and the result is written in vector form, an expression of the form of Eq. (2.24) is again obtained, leading to the following result for the $\mathbf{b}_j$

term of a revolute joint:

$$\mathbf{b}_j = \begin{Bmatrix} \tilde{\mathbf{r}}_j^Q \mathbf{v}_i^P \\ \mathbf{v}_i^P \end{Bmatrix} \tag{2.38}$$

By differentiating the velocity relations, the relative translational and angular accelerations can be obtained

$$\ddot{\mathbf{r}}_j^P = \ddot{\mathbf{r}}_i^P \tag{2.39}$$

$$\dot{\boldsymbol{\omega}}_j^Q = \dot{\boldsymbol{\omega}}_i^P + \ddot{z}_j \mathbf{v}_i^P + \boldsymbol{\omega}_i^P \times \dot{z}_j \mathbf{v}_i^P \tag{2.40}$$

After several manipulations, by following an analogous procedure to that used in the translational joint, the value of $\mathbf{d}_j$ for a revolute joint can be shown to be

$$\mathbf{d}_j = \begin{Bmatrix} \tilde{\mathbf{r}}_i^P \tilde{\boldsymbol{\omega}}_i^P \mathbf{v}_i^P \dot{z}_j + \left( \tilde{\boldsymbol{\omega}}_i^P \tilde{\boldsymbol{\omega}}_i^P - \tilde{\boldsymbol{\omega}}_j^Q \tilde{\boldsymbol{\omega}}_j^Q \right) \mathbf{r}_i^P \\ \tilde{\boldsymbol{\omega}}_i^P \mathbf{v}_i^P \dot{z}_j \end{Bmatrix} \tag{2.41}$$

**Static modes**

Since the static modal amplitudes behave as relative coordinates, recursive relations analogous to those obtained for rigid joints can be established for them. Considering an output joint, placed at point $P$ of body $i$, the relation is to be defined in this case between the $\mathbf{Z}_r$ velocities of the body itself, and those of the local frame attached to the joint

$$\mathbf{Z}_{ri}^P = \mathbf{Z}_{ri} + \boldsymbol{\varphi}_i^P \dot{\boldsymbol{\eta}}_i^P \tag{2.42}$$

where $\boldsymbol{\varphi}_i^P$ is a matrix containing the recursive velocity relations of all the six static modes defined at point $P$ of body $i$. This term is the result of assembling the $\mathbf{b}$ terms of three translational and three revolute joints, whose axes coincide with those of the undeformed frame in absolute coordinates

$$\boldsymbol{\varphi}_i^P = \begin{bmatrix} \mathbf{A}_{ui}^P & \tilde{\mathbf{r}}_i^P \mathbf{A}_{ui}^P \\ 0 & \mathbf{A}_{ui}^P \end{bmatrix} \tag{2.43}$$

In this expression, $\mathbf{A}_{ui}^P$ is the orientation of the undeformed joint frame, in global coordinates, directly obtained as $\mathbf{A}_i \bar{\mathbf{A}}_{ui}^P$. In case that any of the six static modes is not present, let it be because the corresponding degree of freedom of the kinematic joint eliminates it, or because the mode is meant to be neglected, the corresponding column of $\boldsymbol{\varphi}_i^P$ is simply removed from the matrix.

A relation, analogous to that shown in Eq. (2.42), might be defined at the input joint of body $j$. It must be taken into account that, since in the case of input joints the relation goes from the joint to the body, the $\boldsymbol{\varphi}_j^Q$ must have its sign reversed,

$$\boldsymbol{\varphi}_j^Q = -\begin{bmatrix} \mathbf{A}_{uj}^Q & \tilde{\mathbf{r}}_j^Q \mathbf{A}_{uj}^Q \\ 0 & \mathbf{A}_{uj}^Q \end{bmatrix} \tag{2.44}$$

The complete set of recursive relations produced at a joint between two flexible bodies results, after considering the three relative motions

$$\mathbf{Z}_{ri}^P = \mathbf{Z}_{ri} + \boldsymbol{\varphi}_i^P \dot{\boldsymbol{\eta}}_i^P \tag{2.45}$$

$$\mathbf{Z}_{rj}^Q = \mathbf{Z}_{ri}^P + \mathbf{b}_j \dot{z}_j \tag{2.46}$$

$$\mathbf{Z}_{rj} = \mathbf{Z}_{rj}^Q + \boldsymbol{\varphi}_j^Q \dot{\boldsymbol{\eta}}_j^Q \tag{2.47}$$

These three relations can be combined into one, thus yielding the complete relation between the velocities of two flexible bodies $i$ and $j$, as a function of the relative coordinates:

$$\mathbf{Z}_{rj} = \mathbf{Z}_{ri} + \boldsymbol{\varphi}_i^P \dot{\boldsymbol{\eta}}_i^P + \mathbf{b}_j \dot{z}_j + \boldsymbol{\varphi}_j^Q \dot{\boldsymbol{\eta}}_j^Q \tag{2.48}$$

In what regards accelerations, the complete recursive relationship is obtained by differentiating this expression,

$$\dot{\mathbf{Z}}_{rj} = \dot{\mathbf{Z}}_{ri} + \boldsymbol{\varphi}_i^P \ddot{\boldsymbol{\eta}}_i^P + \mathbf{b}_j \ddot{z}_j + \boldsymbol{\varphi}_j^Q \ddot{\boldsymbol{\eta}}_j^Q + \boldsymbol{\gamma}_i^P + \mathbf{d}_j + \boldsymbol{\gamma}_j^Q \tag{2.49}$$

The $\boldsymbol{\gamma}_i^P$ terms appearing here are obtained from the relation between accelerations. In order to calculate them, the angular velocity at the joint frame will be needed. This velocity is directly obtained during the forward velocity analysis, by means of Eq. (2.42), and it is equal to

$$\boldsymbol{\omega}_i^P = \boldsymbol{\omega}_i + \mathbf{A}_{ui}^P \dot{\boldsymbol{\eta}}_{\theta i}^P = \boldsymbol{\omega}_i + \boldsymbol{\omega}_{fi}^P \tag{2.50}$$

where $\boldsymbol{\omega}_{fi}^P$ is the relative angular velocity due to deformation. By using these values of the angular velocities, the total vector of velocity–dependent terms for a deformable boundary results,

$$\boldsymbol{\gamma}_i^P = \begin{Bmatrix} 2\tilde{\boldsymbol{\omega}}_i \mathbf{A}_{ui}^P \dot{\boldsymbol{\eta}}_{ti}^P \\ 0 \end{Bmatrix} + \begin{Bmatrix} \tilde{\mathbf{r}}_i^P \tilde{\boldsymbol{\omega}}_i \boldsymbol{\omega}_{fi}^P + \left(\tilde{\boldsymbol{\omega}}_i \tilde{\boldsymbol{\omega}}_i - \tilde{\boldsymbol{\omega}}_i^P \tilde{\boldsymbol{\omega}}_i^P\right) \mathbf{r}_i^P \\ \tilde{\boldsymbol{\omega}}_i \boldsymbol{\omega}_{fi}^P \end{Bmatrix} \tag{2.51}$$

The first term is a Coriolis acceleration produced by the translational static modes, and the second one includes the velocity–dependent accelerations produced by the elastic rotations. As it happens to the velocity relations, in case the considered point with static modes is placed at the input of a body, the sign of the corresponding $\boldsymbol{\gamma}_j^Q$ vector must be changed.

If a tangent frame at the input joint of each body is used, as suggested when describing the kinematics of a flexible body, there will be no input static modes, so that the velocity and acceleration relations from body $i$ to body $j$ are simplified to

$$\mathbf{Z}_{rj} = \mathbf{Z}_{ri} + \boldsymbol{\varphi}_i^P \dot{\boldsymbol{\eta}}_i^P + \mathbf{b}_j \dot{z}_j \tag{2.52}$$

$$\dot{\mathbf{Z}}_{rj} = \dot{\mathbf{Z}}_{ri} + \boldsymbol{\varphi}_i^P \ddot{\boldsymbol{\eta}}_i^P + \mathbf{b}_j \ddot{z}_j + \boldsymbol{\gamma}_i^P + \mathbf{d}_j \tag{2.53}$$

## 2.4 Inertia terms in Cartesian coordinates

In this section, the derivation of the inertia terms, i.e. the mass matrix $\mathbf{M}$ and the velocity dependent inertia forces vector $\mathbf{Q}_v$, is addressed. As has been mentioned before, the direct calculation of these terms in relative coordinates $\mathbf{z}$ is not practical, and for that reason an intermediate Cartesian set of velocities $\mathbf{Z}$ is used, obtaining intermediate inertia terms $\bar{\mathbf{M}}$ and $\bar{\mathbf{Q}}_v$ for each body, which are later projected into the relative coordinates by means of a velocity transformation. This section is focused on the calculation of these intermediate terms in Cartesian coordinates.

The kinetic energy can be expressed as the mass integral of the square of the modulus of the velocity $\dot{\mathbf{r}}$ over the whole volume $V$ of a body,

$$T = \frac{1}{2}\int_V |\dot{\mathbf{r}}|^2 \, dm = \frac{1}{2}\int_V \dot{\mathbf{r}}^{\mathsf{T}}\dot{\mathbf{r}}\, dm \tag{2.54}$$

and in general, if a relation can be established between the velocity of a particle $\dot{\mathbf{r}}$ and the generalized velocities $\dot{\mathbf{q}}$, it is always possible to find an expression of the form

$$T = \frac{1}{2}\dot{\mathbf{q}}^{\mathsf{T}}\mathbf{M}\dot{\mathbf{q}} \tag{2.55}$$

where $\mathbf{M}$ is the mass matrix, which contains the inertia properties associated to the generalized coordinates. When rigid bodies are considered, and depending on the type of modeling chosen, it is possible to obtain a constant expression for this matrix. In the inertial family of formulations for flexible systems, this matrix is also constant. But in the FFR formulations the mass matrix is highly nonlinear, ant this introduces velocity dependent inertia forces in the system. In what follows, the calculation of the mass matrix and the inertia forces vector is explained in detail.

### 2.4.1 Mass matrix

The mass matrix of each body in Cartesian coordinates is obtained from the kinetic energy expression. The kinetic energy is, in turn, obtained here by following the corotational approach proposed by Cardona and Géradin (1991). This method assumes that the velocity of any given point of the body, previously rotated into the local frame of the corresponding finite element, can be interpolated among the nodal velocities by using the standard finite element interpolation functions. This is not fully consistent with the interpolation used for calculating the elastic potential, although it yields good results and enables to calculate the inertia terms in a very simple way.

The velocity of any given point of a flexible body can be expressed in global coordinates, or in a corotated frame defined by a transformation matrix $\mathbf{R}$, which defines the local orientation in the vicinity of the point. This enables to write the kinetic energy in local coordinates

$$T = \frac{1}{2}\int_V \dot{\mathbf{r}}^\mathsf{T}\dot{\mathbf{r}}\,dm = \frac{1}{2}\int_V \dot{\mathbf{r}}^\mathsf{T}\mathbf{R}\mathbf{R}^\mathsf{T}\dot{\mathbf{r}}\,dm \tag{2.56}$$

The corotated frame can be defined at finite element level. The absolute orientation of a finite element $e$ with respect to the global frame, $\mathbf{R}^e$, is the result of two transformations:

$$\mathbf{R}^e = \mathbf{A}\bar{\mathbf{A}}^e \tag{2.57}$$

where $\mathbf{A}$ is the transformation matrix of the body, and $\bar{\mathbf{A}}^e$ is that of the finite element, within the local frame defined by $\mathbf{A}$. The corotational approximation introduced by Géradin and Cardona assumes that the velocity, expressed in the corotated frame, can be interpolated among the nodal velocities by using the finite element interpolation functions. In a general non–isoparametric element with $n$ nodes, this can be written as

$$\mathbf{R}^{e\mathsf{T}}\dot{\mathbf{r}} \approx \mathbf{N}\dot{\bar{\mathbf{q}}}^e \tag{2.58}$$

where $\dot{\bar{\mathbf{q}}}^e$ is a vector containing the derivatives of the nodal positions, $\dot{\mathbf{r}}_i^e$, and infinitesimal rotations, $\boldsymbol{\Omega}_i^e$, rotated into the local coordinates of the element

$$\dot{\bar{\mathbf{q}}}^e = \begin{Bmatrix} \mathbf{R}^{e\mathsf{T}}\dot{\mathbf{r}}_1^e \\ \mathbf{R}^{e\mathsf{T}}\boldsymbol{\Omega}_1^e \\ \vdots \\ \mathbf{R}^{e\mathsf{T}}\dot{\mathbf{r}}_n^e \\ \mathbf{R}^{e\mathsf{T}}\boldsymbol{\Omega}_n^e \end{Bmatrix} \tag{2.59}$$

The kinetic energy of a finite element can then be obtained as a function of the nodal velocities in element coordinates

$$T^e = \frac{1}{2}\int_{V^e} \dot{\bar{\mathbf{q}}}^{e\mathsf{T}}\mathbf{N}^\mathsf{T}\mathbf{N}\dot{\bar{\mathbf{q}}}^e\,dm \tag{2.60}$$

If the local element orientation $\bar{\mathbf{A}}^e$ appearing in the $\mathbf{R}^e$ matrices of $\dot{\bar{\mathbf{q}}}^e$ is retained inside the integral, the *discrete form* of the kinetic energy is obtained

$$T^e = \frac{1}{2}\dot{\mathbf{q}}^{e\mathsf{T}}\mathbf{M}^e\dot{\mathbf{q}}^e \tag{2.61}$$

where $\mathbf{M}^e$ is the standard mass matrix of the finite element, and $\dot{\mathbf{q}}^e$ is a vector containing the nodal velocities, both of them expressed in the local frame of the body, so that

$$\dot{\mathbf{q}}^e = \begin{Bmatrix} \mathbf{A}^{\mathsf{T}}\dot{\mathbf{r}}_1^e \\ \mathbf{A}^{\mathsf{T}}\boldsymbol{\Omega}_1^e \\ \vdots \\ \mathbf{A}^{\mathsf{T}}\dot{\mathbf{r}}_n^e \\ \mathbf{A}^{\mathsf{T}}\boldsymbol{\Omega}_n^e \end{Bmatrix} = \begin{Bmatrix} \dot{\bar{\mathbf{r}}}_1^e \\ \bar{\boldsymbol{\Omega}}_1^e \\ \vdots \\ \dot{\bar{\mathbf{r}}}_n^e \\ \bar{\boldsymbol{\Omega}}_n^e \end{Bmatrix} \tag{2.62}$$

This approximation can be exact in some certain cases, as long as the interpolation matrix is invariant to rotation, as it happens in the case of isoparametric elements. The interpolation of positions and displacements in isoparametric elements of $n$ nodes has the form

$$x = \sum_{i=1}^{n} x_i^e n_i; \quad y = \sum_{i=1}^{n} y_i^e n_i; \quad z = \sum_{i=1}^{n} z_i^e n_i \tag{2.63}$$

where $n_i$ is the interpolation function corresponding to node $i$. According to this, the interpolation matrix will be always formed by diagonal blocks, which are invariant to rotation, so that in Eq. (2.61), even the absolute velocities can be used with the same mass matrix without changing the results.

The total kinetic energy will be the sum of that of all the $n_e$ finite elements

$$T = \sum_{e=1}^{n_e} T^e = \frac{1}{2}\sum_{e=1}^{n_e} \dot{\mathbf{q}}^{e\mathsf{T}}\mathbf{M}^e\dot{\mathbf{q}}^e \tag{2.64}$$

A finite element mass matrix can be assembled for the whole body, following the standard finite element procedure. The velocities of all the finite element nodes are put together in a nodal velocity vector $\dot{\mathbf{q}}^*$. Then, a full–size mass matrix $\mathbf{M}^{e*}$ is defined for each finite element, being its size the number of degrees of freedom of the finite element model. It contains zeros in all elements, and the element mass matrix $\mathbf{M}^e$ at the positions corresponding to the coordinates of the element within the nodal velocities vector,

$$\dot{\mathbf{q}}^* = \begin{Bmatrix} \dot{\mathbf{q}}^1 \\ \dot{\mathbf{q}}^2 \\ \vdots \\ \dot{\mathbf{q}}^e \\ \vdots \\ \dot{\mathbf{q}}^{n_e} \end{Bmatrix} \Longrightarrow \mathbf{M}^{e*} = \begin{bmatrix} 0 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{M}^e & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \cdots & 0 \end{bmatrix} \tag{2.65}$$

where $\dot{\mathbf{r}}^e$ includes the velocities of all the nodes of finite element $e$. The total finite element mass matrix is the sum of all these full–size element mass matrices

$$\mathbf{M}^* = \sum_{e=1}^{n_e} \mathbf{M}^{e*} \tag{2.66}$$

This allows for writing the kinetic energy into its discrete form

$$T = \frac{1}{2}\dot{\mathbf{q}}^{*\mathsf{T}}\mathbf{M}^*\dot{\mathbf{q}}^* \tag{2.67}$$

The finite element mass matrix is constant, and it can be directly obtained from any standard finite element code, so that this approximation leads to a very convenient method for calculating the kinetic energy. The accuracy is also very good, and it converges to the exact energy as the finite element size decreases, being, as has been noted before, exact in case that isoparametric elements are used.

In order to calculate the mass matrix expressed in the intermediate Cartesian coordinates $\mathbf{Z}$, a linear relationship between the velocity of any given node of the body $\dot{\mathbf{q}}$ and the body velocities $\mathbf{Z}$ must be established. The position shown in Eq. (2.1) can be differentiated to yield the velocity, by taking into account that the local deformation velocity $\dot{\bar{\mathbf{r}}}$ is equal to $\mathbf{X}\dot{\mathbf{y}}$, since both the undeformed position $\bar{\mathbf{r}}_u$ and the mode shapes matrix $\mathbf{X}$ are constant,

$$\dot{\mathbf{r}} = \dot{\mathbf{r}}_0 + \dot{\mathbf{A}}\bar{\mathbf{r}} + \mathbf{A}\mathbf{X}_t\dot{\mathbf{y}} \tag{2.68}$$

where the matrix $\mathbf{X}_t$ contains only the translational components of the mode shapes. At this point, the instantaneous angular velocity vector $\boldsymbol{\omega}$ is introduced, since the time variation of the orientation matrix is

$$\dot{\mathbf{A}} = \begin{bmatrix} \boldsymbol{\omega} \times \mathbf{u} & \boldsymbol{\omega} \times \mathbf{v} & \boldsymbol{\omega} \times \mathbf{w} \end{bmatrix} = \tilde{\boldsymbol{\omega}}\mathbf{A} \tag{2.69}$$

where the cross product has been substituted by the skew–symmetric matrix associated to the angular velocity $\boldsymbol{\omega}$. This leads to

$$\dot{\mathbf{r}} = \dot{\mathbf{r}}_0 + \tilde{\boldsymbol{\omega}}\mathbf{A}\bar{\mathbf{r}} + \mathbf{A}\mathbf{X}_t\dot{\mathbf{y}} \tag{2.70}$$

The velocity of the origin of the frame of reference, $\dot{\mathbf{r}}_0$, must be expressed as a function of $\dot{\mathbf{s}}$ and $\boldsymbol{\omega}$, and this is done by applying Eq. (2.26),

$$\dot{\mathbf{r}}_0 = \dot{\mathbf{s}} + \boldsymbol{\omega} \times \mathbf{r}_0 = \dot{\mathbf{s}} + \tilde{\boldsymbol{\omega}}\mathbf{r}_0 \tag{2.71}$$

By substituting this into Eq. (2.70), it can be shown that,

$$\dot{\mathbf{r}} = \dot{\mathbf{s}} + \tilde{\boldsymbol{\omega}}\left(\mathbf{r}_0 + \mathbf{A}\bar{\mathbf{r}}\right) + \mathbf{A}\mathbf{X}_t\dot{\mathbf{y}} = \dot{\mathbf{s}} + \tilde{\boldsymbol{\omega}}\mathbf{r} + \mathbf{A}\mathbf{X}_t\dot{\mathbf{y}} \tag{2.72}$$

The cross product can be inverted, in order to obtain the velocity $\dot{\mathbf{r}}$ as a linear function of the intermediate Cartesian velocities **Z**

$$\dot{\mathbf{r}} = \dot{\mathbf{s}} - \tilde{\mathbf{r}}\boldsymbol{\omega} + \mathbf{A}\mathbf{X}_t\dot{\mathbf{y}} \tag{2.73}$$

The angular velocity at a node is that of the body, plus the part due to the deformation, expressed in absolute coordinates,

$$\boldsymbol{\Omega} = \boldsymbol{\omega} + \mathbf{A}\mathbf{X}_\theta\dot{\mathbf{y}} \tag{2.74}$$

where $\mathbf{X}_\theta$ contains the rotational components of the mode shapes. This expression, along with Eq. (2.73), can be rotated into the local frame of the body and written in matrix form, to yield a matrix **B** that relates the nodal translational and angular velocities in the local axes $\dot{\mathbf{q}}$ to the intermediate Cartesian velocities **Z**:

$$\dot{\mathbf{q}} = \begin{Bmatrix} \mathbf{A}^\mathsf{T}\dot{\mathbf{r}} \\ \mathbf{A}^\mathsf{T}\boldsymbol{\Omega} \end{Bmatrix} = \begin{bmatrix} \mathbf{A}^\mathsf{T} & -\mathbf{A}^\mathsf{T}\tilde{\mathbf{r}} & \mathbf{X}_t \\ 0 & \mathbf{A}^\mathsf{T} & \mathbf{X}_\theta \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{s}} \\ \boldsymbol{\omega} \\ \dot{\mathbf{y}} \end{Bmatrix} = \mathbf{B}\mathbf{Z} \tag{2.75}$$

In the case of isoparametric elements, since the interpolation functions are invariant to rotation, and there exist no infinitesimal rotations, this can be written in a simpler form, directly relating the absolute velocities of the nodes to **Z**

$$\dot{\mathbf{r}} = \begin{bmatrix} \mathbf{I}_3 & -\tilde{\mathbf{r}} & \mathbf{A}\mathbf{X} \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{s}} \\ \boldsymbol{\omega} \\ \dot{\mathbf{y}} \end{Bmatrix} = \mathbf{B}\mathbf{Z} \tag{2.76}$$

In what follows, the isoparametric expressions are used for the sake of simplicity, being the development for non–isoparametric elements straightforward. A full $\mathbf{B}^*$ matrix can be assembled, containing the transformation for all the $n$ nodes, as follows:

$$\mathbf{B}^* = \begin{bmatrix} \mathbf{B}_1 \\ \mathbf{B}_2 \\ \cdots \\ \mathbf{B}_n \end{bmatrix} = \begin{bmatrix} \mathbf{I}_3 & -\tilde{\mathbf{r}}_1 & \mathbf{A}\mathbf{X}_1 \\ \mathbf{I}_3 & -\tilde{\mathbf{r}}_2 & \mathbf{A}\mathbf{X}_2 \\ \cdots & \cdots & \cdots \\ \mathbf{I}_3 & -\tilde{\mathbf{r}}_n & \mathbf{A}\mathbf{X}_n \end{bmatrix} \tag{2.77}$$

This large $\mathbf{B}^*$ matrix, which has $3n$ rows ($6n$ in structural finite elements) and $6 + n_m$ columns, can be used to calculate the velocities of all finite element nodes at once,

$$\dot{\mathbf{r}}^* = \mathbf{B}^*\mathbf{Z} \tag{2.78}$$

so that it can be substituted into the discrete form of the kinetic energy to yield,

$$T = \frac{1}{2}\mathbf{Z}^\mathsf{T}\mathbf{B}^{*\mathsf{T}}\mathbf{M}^*\mathbf{B}^*\mathbf{Z} \tag{2.79}$$

which means that the mass matrix is the result of projecting the finite element mass matrix into the intermediate Cartesian coordinates, by means of the velocity transformation matrix $\mathbf{B}^*$,

$$\bar{\mathbf{M}} = \mathbf{B}^{*\mathsf{T}}\mathbf{M}^*\mathbf{B}^* \tag{2.80}$$

The calculation of the mass matrix at every iteration of the integrator involves then two steps. First, the $\mathbf{B}^*$ matrix must be assembled, by calculating its value at each finite element node. Then, the finite element mass matrix is projected into the $\mathbf{Z}$ coordinates by performing the product in Eq. (2.80). This product can become a very CPU intensive task if the finite element models are too large, which is one of the main drawbacks of this method. The reduction of the finite element model by means of component mode synthesis is not fully taken advantage of, since although only the modal amplitudes are problem variables, operations involving the finite element mass matrix still need to be performed. It must be noted that the finite element mass matrix is in general highly sparse, and if the projection is carried out taking advantage of this sparsity, the method is still applicable to relatively large finite element models. The solution to this problem is addressed in the next chapter, by introducing the inertia shape integrals, which allow for completely eliminating the finite element mass matrix from the problem, at the cost of a more involved implementation.

In Avello (1995) and Gutiérrez (2003), the finite elements are always treated as if they were isoparametric. Instead of using the $\mathbf{B}^*$ matrix as defined in Eq. (2.75), i.e. in local coordinates and including the rows corresponding to rotations, the expression for isoparametric elements is adopted. This implies that a new finite element mass matrix must be calculated by using the isoparametric interpolation functions, in order to perform the projection in Eq. (2.80). This approximation works very well and significantly reduces the computation time, since the number of rows of $\mathbf{B}^*$ is cut to its half, and so happens to the order of the finite element mass matrix. In the examples addressed in this thesis, no difference in the results has been observed from using this simplified interpolation, so that this is the method adopted.

### 2.4.2 Centrifugal and Coriolis forces vector

Application of the Lagrange's equations to a single body, with the kinetic energy expressed in the $\mathbf{Z}$ coordinates, leads to the following expression for the velocity dependent inertia forces:

$$\bar{\mathbf{Q}}_v = -\mathbf{B}^{*\mathsf{T}}\mathbf{M}^*\dot{\mathbf{B}}^*\mathbf{Z} \tag{2.81}$$

It can be observed that the product $\mathbf{B}^{*\mathsf{T}}\mathbf{M}^*$ has been already performed when projecting the mass matrix, so that it can be stored and used to calculate the inertia forces also. Once this product has been carried out and stored, the $\dot{\mathbf{B}}^*\mathbf{Z}$ product is the only thing left to be calculated, and the acceleration will be calculated for this purpose. The acceleration of any given point is, by differentiating Eq. (2.76),

$$\ddot{\mathbf{r}} = \mathbf{B}\dot{\mathbf{Z}} + \dot{\mathbf{B}}\mathbf{Z} \tag{2.82}$$

The acceleration can also be obtained by differentiating the velocity of a generic point shown in Eq. (2.70),

$$\ddot{\mathbf{r}} = \ddot{\mathbf{r}}_0 + \dot{\tilde{\boldsymbol{\omega}}}\mathbf{A}\bar{\mathbf{r}} + \tilde{\boldsymbol{\omega}}\dot{\mathbf{A}}\bar{\mathbf{r}} + \tilde{\boldsymbol{\omega}}\mathbf{A}\mathbf{X}\dot{\mathbf{y}} + \dot{\mathbf{A}}\mathbf{X}\dot{\mathbf{y}} + \mathbf{A}\mathbf{X}\ddot{\mathbf{y}} \tag{2.83}$$

which can be rewritten as,

$$\ddot{\mathbf{r}} = \ddot{\mathbf{r}}_0 + \dot{\tilde{\boldsymbol{\omega}}}\mathbf{A}\bar{\mathbf{r}} + \tilde{\boldsymbol{\omega}}\tilde{\boldsymbol{\omega}}\mathbf{A}\bar{\mathbf{r}} + 2\dot{\mathbf{A}}\mathbf{X}\dot{\mathbf{y}} + \mathbf{A}\mathbf{X}\ddot{\mathbf{y}} \tag{2.84}$$

By using Eq. (2.27), the acceleration of the origin of the floating frame is derived

$$\ddot{\mathbf{r}}_0 = \ddot{\mathbf{s}} + \dot{\boldsymbol{\omega}} \times \mathbf{r}_0 + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}_0) = \ddot{\mathbf{s}} + \dot{\tilde{\boldsymbol{\omega}}}\mathbf{r}_0 + \tilde{\boldsymbol{\omega}}\tilde{\boldsymbol{\omega}}\mathbf{r}_0 \tag{2.85}$$

and this can be substituted into Eq. (2.84),

$$\ddot{\mathbf{r}} = \ddot{\mathbf{s}} + \dot{\tilde{\boldsymbol{\omega}}}\left(\mathbf{r}_0 + \mathbf{A}\bar{\mathbf{r}}\right) + \tilde{\boldsymbol{\omega}}\tilde{\boldsymbol{\omega}}\left(\mathbf{r}_0 + \mathbf{A}\bar{\mathbf{r}}\right) + 2\dot{\mathbf{A}}\mathbf{X}\dot{\mathbf{y}} + \mathbf{A}\mathbf{X}\ddot{\mathbf{y}} \tag{2.86}$$

where, after substituting $\mathbf{r}_0 + \mathbf{A}\bar{\mathbf{r}}$ by $\mathbf{r}$ and swapping the angular acceleration product, the first term of Eq. (2.82) is easily identified,

$$\ddot{\mathbf{r}} = (\ddot{\mathbf{s}} - \tilde{\mathbf{r}}\dot{\boldsymbol{\omega}} + \mathbf{A}\mathbf{X}\ddot{\mathbf{y}}) + \tilde{\boldsymbol{\omega}}\tilde{\boldsymbol{\omega}}\mathbf{r} + 2\dot{\mathbf{A}}\mathbf{X}\dot{\mathbf{y}} = \mathbf{B}\dot{\mathbf{Z}} + \tilde{\boldsymbol{\omega}}\tilde{\boldsymbol{\omega}}\mathbf{r} + 2\dot{\mathbf{A}}\mathbf{X}\dot{\mathbf{y}} \tag{2.87}$$

which means that the product $\dot{\mathbf{B}}\mathbf{Z}$ must be

$$\dot{\mathbf{B}}\mathbf{Z} = \tilde{\boldsymbol{\omega}}\tilde{\boldsymbol{\omega}}\mathbf{r} + 2\dot{\mathbf{A}}\mathbf{X}\dot{\mathbf{y}} \tag{2.88}$$

Since $\dot{\mathbf{A}}$ is equal to $\tilde{\boldsymbol{\omega}}\mathbf{A}$, and $\mathbf{A}\mathbf{X}\dot{\mathbf{y}}$ is nothing but the relative velocity of deformation in global coordinates, these two terms are easily identifiable as the centrifugal and Coriolis accelerations of point $\mathbf{r}$, respectively. The resulting expression for $\dot{\mathbf{B}}$ will be

$$\dot{\mathbf{B}} = \begin{bmatrix} 0 & -\tilde{\boldsymbol{\omega}}\tilde{\mathbf{r}} & 2\dot{\mathbf{A}}\mathbf{X} \end{bmatrix} \tag{2.89}$$

The total vector for the whole body can be obtained by evaluating this at every finite element node,

$$\dot{\mathbf{B}}^{*}\mathbf{Z} = \begin{bmatrix} 0 & -\tilde{\boldsymbol{\omega}}\tilde{\mathbf{r}}_1 & 2\dot{\mathbf{A}}\mathbf{X}_1 \\ 0 & -\tilde{\boldsymbol{\omega}}\tilde{\mathbf{r}}_2 & 2\dot{\mathbf{A}}\mathbf{X}_2 \\ \vdots & \vdots & \vdots \\ 0 & \tilde{\boldsymbol{\omega}}\tilde{\mathbf{r}}_n & 2\dot{\mathbf{A}}\mathbf{X}_n \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{s}} \\ \boldsymbol{\omega} \\ \dot{\mathbf{y}} \end{Bmatrix} = \begin{Bmatrix} \tilde{\boldsymbol{\omega}}\tilde{\boldsymbol{\omega}}\mathbf{r}_1 + 2\dot{\mathbf{A}}\mathbf{X}_1\dot{\mathbf{y}} \\ \tilde{\boldsymbol{\omega}}\tilde{\boldsymbol{\omega}}\mathbf{r}_2 + 2\dot{\mathbf{A}}\mathbf{X}_2\dot{\mathbf{y}} \\ \vdots \\ \tilde{\boldsymbol{\omega}}\tilde{\boldsymbol{\omega}}\mathbf{r}_n + 2\dot{\mathbf{A}}\mathbf{X}_n\dot{\mathbf{y}} \end{Bmatrix} \tag{2.90}$$

The most practical way to compute the inertia terms is to calculate $\dot{\mathbf{B}}^{*}\mathbf{Z}$ within the same loop in which $\mathbf{B}^{*}$ is assembled. Then, the sparse $\mathbf{B}^{*\mathsf{T}}\mathbf{M}^{*}$ product is evaluated, and used to calculate both the mass matrix and the inertia forces vector.

## 2.5 Non–inertial forces

In this section, all the forces that are not related to inertia are explained. The elastic forces, due to the fact that the elastic deformation is obtained in local coordinates, are very easy to implement, since they only involve the modal amplitudes, and are completely linear. The remaining forces are calculated first in absolute Cartesian coordinates. The externally applied forces are then projected into the intermediate coordinates **Z**, enabling them to be added to the Cartesian forces vector $\bar{\mathbf{Q}}$, which will be in turn projected into the relative coordinates **z**. The forces that depend on the position or velocity, such as those originated by springs and dampers, are directly projected into the relative coordinates, since this makes the calculation of their generalized stiffness or damping matrices easier.

### 2.5.1 Elastic forces

The elastic potential of a deformed body is obtained from the finite element stiffness matrix $\mathbf{K}^*$ and the nodal elastic displacements (Bathe, 1995),

$$U = \frac{1}{2}\bar{\mathbf{r}}_f^{*\mathsf{T}}\mathbf{K}^*\bar{\mathbf{r}}_f^* \tag{2.91}$$

This potential can be obtained also in terms of the modal amplitudes, by projecting the stiffness matrix using the transformation defined in Eq. (2.3). The stiffness matrix projected into the elastic coordinates results,

$$U = \frac{1}{2}\mathbf{y}^{\mathsf{T}}\mathbf{X}^{*\mathsf{T}}\mathbf{K}^*\mathbf{X}^*\mathbf{y} \implies \mathbf{K} = \mathbf{X}^{*\mathsf{T}}\mathbf{K}^*\mathbf{X}^* \tag{2.92}$$

and this constant matrix can be used for the calculation of the elastic forces,

$$\bar{\mathbf{Q}}_f = -\frac{\partial U}{\partial \mathbf{q}} = -\mathbf{K}\mathbf{y} \tag{2.93}$$

In case thata structural finite elements are considered, the stiffness matrix should include the effect of the infinitesimal rotations, so that the full $\mathbf{K}^*$ and $\mathbf{X}^*$ matrices, including all the degrees of freedom, must be used for the projection into the modal space. After the projected stiffness matrix is calculated, the rotations can be eliminated from the mode shapes matrix, since they are not needed for the evaluation of the inertia terms. The fact that the elastic coordinates are the same in both the intermediate Cartesian coordinates **Z** and the relative dependent coordinates **z**, implies that the elastic forces are immediately obtained in relative dependent coordinates by performing the **Ky** product, so that they do not need to be projected.

When a Craig–Bampton reduction is used, the discrete mode shapes matrix $\mathbf{X}^*$ is as follows, according to Eqs. (2.7) and (2.8):

$$\mathbf{X}^* = \begin{bmatrix}\mathbf{X}_s^* & \mathbf{X}_d^*\end{bmatrix} = \begin{bmatrix}\mathbf{I} & 0 \\ -\left(\mathbf{K}_{ii}^*\right)^{-1}\mathbf{K}_{ib}^* & \mathbf{X}_{id}^*\end{bmatrix} \tag{2.94}$$

If the partition is applied to $\mathbf{K}^*$ and the projection is carried out, the projected stiffness matrix can be shown to be

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_s & 0 \\ 0 & \boldsymbol{\Omega}^2 \end{bmatrix} \tag{2.95}$$

where $\mathbf{K}_s$ is a symmetric matrix representing the stiffness of the static modes, and $\boldsymbol{\Omega}^2$ is a diagonal matrix containing the squared natural frequencies of the dynamic modes, which are always mass and stiffness–orthogonal. The off–diagonal blocks are zero, meaning that the static modes are stiffness–orthogonal to the dynamic ones.

## 2.5.2 Applied forces

Applied forces are introduced in body coordinates by means of the virtual power principle. The virtual power of a point force $\mathbf{F}$ applied at node $i$ will be,

$$\dot{W}_a^* = \dot{\mathbf{r}}_i^{*\mathsf{T}}\mathbf{F} \tag{2.96}$$

where the asterisk denotes virtual power or velocity. The virtual velocity can be expressed in terms of the $\mathbf{Z}$ velocities,

$$\dot{\mathbf{r}}_i^* = \left(\frac{\partial \dot{\mathbf{r}}_i}{\partial \mathbf{Z}}\right)\mathbf{Z}^* \tag{2.97}$$

Since the virtual power can also be expressed in the intermediate Cartesian coordinates,

$$\dot{W}_a^* = \mathbf{Z}^{*\mathsf{T}}\bar{\mathbf{Q}}_a \tag{2.98}$$

the generalized forces are identified as

$$\bar{\mathbf{Q}}_a = \left(\frac{\partial \dot{\mathbf{r}}_i}{\partial \mathbf{Z}}\right)^{\mathsf{T}}\mathbf{F} = \mathbf{B}_{ti}^{\mathsf{T}}\mathbf{F} = \begin{bmatrix} \mathbf{I}_3 & -\tilde{\mathbf{r}} & \mathbf{A}\mathbf{X}_{ti} \end{bmatrix}^{\mathsf{T}}\mathbf{F} \tag{2.99}$$

where $\mathbf{B}_{ti}$ is the three–row submatrix of $\mathbf{B}$, expressed in global coordinates, corresponding to the three translational degrees of freedom at node $i$. If the point is not a node, the finite element interpolation functions can be used for evaluating $\mathbf{B}$. The case of an applied moment $\mathbf{T}$ is treated analogously, but using the rotation part of the transformation matrix,

$$\bar{\mathbf{Q}}_a = \left(\frac{\partial \boldsymbol{\Omega}_i}{\partial \mathbf{Z}}\right)^{\mathsf{T}}\mathbf{T} = \mathbf{B}_{\theta i}^{\mathsf{T}}\mathbf{T} = \begin{bmatrix} 0 & \mathbf{I}_3 & \mathbf{A}\mathbf{X}_{\theta i} \end{bmatrix}^{\mathsf{T}}\mathbf{T} \tag{2.100}$$

In case that the point of application of an external force is known a priori, it can be a good idea to include that point as a boundary, including static modes associated to it, in such a way that the deformation produced by that force is more accurately captured.

### 2.5.3 Volume forces

In order to introduce volume forces such as weight in the system, the same approach for point forces is used, but integrating them over all the volume of the body to which the force is applied. Expressing Eq. (2.99) in integral form

$$\bar{\mathbf{Q}}_V = \int_V \mathbf{B}^{\mathsf{T}} \mathbf{f} \, dV \tag{2.101}$$

where **f** is the force per unit volume, expressed in absolute coordinates,

$$\mathbf{f} = \begin{Bmatrix} f_x \\ f_y \\ f_z \end{Bmatrix} \tag{2.102}$$

By introducing this and the expression of **B** previously obtained into Eq. (2.101), the integral remains,

$$\bar{\mathbf{Q}}_V = \int_V \begin{bmatrix} \mathbf{I}_3 \\ \tilde{\mathbf{r}} \\ \mathbf{X}^{\mathsf{T}}\mathbf{A}^{\mathsf{T}} \end{bmatrix} \begin{Bmatrix} f_x \\ f_y \\ f_z \end{Bmatrix} dV \tag{2.103}$$

The force applied to the translational coordinate is the total force acting in the three global directions,

$$\bar{\mathbf{Q}}_{Vr} = \int_V \begin{Bmatrix} f_x \\ f_y \\ f_z \end{Bmatrix} dm \tag{2.104}$$

The next block contains the moment of that force about the global origin of coordinates, since the components of the position appearing in the integrals are expressed with respect to that point

$$\bar{\mathbf{Q}}_{V\theta} = \int_V \begin{Bmatrix} y f_z - z f_y \\ z f_x - x f_z \\ x f_y - y f_x \end{Bmatrix} dm \tag{2.105}$$

and the elements represent the effect of the volume forces on the body deformation.

In the particular case of weight, considering a density $\rho$, and assuming that the gravity acts in the negative direction of the $z$ global axis, with an acceleration $g$, the force per unit volume can be written,

$$\mathbf{f} = \begin{Bmatrix} 0 \\ 0 \\ -\rho g \end{Bmatrix} \tag{2.106}$$

The three first components are the translational forces, which are nothing but the weight acting in the negative direction of the $z$ axis:

$$\bar{\mathbf{Q}}_{Vr} = -g \int_V \begin{Bmatrix} 0 \\ 0 \\ 1 \end{Bmatrix} \rho dV = \begin{Bmatrix} 0 \\ 0 \\ -mg \end{Bmatrix} \tag{2.107}$$

being $m$ the total mass of the body. The rotational forces are

$$\bar{\mathbf{Q}}_{V\theta} = -g \int_V \begin{Bmatrix} y \\ -x \\ 0 \end{Bmatrix} \rho dV = \begin{Bmatrix} -gm_y \\ gm_x \\ 0 \end{Bmatrix} \tag{2.108}$$

where $m_x$, $m_y$ and $m_z$ are the three components of the static moment of the deformed body, $\mathbf{m}$, expressed in global coordinates. This is the moment of the weight acting in the deformed center of gravity $\mathbf{r}^G$, about the global origin of coordinates. Finally, the forces affecting the modal amplitudes,

$$\bar{\mathbf{Q}}_{Vf} = -g \int_V \mathbf{X}^{\mathsf{T}} \mathbf{A}_3^{\mathsf{T}} \rho dV = -g \mathbf{S}^{\mathsf{T}} \mathbf{A}_3^{\mathsf{T}} \tag{2.109}$$

where the inertia shape integral $\mathbf{S}$ appears. The calculation of both $\mathbf{m}$ and $\mathbf{S}$ is addressed in more detail in the second chapter, where the inertia shape integrals are defined. It can be said in advance that the integral of the mode shapes $\mathbf{S}$, which is defined in Eq. (3.9), is needed for obtaining the static moment, as shown in Eq. (3.21).

### 2.5.4 Springs and dampers

Springs and dampers connect different bodies of a mechanism by introducing forces or moments between them, which depend on the position in the case of springs, and on the velocity in the case of dampers. Since they explicitly depend on the generalized coordinates $\mathbf{z}$ or their time derivatives $\dot{\mathbf{z}}$, it is more convenient to project them directly into them without performing any intermediate step. In general, when using relative coordinates, rotational springs and dampers connect consecutive bodies, and since there should exist a coordinate representing the relative angle at that point, introducing them is completely straightforward, as will be seen later. In what follows, the introduction of translational springs and dampers is described in detail.

A translational force element, connecting two points $A$ and $B$, introduces two opposite forces in the mechanism, $\mathbf{f}_{AB}$ at point $A$, and $-\mathbf{f}_{AB}$ at point $B$. Both forces actuate along the straight line connecting them, as can be seen in Figure 2.18, and their magnitude $f_s$ depends on the distance $s$ between $A$ and $B$, if it is a spring, and of its time derivative, $\dot{s}$, if it is a viscous damper. In general, since in many cases the same element may include stiffness and viscous damping, the magnitude of the force will be considered as a general nonlinear function of the distance and its derivative $f_s(s, \dot{s})$, so that the following applies to any generic force element of this kind.

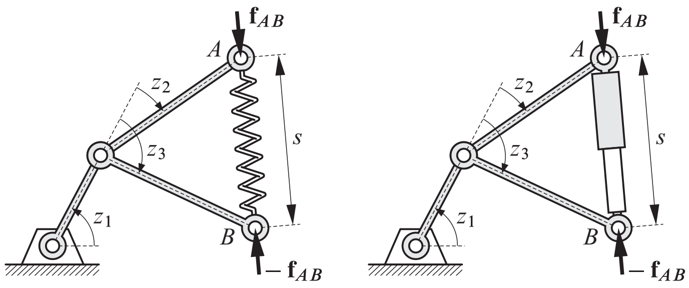


Figure 2.18: Spring and damper in an open–loop mechanism.

The first step is to obtain the force in Cartesian coordinates, which is not difficult provided that the positions and velocities of both points $A$ and $B$ are known. It is a good idea to include those points as boundaries in the flexible body model, adding the corresponding static modes in order to obtain a good approximation of the actual deformation field. The distance between two points of absolute positions $\mathbf{r}_A$ and $\mathbf{r}_B$ is obtained as

$$s = |\mathbf{r}_B - \mathbf{r}_A| = \left[(\mathbf{r}_B - \mathbf{r}_A)^{\mathsf{T}} (\mathbf{r}_B - \mathbf{r}_A)\right]^{1/2} \tag{2.110}$$

and its time derivative, $\dot{s}$, is

$$\dot{s} = \frac{1}{s} (\dot{\mathbf{r}}_B - \dot{\mathbf{r}}_A)^{\mathsf{T}} (\mathbf{r}_B - \mathbf{r}_A) = (\dot{\mathbf{r}}_B - \dot{\mathbf{r}}_A)^{\mathsf{T}} \mathbf{u}_{AB} \tag{2.111}$$

where $\mathbf{u}_{AB}$ is a unit vector, pointing from $A$ to $B$, so that the derivative of the distance is actually the projection of the relative velocity into the direction defined by $\mathbf{u}_{AB}$. By introducing the constitutive law $f_s(s, \dot{s})$ as the magnitude, the force is obtained directly in Cartesian coordinates as the product of the unit vector times the magnitude

$$\mathbf{f}_{AB} = f_s \mathbf{u}_{AB} \tag{2.112}$$

In order to project the forces into the dependent relative coordinates, the virtual power principle is applied as usual, although the projection is performed into the relative coordinates directly. If the two forces corresponding to a spring–damper are introduced simultaneously,

$$\mathbf{Q}_s = -\left(\frac{\partial \mathbf{r}_B}{\partial \mathbf{z}} - \frac{\partial \mathbf{r}_A}{\partial \mathbf{z}}\right)^{\mathsf{T}} \mathbf{f}_{AB} \tag{2.113}$$

By introducing the definition of $\mathbf{f}_{AB}$ from Eq. (2.112), and after some manipulations, the following expression for the generalized forces of a translational force element can be derived:

$$\mathbf{Q}_s = -f_s \mathbf{s}_{\mathbf{z}}^{\mathsf{T}} \tag{2.114}$$

where $\mathbf{s_z}$ is the derivative of $s$ with respect to $\mathbf{z}$,

$$\mathbf{s_z} = \frac{\partial s}{\partial \mathbf{z}} = \left\{ \frac{\partial s}{\partial z_1} \quad \frac{\partial s}{\partial z_2} \quad \cdots \quad \frac{\partial s}{\partial z_n} \right\} \tag{2.115}$$

Each of its components can be expressed as follows, after performing the derivative:

$$\frac{\partial s}{\partial z_i} = \frac{\partial \left(\mathbf{r}_B - \mathbf{r}_A\right)^{\mathsf{T}}}{\partial z_i} \mathbf{u}_{AB} \tag{2.116}$$

This can be seen as a vector that contains the variation produced in $s$ by a unit variation of any coordinate in $\mathbf{z}$. This can be easily calculated considering that the derivative of $s$ with respect to a relative coordinate $z_i$ is the relative velocity between points $A$ and $B$ when a unit velocity is applied to coordinate $z_i$ and the rest of the $z$ coordinates remains unchanged. It is obvious that only coordinates that affect the distance $s$ would have a derivative different from zero. In the examples shown in Figure 2.18, a variation of $z_1$ does not affect $s$ if $z_2$ and $z_3$ remain unchanged, so that the derivative of $s$ with respect to $z_1$ will be zero.

It is useful to obtain the stiffness and damping matrices for these forces, since they are needed later for the tangent matrix of the Newton–Raphson iteration of the integrator. The generalized stiffness and damping matrices can be expressed as,

$$\mathbf{K} = -\frac{\partial \mathbf{Q}}{\partial \mathbf{z}}; \quad \mathbf{C} = -\frac{\partial \mathbf{Q}}{\partial \dot{\mathbf{z}}} \tag{2.117}$$

The stiffness matrix due to a spring force can be obtained by differentiating Eq. (2.114) with respect to $\mathbf{z}$

$$\mathbf{K}_s = -\frac{\partial \mathbf{Q}_s}{\partial \mathbf{z}} = \frac{\partial}{\partial \mathbf{z}} \left( f_s \frac{\partial s}{\partial \mathbf{z}} \right) \approx \frac{\partial f_s}{\partial \mathbf{z}} \frac{\partial s}{\partial \mathbf{z}} = \frac{\partial f_s}{\partial s} \mathbf{s_z}^{\mathsf{T}} \mathbf{s_z} \tag{2.118}$$

When the force element is a damper, the following analogous expression can be obtained,

$$\mathbf{C}_s = -\frac{\partial \mathbf{Q}_s}{\partial \dot{\mathbf{z}}} = \frac{\partial}{\partial \dot{\mathbf{z}}} \left( f_s \frac{\partial s}{\partial \mathbf{z}} \right) \approx \frac{\partial f_s}{\partial \dot{\mathbf{z}}} \frac{\partial s}{\partial \mathbf{z}} = \frac{\partial f_s}{\partial \dot{s}} \mathbf{s_z}^{\mathsf{T}} \mathbf{s_z} \tag{2.119}$$

It can be observed that the same product $\mathbf{s_z}^{\mathsf{T}} \mathbf{s_z}$ appears in both the stiffness and damping matrices, so it only has to be calculated once, and the difference resides in the scalar multiplying it. In the common case of linear springs or dampers, this scalar will be a constant, let it be the stiffness constant of the spring $k_s$ or the damping constant of the damper $c_s$.

## 2.6 Kinematic constraints

Once the dynamic terms have been obtained in relative coordinates for the open loop system, the closure conditions must be imposed. Constraints are expressed in natural coordinates, naming $\mathbf{q}$ the set of points and unit vectors involved in the constraints. The conditions are expressed as a set of nonlinear constraints, as it would have been done in a formulation in natural coordinates, although including only the constraints that are needed to close the previously cut joints. The example of the double four bar mechanism shown in Figure 2.11 will be used to more clearly illustrate the procedure. In this example, the natural coordinates involved are the positions of points $C$ and $E$, which must coincide with those of the fixed points $C'$ and $E'$ respectively, so that the vector $\mathbf{q}$ will have four variables, and the constraints vector is

$$\mathbf{\Phi}(\mathbf{q}) = \begin{Bmatrix} x_C - x_{C'} \\ y_C - y_{C'} \\ x_E - x_{E'} \\ y_E - y_{E'} \end{Bmatrix} \tag{2.120}$$

Obviously, in order to evaluate the constraints, the positions of $C$ and $E$ must be calculated when solving the forward position problem.

When the equations of motion are established in relative coordinates, the Jacobian matrix of the constraints is necessary. In order to obtain it, the chain differentiation rule is used

$$\mathbf{\Phi_z} = \mathbf{\Phi_q q_z} \tag{2.121}$$

The first term is the classical Jacobian in natural coordinates, which happens to be a $4 \times 4$ identity matrix for the example, since each variable appears in one constraint. In general, since the kinematic constraints are imposed to close previously cut closed loops, the constraints in natural coordinates will either impose the values of coordinates, or equalities between them, so that the Jacobian in natural coordinates is likely constant. The second term can be seen as a matrix containing one column for each relative coordinate. The column corresponding to $z_i$ will contain the velocities in natural coordinates $\dot{\mathbf{q}}$, when a unit velocity is given to $z_i$ and the remaining relative coordinates $z_j$ are fixed

$$\frac{\partial \mathbf{q}}{\partial z_i} = \dot{\mathbf{q}}|_{\dot{z}_i=1,\dot{z}_j=0;j\neq i} \tag{2.122}$$

According to this, the computation of $\mathbf{q_z}$ is a simple procedure. It is easy to demonstrate that the derivative of the position of a point in natural coordinates $\mathbf{r}_i$, with respect to a relative coordinate $z_j$ or a static modal amplitude $\eta_j$, is, in the case of a translational joint or mode, the actual vector defining the translational direction $\mathbf{u}_j$, and, in the case of a revolute joint, it is equal to $\mathbf{u}_j \times (\mathbf{r}_i - \mathbf{r}_j)$, where in this case $\mathbf{u}_j$ defines the axis of rotation. If the required derivative is that of a unit vector, it is obvious that its direction is not modified by any change in translational relative coordinates, whereas in the case of revolute joints, the unit vector is modified exactly as it happened to the position, i.e. $\mathbf{u}_j \times \mathbf{u}_i$.

In order to impose the fulfillment of the constraints at acceleration level, the evaluation of their second derivative is also needed. In this derivative, the term $\dot{\mathbf{\Phi}}_{\mathbf{z}}\dot{\mathbf{z}}$ will appear, and it can be expressed as,

$$\dot{\mathbf{\Phi}}_{\mathbf{z}}\dot{\mathbf{z}} = \left(\dot{\mathbf{\Phi}}_{\mathbf{q}}\mathbf{q}_{\mathbf{z}} + \mathbf{\Phi}_{\mathbf{q}}\dot{\mathbf{q}}_{\mathbf{z}}\right)\dot{\mathbf{z}} \tag{2.123}$$

The time derivative of the Jacobian in natural coordinates needs the velocities of the cut joints in natural coordinates, which are obtained, along with their positions, during the forward analysis. However, in most cases, since this Jacobian will be probably constant, the first term of this equation does not need to be evaluated. The only unknown term so far is the time derivative of $\mathbf{q}_{\mathbf{z}}$, which is directly obtained by differentiating its terms. For instance, the time derivative of $\mathbf{u}_j \times \mathbf{u}_i$ will be $\dot{\mathbf{u}}_j \times \mathbf{u}_i + \mathbf{u}_j \times \dot{\mathbf{u}}_i$, where all the appearing terms have been calculated in the forward analysis.

## 2.7 Projection of the dynamic terms

For the sake of clarity, the Cartesian and relative coordinates of all the $n_b$ rigid and flexible bodies in the system will be grouped into two vectors, in such a way that the reference coordinates are put together at the beginning, followed by the static modal amplitudes, then leaving the dynamic modal amplitudes at the end,

$$\begin{aligned} \mathbf{Z} &= \left\{ \mathbf{Z}_{r1}^{\mathsf{T}} \quad \cdots \quad \mathbf{Z}_{rn_b}^{\mathsf{T}} \quad \boldsymbol{\eta}_1^{\mathsf{T}} \quad \cdots \quad \boldsymbol{\eta}_{n_b}^{\mathsf{T}} \quad \boldsymbol{\xi}_1^{\mathsf{T}} \quad \cdots \quad \boldsymbol{\xi}_{n_b}^{\mathsf{T}} \right\}^{\mathsf{T}} \\ \dot{\mathbf{z}} &= \left\{ \dot{z}_1 \quad \cdots \quad \dot{z}_{n_b} \quad \dot{\boldsymbol{\eta}}_1^{\mathsf{T}} \quad \cdots \quad \dot{\boldsymbol{\eta}}_{n_b}^{\mathsf{T}} \quad \dot{\boldsymbol{\xi}}_1^{\mathsf{T}} \quad \cdots \quad \dot{\boldsymbol{\xi}}_{n_b}^{\mathsf{T}} \right\}^{\mathsf{T}} \end{aligned} \tag{2.124}$$

The total number of bodies $n_b$ includes the virtual massless bodies that are added when the kinematic joints have more than one degree of freedom. This number will coincide with the number of reference relative coordinates of the open–loop mechanism, since each body is allowed to have only one input joint, so that there will exist a $\mathbf{Z}_{ri}$ set of six reference Cartesian coordinates for each $\mathbf{z}_i$ relative reference coordinate, both associated to the same body $i$.

The mass matrix and forces vector of a body $i$ in Cartesian coordinates, after being obtained as explained in sections 2.4 and 2.5, can be partitioned according to the same convention used for the $\mathbf{Z}_i$ coordinates

$$\bar{\mathbf{M}}_i = \begin{bmatrix} \bar{\mathbf{M}}_{ri} & \bar{\mathbf{M}}_{r\eta i} & \bar{\mathbf{M}}_{r\xi i} \\ & \bar{\mathbf{M}}_{\eta i} & \bar{\mathbf{M}}_{\eta\xi i} \\ sym. & & \bar{\mathbf{M}}_{\xi i} \end{bmatrix}; \quad \bar{\mathbf{Q}}_i = \begin{Bmatrix} \bar{\mathbf{Q}}_{ri} \\ \bar{\mathbf{Q}}_{\eta i} \\ \bar{\mathbf{Q}}_{\xi i} \end{Bmatrix} \tag{2.125}$$

Unless only eigenmodes were used for the representation of the elastic deformation, for example in combination with a Buckens frame, the three lower blocks of the mass matrix, which contain the inertia of the elastic coordinates, do not form a diagonal matrix, since the Craig–Bampton modes are not mass–orthogonal (Géradin and Cardona, 2001). Only the set of dynamic modes is mass–orthogonal, since they are obtained as eigenmodes, so that the last block $\bar{\mathbf{M}}_{\xi i}$ is the only one that will be always diagonal. If the mode shapes are constant, however, these three blocks will be constant, as it will be seen in the next hapter.

The full mass matrix and force vector can be assembled for the whole system, according to the full Cartesian coordinates vector **Z** defined in Eq. (2.124), having the same structure as in the case of an individual body, with separated blocks for the reference coordinates, the static modal amplitudes, and the dynamic modal amplitudes.

$$\bar{\mathbf{M}} = \begin{bmatrix} \bar{\mathbf{M}}_r & \bar{\mathbf{M}}_{r\eta} & \bar{\mathbf{M}}_{r\xi} \\ & \bar{\mathbf{M}}_{\eta} & \bar{\mathbf{M}}_{\eta\xi} \\ sym. & & \bar{\mathbf{M}}_{\xi} \end{bmatrix}; \quad \bar{\mathbf{Q}} = \begin{Bmatrix} \bar{\mathbf{Q}}_r \\ \bar{\mathbf{Q}}_{\eta} \\ \bar{\mathbf{Q}}_{\xi} \end{Bmatrix} \tag{2.126}$$

The rigid bodies will have only the block corresponding to the reference coordinates, and the intermediate virtual bodies are massless, so that their mass matrices, which appear only in the reference coordinates block, are equal to zero. This global mass assembly is never carried out actually, it is used only to describe the recursive procedure used for projecting itself into the relative coordinates. The structure that it would have for the example of a double four–bar mechanism shown in Figure 2.11 can be seen in Eq. (A.3) of the Appendix.

In order to derive the equations of motion of the open–loop system, the virtual power principle is used. Given a set of virtual velocities $\mathbf{Z}^*$, the virtual power produced by them can be obtained as

$$\mathbf{Z}^{*\mathsf{T}}\left(\bar{\mathbf{M}}\dot{\mathbf{Z}} - \bar{\mathbf{Q}}\right) = 0 \tag{2.127}$$

Since the **Z** velocities are not independent, this expression does not mean that $\bar{\mathbf{M}}\dot{\mathbf{Z}} - \bar{\mathbf{Q}} = 0$. It must be expressed in the dependent relative coordinates, which are independent if the open–loop version of the mechanism is considered. At this point, the velocity transformation between **Z** and $\dot{\mathbf{z}}$ is introduced. By assembling all the recursive velocity relations **b** and $\boldsymbol{\varphi}$ defined when addressing the recursive kinematics, a position dependent matrix **R** can be defined, such that

$$\mathbf{Z} = \mathbf{R}\dot{\mathbf{z}} \tag{2.128}$$

Differentiating the velocities with respect to time, the accelerations can also be obtained,

$$\dot{\mathbf{Z}} = \mathbf{R}\ddot{\mathbf{z}} + \dot{\mathbf{R}}\dot{\mathbf{z}} \tag{2.129}$$

In this expression, the $\dot{\mathbf{R}}\dot{\mathbf{z}}$ term contains the position and velocity dependent **d** and $\boldsymbol{\gamma}$ terms, also defined when the kinematic relationships were established. The velocity transformation can be applied also to the virtual velocities, leading to the following expression if it is introduced in Eq. (2.127), along with the acceleration transformation defined in Eq. (2.129):

$$\dot{\mathbf{z}}^{*\mathsf{T}}\mathbf{R}^{\mathsf{T}}\left(\bar{\mathbf{M}}\mathbf{R}\ddot{\mathbf{z}} + \bar{\mathbf{M}}\dot{\mathbf{R}}\dot{\mathbf{z}} - \bar{\mathbf{Q}}\right) = 0 \tag{2.130}$$

and, since the $\mathbf{z}$ coordinates are independent, the expression into parentheses must be always equal to zero to fulfill the virtual power principle. This means that the equations of motion can be written, for the open–loop version of the mechanism, as

$$\mathbf{R}^{\mathsf{T}}\bar{\mathbf{M}}\mathbf{R}\ddot{\mathbf{z}} = \mathbf{R}^{\mathsf{T}}\left(\bar{\mathbf{Q}} - \bar{\mathbf{M}}\dot{\mathbf{R}}\dot{\mathbf{z}}\right) \tag{2.131}$$

Leading to the following expressions for the mass matrix and the generalized forces vector in relative coordinates:

$$\mathbf{M} = \mathbf{R}^{\mathsf{T}}\bar{\mathbf{M}}\mathbf{R}; \quad \mathbf{Q} = \mathbf{R}^{\mathsf{T}}\left(\bar{\mathbf{Q}} - \bar{\mathbf{M}}\dot{\mathbf{R}}\dot{\mathbf{z}}\right) \tag{2.132}$$

These operations can be performed very efficiently by taking advantage of the open loop topology. The $\mathbf{R}$ matrix is the result of assembling in matrix form the recursive velocity relationships defined for the open loop system, which makes its structure rather particular. It can be divided into blocks if the $\mathbf{Z}$ and $\dot{\mathbf{z}}$ coordinates are arranged as described in Eq. (2.124),

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_r & \mathbf{R}_\eta & 0 \\ 0 & \mathbf{I} & 0 \\ 0 & 0 & \mathbf{I} \end{bmatrix} \tag{2.133}$$

where $\mathbf{R}_r$ and $\mathbf{R}_\eta$ are two submatrices which relate the Cartesian rigid body velocities, $\mathbf{Z}_r$, to the relative velocities $\dot{\mathbf{z}}$ and to the time derivatives of the static modal amplitudes $\dot{\boldsymbol{\eta}}$ respectively. The first submatrix, the rigid body or reference part of $\mathbf{R}$, would be the $\mathbf{R}$ matrix of an equivalent rigid mechanism in the current deformed configuration.

The example mechanism described in Figure 2.11 is used to show how the $\mathbf{R}$ matrix terms look like. It consists, as pointed out before, of a double four–bar mechanism, in which the joints at points $c$ and $e$ have been cut, and bodies 2 and 3 are considered as flexible. The reference block of the projection matrix can be obtained as a product of a connectivity matrix $\mathbf{T}_r$, which depends exclusively on the topology of the mechanism, times a block diagonal matrix $\mathbf{R}_r^d$, containing the kinematic $\mathbf{b}$ terms associated to the reference relative coordinates

$$\mathbf{R}_r = \begin{bmatrix} \mathbf{I}_6 & 0 & 0 & 0 & 0 \\ \mathbf{I}_6 & \mathbf{I}_6 & 0 & 0 & 0 \\ \mathbf{I}_6 & \mathbf{I}_6 & \mathbf{I}_6 & 0 & 0 \\ \mathbf{I}_6 & \mathbf{I}_6 & 0 & \mathbf{I}_6 & 0 \\ \mathbf{I}_6 & \mathbf{I}_6 & 0 & \mathbf{I}_6 & \mathbf{I}_6 \end{bmatrix} \begin{bmatrix} \mathbf{b}_1 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{b}_2 & 0 & 0 & 0 \\ 0 & 0 & \mathbf{b}_3 & 0 & 0 \\ 0 & 0 & 0 & \mathbf{b}_4 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{b}_5 \end{bmatrix} = \mathbf{T}_r\mathbf{R}_r^d \tag{2.134}$$

The structure of the connectivity matrix $\mathbf{T}_r$ is such that each block column is associated to a joint, and it contains identity matrices in the block rows corresponding to those bodies which would be affected by a variation of the column's relative coordinate, while the rest of the relative coordinates remain fixed. The connectivity matrix $\mathbf{T}_r$ contains an identity at block $ij$ if and only if body $j$ precedes body $i$ when going from the root to the leaves, and they are both in the same branch. In the example mechanism, the zero blocks at positions (4,3) and (5,3) are due to the fact that body 3 is not in the same branch as bodies 4 and 5, so that a variation of $z_3$ would not affect them. If the bodies are numbered and sorted from the root to the leaves, this matrix will be always lower triangular, leading to an also lower triangular block structure for the $\mathbf{R}_r$ matrix, as it happens in the example.

A similar procedure can be employed to define the flexible transformation matrix, where in this case the connectivity matrix contains identity blocks when the corresponding body is affected by the elastic displacement of a boundary point, and the diagonal matrix contains the $\boldsymbol{\varphi}$ terms, which have one column for each static mode defined at a boundary point. In the example mechanism, these matrices are as follows,

$$\mathbf{R}_\eta = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \mathbf{I}_6 & 0 & 0 & 0 \\ \mathbf{I}_6 & \mathbf{I}_6 & \mathbf{I}_6 & 0 \\ \mathbf{I}_6 & \mathbf{I}_6 & 0 & 0 \\ \mathbf{I}_6 & \mathbf{I}_6 & 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\varphi}_2^A & 0 & 0 & 0 \\ 0 & \boldsymbol{\varphi}_2^B & 0 & 0 \\ 0 & 0 & \boldsymbol{\varphi}_3^B & 0 \\ 0 & 0 & 0 & \boldsymbol{\varphi}_3^C \end{bmatrix} = \mathbf{T}_\eta \mathbf{R}_\eta^d \tag{2.135}$$

and the full $\mathbf{R}$ matrix for this mechanism can be seen in the Appendix, although, as it happens to the $\bar{\mathbf{M}}$ matrix, it is never actually assembled.

### 2.7.1 Mass matrix projection

The mass matrix projection can be very efficiently calculated by dividing the products into blocks. By expanding the products needed to perform the mass matrix projection in Eq. (2.132), it can be found that the resulting mass matrix is the sum of three different terms, as pointed out by Funes et al. (2004),

$$\mathbf{R}^{\mathsf{T}}\bar{\mathbf{M}}\mathbf{R} = \left(\mathbf{R}^{\mathsf{T}}\bar{\mathbf{M}}\mathbf{R}\right)_r + \left(\mathbf{R}^{\mathsf{T}}\bar{\mathbf{M}}\mathbf{R}\right)_{rf} + \left(\mathbf{R}^{\mathsf{T}}\bar{\mathbf{M}}\mathbf{R}\right)_f \tag{2.136}$$

The first term contains the projections of the mass matrix blocks corresponding to the reference coordinates, and it is the only one which needs to be calculated in a recursive way,

$$\left(\mathbf{R}^{\mathsf{T}}\bar{\mathbf{M}}\mathbf{R}\right)_r = \begin{bmatrix} \mathbf{R}_r^{\mathsf{T}}\bar{\mathbf{M}}_r\mathbf{R}_r & \mathbf{R}_r^{\mathsf{T}}\bar{\mathbf{M}}_r\mathbf{R}_\eta & 0 \\ & \mathbf{R}_\eta^{\mathsf{T}}\bar{\mathbf{M}}_r\mathbf{R}_\eta & 0 \\ sym. & & 0 \end{bmatrix} \tag{2.137}$$

The second one is formed by projections of the blocks that couple the inertias of the reference and elastic coordinates. These terms are not calculated in a recursive way due to the fact that in the assembled $\bar{\mathbf{M}}$ matrix in Cartesian coordinates, the blocks of different bodies are not coupled

$$\left(\mathbf{R}^{\mathsf{T}}\bar{\mathbf{M}}\mathbf{R}\right)_{rf} = \begin{bmatrix} 0 & \mathbf{R}_r^{\mathsf{T}}\bar{\mathbf{M}}_{r\eta} & \mathbf{R}_r^{\mathsf{T}}\bar{\mathbf{M}}_{r\xi} \\ & \mathbf{R}_\eta^{\mathsf{T}}\bar{\mathbf{M}}_{r\eta} + \bar{\mathbf{M}}_{\eta r}\mathbf{R}_\eta & \mathbf{R}_\eta^{\mathsf{T}}\bar{\mathbf{M}}_{r\xi} \\ sym. & & 0 \end{bmatrix} \tag{2.138}$$

The blocks of the mass matrix involving only the elastic coordinates appear unchanged in the last term. They do not need to be projected since the elastic coordinates are the same in both the $\dot{\mathbf{z}}$ and $\mathbf{Z}$ coordinate sets,

$$\left(\mathbf{R}^{\mathsf{T}}\bar{\mathbf{M}}\mathbf{R}\right)_{f} = \begin{bmatrix} 0 & 0 & 0 \\ & \bar{\mathbf{M}}_\eta & \bar{\mathbf{M}}_{\eta\xi} \\ sym. & & \bar{\mathbf{M}}_\xi \end{bmatrix} \tag{2.139}$$

It can be seen, by observing the structure of the first term of the projected mass matrix, that there exists inertia coupling between the inertias of the reference and elastic coordinates, even between those of different bodies. The inertias of the elastic coordinates, except the dynamic modes block $\mathbf{M}_\xi$, are no longer constant after the projection into the relative coordinates. The efficient calculation of these two terms will now be addressed in detail, and the actual matrices obtained for the example double four–bar mechanism can be found in the Appendix.

### Projection of the reference mass matrix blocks

In order to calculate the three different blocks of the first term of the mass matrix, shown in Eq. (2.137), an optimal method that takes advantage of the structure of the mechanism, accumulating the inertias and forces from the leaves to the root, is used. A recursive accumulation of the reference mass matrices can be performed in such a way that the accumulated mass $\mathbf{M}_{ri}$ at a joint $i$ is the sum of the reference mass submatrices of all the bodies that would be affected by a variation of the relative coordinate $z_i$, while the rest of the coordinates remain fixed. In the example mechanism shown in Figure 2.11, the accumulation of reference mass matrices would be:

$$\begin{aligned} \mathbf{M}_{r5} &= \bar{\mathbf{M}}_{r5} \\ \mathbf{M}_{r4} &= \bar{\mathbf{M}}_{r4} + \mathbf{M}_{r5} \\ \mathbf{M}_{r3} &= \bar{\mathbf{M}}_{r3} \\ \mathbf{M}_{r2} &= \bar{\mathbf{M}}_{r2} + \mathbf{M}_{r3} + \mathbf{M}_{r4} \\ \mathbf{M}_{r1} &= \bar{\mathbf{M}}_{r1} + \mathbf{M}_{r2} \end{aligned} \tag{2.140}$$

Each accumulated $\mathbf{M}_{ri}$ matrix is equal to the sum of the reference mass matrices of the bodies with an identity block at the column $i$ of the connectivity matrix $\mathbf{T}_r$; therefore, the connectivity matrix can be used to automate this process, going from the rightmost column to the first and identifying the sums already performed in order to avoid repeated operations. There exists a second type of accumulation, in this case that due to the static modal amplitudes, which is performed exactly in the same way, taking into account the topology of the open–loop mechanism:

$$\begin{aligned}\mathbf{M}^B_{r3} &= \bar{\mathbf{M}}_{r3}\\ \mathbf{M}^B_{r2} &= \bar{\mathbf{M}}_{r4} + \bar{\mathbf{M}}_{r5} + \mathbf{M}^B_{r3}\\ \mathbf{M}^A_{r2} &= \bar{\mathbf{M}}_{r2} + \mathbf{M}^B_{r2}\end{aligned} \tag{2.141}$$

where each $\mathbf{M}^P_{ri}$ is the accumulated mass at the point $P$, considered as pertaining to body $i$, analogously as done in Eq. (2.140). This accumulated mass is that of the bodies which would be affected by a variation of the modal amplitudes at point $P$, due to a deformation of body $i$. As can be seen, $\mathbf{M}^B_{r3}$ is only the mass of body 3, since its own deformation does not affect any other body in the open–loop configuration, whereas $\mathbf{M}^B_{r2}$ is the accumulated mass of bodies 3, 4 and 5, i.e. those affected by a displacement of point $B$, produced by a deformation of body 2. Also, it is observed that point $c$ does not have an accumulated mass associated, since, in the open–loop mechanism, its deformation affects nothing but its own position.

The first block, $\mathbf{R}_r^{\mathsf{T}}\bar{\mathbf{M}}_r\mathbf{R}_r$, coincides with the mass matrix of an equivalent rigid body mechanism, in the current deformed configuration. Each one of its sub blocks will correspond to a pair of relative coordinates (or bodies) $i$ and $j$, and it can be calculated from the accumulated mass at joint $j$, and the kinematic terms of both joints $\mathbf{b}_i$ and $\mathbf{b}_j$ as follows:

$$\left(\mathbf{R}_r^{\mathsf{T}}\bar{\mathbf{M}}_r\mathbf{R}_r\right)_{ij} = \mathbf{b}_i^{\mathsf{T}}\mathbf{M}_{rj}\mathbf{b}_j; \quad i \le j \tag{2.142}$$

where only the upper triangle is considered, since the use of sparse symmetric solvers makes unnecessary to fill in the symmetric part of the matrix. Some of these terms are zero, however, since there is no inertia coupling between the reference coordinates of bodies pertaining to different branches of the mechanism. In the example mechanism, the blocks (3,4) and (3,5) are zero since body 3 is in a different branch from bodies 4 and 5. It is easy to automate the decision by examining the structure of $\mathbf{T}_r^{\mathsf{T}}\mathbf{T}_r$, since a block $ij$ in Eq. (2.142) is zero if the corresponding block of $\mathbf{T}_r^{\mathsf{T}}$ is also zero. The resulting matrix obtained by following this procedure in the example mechanism can be seen in the Appendix.

The projection of the reference mass matrices into the modal amplitudes is performed exactly in the same way, by using the recursive velocity relations associated to the static modes $\boldsymbol{\varphi}$ and the accumulated mass matrices defined in Eq. (2.141), so that the sub–block corresponding to a pair of boundary points will be

$$\left(\mathbf{R}_\eta^{\mathsf{T}}\bar{\mathbf{M}}_r\mathbf{R}_\eta\right)^{pq}_{ij} = \boldsymbol{\varphi}_i^{P\mathsf{T}}\mathbf{M}^P_{rj}\boldsymbol{\varphi}_j^Q; \quad i \le j, P \le Q \tag{2.143}$$

where the position of the null blocks can be determined analogously as done in the reference coordinates block, by examining the structure of $\mathbf{T}_\eta^{\mathsf{T}}$.

And the remaining block, which has the same nonzero blocks structure as $\mathbf{T}_r^{\mathsf{T}}\mathbf{T}_\eta$, has a mixed structure, using the mass accumulation from Eq. (2.140) in the lower triangle and that based on the static modes in the upper triangle, in such a way that,

$$\left(\mathbf{R}_r^{\mathsf{T}}\bar{\mathbf{M}}_r\mathbf{R}_\eta\right)_{ij}^P = \begin{cases} \mathbf{b}_i^{\mathsf{T}}\mathbf{M}_{rj}^P\boldsymbol{\varphi}_j^P & i \le j \\ \\ \mathbf{b}_i^{\mathsf{T}}\mathbf{M}_{rj}\boldsymbol{\varphi}_j^P & i > j \end{cases} \tag{2.144}$$

**Projection of the reference–elastic coupling blocks**

The terms appearing in Eq. (2.138) are simpler to obtain than those in Eq. (2.137), since they are not obtained from recursive accumulation. The structure of the assembled $\bar{\mathbf{M}}$ matrix is such that there is no coupling between the different bodies, so that the $\bar{\mathbf{M}}_{r\xi}$ and $\bar{\mathbf{M}}_{r\eta}$ blocks will be also an assembly of independent $\bar{\mathbf{M}}_{r\eta i}$ and $\bar{\mathbf{M}}_{r\xi i}$ blocks. The projections into the relative coordinates are

$$\left(\mathbf{R}_r^{\mathsf{T}}\bar{\mathbf{M}}_{r\eta}\right)_{ij} = \mathbf{b}_i^{\mathsf{T}}\bar{\mathbf{M}}_{r\eta j}; \quad \left(\mathbf{R}_r^{\mathsf{T}}\bar{\mathbf{M}}_{r\xi}\right)_{ij} = \mathbf{b}_i^{\mathsf{T}}\bar{\mathbf{M}}_{r\xi j} \tag{2.145}$$

being each term different from zero if a variation of $z_i$, while all other coordinates remain fixed, affects the position of body $j$. The remaining terms are obtained in the same way,

$$\left(\mathbf{R}_r^{\mathsf{T}}\bar{\mathbf{M}}_{r\eta}\right)_{ij}^P = \boldsymbol{\varphi}_i^{P\mathsf{T}}\bar{\mathbf{M}}_{r\eta j}; \quad \left(\mathbf{R}_r^{\mathsf{T}}\bar{\mathbf{M}}_{r\xi}\right)_{ij}^P = \boldsymbol{\varphi}_i^{P\mathsf{T}}\bar{\mathbf{M}}_{r\xi j} \tag{2.146}$$

with a structure determined in a similar way, although in this case the static modes at point $P$ of body $i$ are the coordinates that should affect the position of body $j$. All the blocks of the projected mass matrix are shown in the Appendix.

## 2.7.2 Projection of the forces vector

The generalized forces vector $\mathbf{Q}$ can be projected into the relative coordinates by means of Eq. (2.132),

$$\mathbf{Q} = \mathbf{R}^{\mathsf{T}}\left(\bar{\mathbf{Q}} - \bar{\mathbf{M}}\dot{\mathbf{R}}\dot{\mathbf{z}}\right) = \mathbf{R}^{\mathsf{T}}\bar{\mathbf{Q}}^t \tag{2.147}$$

The term $\bar{\mathbf{Q}}^t$ contains the forces computed for each body in Cartesian coordinates, assembled into a vector $\bar{\mathbf{Q}}$, to which velocity dependent inertia forces are added, due to the time dependency of the projection matrix $\mathbf{R}$. The first operation needed in order to calculate $\bar{\mathbf{Q}}^t$ is the evaluation of $\dot{\mathbf{R}}\dot{\mathbf{z}}$, which is totally straightforward since it is formed by the $\mathbf{d}_j$ and $\boldsymbol{\gamma}_j^P$ terms appearing in the acceleration kinematic relationships. This product can be divided into three blocks,

$$\left(\dot{\mathbf{R}}\dot{\mathbf{z}}\right)_r = \dot{\mathbf{R}}_r\dot{\mathbf{z}}_r + \dot{\mathbf{R}}_\eta\dot{\mathbf{z}}_\eta; \quad \left(\dot{\mathbf{R}}\dot{\mathbf{z}}\right)_\eta = 0; \quad \left(\dot{\mathbf{R}}\dot{\mathbf{z}}\right)_\xi = 0 \tag{2.148}$$

The only nonzero block, which corresponds to the reference coordinates, will contain a $6 \times 1$ block for each body. Each one of these blocks can be efficiently calculated as the accumulation, in this case from the root to the leaves, of the $\mathbf{d}_j$ and $\boldsymbol{\gamma}_j^P$ terms associated to the coordinates whose variation would affect the position of the body. In the example mechanism, this results

$$\begin{aligned}
\left(\dot{\mathbf{R}}\dot{\mathbf{z}}\right)_{r1} &= \mathbf{d}_1 \\
\left(\dot{\mathbf{R}}\dot{\mathbf{z}}\right)_{r2} &= \left(\dot{\mathbf{R}}\dot{\mathbf{z}}\right)_{r1} + \mathbf{d}_2 + \boldsymbol{\gamma}_2^A \\
\left(\dot{\mathbf{R}}\dot{\mathbf{z}}\right)_{r3} &= \left(\dot{\mathbf{R}}\dot{\mathbf{z}}\right)_{r2} + \mathbf{d}_3 + \boldsymbol{\gamma}_2^B + \boldsymbol{\gamma}_3^B \\
\left(\dot{\mathbf{R}}\dot{\mathbf{z}}\right)_{r4} &= \left(\dot{\mathbf{R}}\dot{\mathbf{z}}\right)_{r2} + \mathbf{d}_4 + \boldsymbol{\gamma}_2^B \\
\left(\dot{\mathbf{R}}\dot{\mathbf{z}}\right)_{r5} &= \left(\dot{\mathbf{R}}\dot{\mathbf{z}}\right)_{r4} + \mathbf{d}_5
\end{aligned} \tag{2.149}$$

This accumulated vector can be directly calculated when performing the position and velocity analyses during the forward loop, since the $\mathbf{d}_j$ and $\boldsymbol{\gamma}_j^P$ terms are not needed elsewhere. Once this vector is computed, the calculation of $\bar{\mathbf{Q}}^t$ is straightforward. Due to the block diagonal structure of $\bar{\mathbf{M}}_r$, each sub–block of $\bar{\mathbf{Q}}^t$ associated to a body $j$ will depend only on the terms of its individual reference mass matrix, and the corresponding block of the reference part of $\dot{\mathbf{R}}\dot{\mathbf{z}}$, so that

$$\bar{\mathbf{Q}}^t_{rj} = \bar{\mathbf{Q}}_{rj} - \bar{\mathbf{M}}_{rj}\left(\dot{\mathbf{R}}\dot{\mathbf{z}}\right)_{rj} \tag{2.150}$$

In the case of the modal amplitudes blocks, the calculation is the same, existing one block for each body $j$ having whether static or dynamic modes,

$$\bar{\mathbf{Q}}^t_{\eta j} = \bar{\mathbf{Q}}_{\eta j} - \bar{\mathbf{M}}^{\mathsf{T}}_{r\eta j}\left(\dot{\mathbf{R}}\dot{\mathbf{z}}\right)_{rj}\,; \quad \bar{\mathbf{Q}}^t_{\xi j} = \bar{\mathbf{Q}}_{\xi j} - \bar{\mathbf{M}}^{\mathsf{T}}_{r\xi j}\left(\dot{\mathbf{R}}\dot{\mathbf{z}}\right)_{rj} \tag{2.151}$$

Finally, the projection of $\mathbf{R}^{\mathsf{T}}\bar{\mathbf{Q}}^t$ is performed

$$\mathbf{Q} = \mathbf{R}^{\mathsf{T}}\bar{\mathbf{Q}}^t = \begin{bmatrix} \mathbf{R}_r^{\mathsf{T}}\bar{\mathbf{Q}}^t_r \\ \mathbf{R}_\eta^{\mathsf{T}}\bar{\mathbf{Q}}^t_r \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \bar{\mathbf{Q}}^t_\eta \\ \bar{\mathbf{Q}}^t_\xi \end{bmatrix} \tag{2.152}$$

The second term of this equation is already available, and the products in the first one are obtained by accumulation, from the leaves to the root, as has been done with the masses, i.e. calculating the accumulated rigid body forces both from the point of view of relative coordinates and modal amplitudes. A block $i$ of $\mathbf{R}_r^{\mathsf{T}}\bar{\mathbf{Q}}^t_r$ is obtained as

$$\left(\mathbf{R}_r^{\mathsf{T}}\bar{\mathbf{Q}}^t_r\right)_i = \mathbf{b}_i^{\mathsf{T}}\mathbf{Q}_{ri} \tag{2.153}$$

where the $\mathbf{Q}_{ri}$ is the accumulated force at joint $i$. These accumulations are as follows for the example mechanism,

$$\begin{aligned}
\mathbf{Q}_{r5} &= \bar{\mathbf{Q}}^t_{r5} \\
\mathbf{Q}_{r4} &= \bar{\mathbf{Q}}^t_{r4} + \mathbf{Q}_{r5} \\
\mathbf{Q}_{r3} &= \bar{\mathbf{Q}}^t_{r3} \\
\mathbf{Q}_{r2} &= \bar{\mathbf{Q}}^t_{r2} + \mathbf{Q}_{r3} + \mathbf{Q}_{r4} \\
\mathbf{Q}_{r1} &= \bar{\mathbf{Q}}^t_{r1} + \mathbf{Q}_{r2}
\end{aligned} \tag{2.154}$$

The calculation of the projection of the reference forces into the modal amplitudes will be obtained analogously

$$\left(\mathbf{R}_\eta^{\mathsf{T}} \bar{\mathbf{Q}}^t_r\right)_i^P = \varphi_i^{P\mathsf{T}} \mathbf{Q}^p_{ri} \tag{2.155}$$

There exists a second type of accumulation, in this case that due to the static modal amplitudes, which is performed exactly in the same way, taking into account the topology of the open–loop mechanism:

$$\begin{aligned}
\mathbf{Q}^B_{r3} &= \bar{\mathbf{Q}}^t_{r3} \\
\mathbf{Q}^B_{r2} &= \bar{\mathbf{Q}}^t_{r4} + \bar{\mathbf{Q}}^t_{r5} + \mathbf{Q}^B_{r3} \\
\mathbf{Q}^A_{r2} &= \bar{\mathbf{Q}}^t_{r2} + \mathbf{Q}^B_{r2}
\end{aligned} \tag{2.156}$$

## 2.8 Dynamic formalism

### 2.8.1 Equations of motion

This formulation uses dependent coordinates, along with a set of algebraic constraints, therefore the problem can be formulated by the classical Lagrange multipliers approach. This means that the equations of motion will be a set of $n$ second order ordinary differential equations, with $m$ added algebraic constraints that turn it into a set of differential algebraic equations, more specifically an index–3 DAE system

$$\begin{aligned}
\mathbf{M}\ddot{\mathbf{z}} + \boldsymbol{\Phi}_{\mathbf{z}}^{\mathsf{T}} \boldsymbol{\lambda} &= \mathbf{Q} \\
\boldsymbol{\Phi} &= 0
\end{aligned} \tag{2.157}$$

Most of the strategies for DAE integration are based on turning it into an ODE system, since there exist many well–known methods for their integration. The simplest approach is to differentiate the constraints twice with respect to time, leading to an index–1 DAE system that can be directly integrated (Goldstein, 1950),

$$\begin{bmatrix} \mathbf{M} & \boldsymbol{\Phi}_{\mathbf{z}}^{\mathsf{T}} \\ \boldsymbol{\Phi}_{\mathbf{z}} & 0 \end{bmatrix} \begin{Bmatrix} \ddot{\mathbf{z}} \\ \boldsymbol{\lambda} \end{Bmatrix} = \begin{Bmatrix} \mathbf{Q} \\ -\dot{\boldsymbol{\Phi}}_{\mathbf{z}}\dot{\mathbf{z}} - \dot{\boldsymbol{\Phi}}_t \end{Bmatrix} \tag{2.158}$$

The problem of this approach is that it does not impose the fulfillment of the constraints but their second derivative, thus causing a position drift in long simulations. A solution to this problem, proposed by Baumgarte (1972), consists of stabilizing the constraints by substituting $\ddot{\mathbf{\Phi}} = 0$ by

$$\ddot{\mathbf{\Phi}} + 2\xi\omega\dot{\mathbf{\Phi}} + \omega^2\mathbf{\Phi} = 0 \tag{2.159}$$

where the parameters $\xi$ and $\omega$ can be considered as the damping ratio and the natural frequency of a one degree of freedom vibrating system, thus $\xi$ commonly takes the value of 1 in order to obtain the fastest stabilization of the constraints possible, and the usual value of $\omega$ is 10. This leads to the following set of equations

$$\begin{bmatrix} \mathbf{M} & \mathbf{\Phi}_{\mathbf{z}}^{\mathsf{T}} \\ \mathbf{\Phi}_{\mathbf{z}} & 0 \end{bmatrix} \begin{Bmatrix} \ddot{\mathbf{z}} \\ \boldsymbol{\lambda} \end{Bmatrix} = \begin{Bmatrix} \mathbf{Q} \\ -\dot{\mathbf{\Phi}}_{\mathbf{z}}\dot{\mathbf{z}} - 2\xi\omega\dot{\mathbf{\Phi}} - \omega^2\mathbf{\Phi} \end{Bmatrix} \tag{2.160}$$

This method does not strictly impose the fulfillment of the constraints or their derivatives but a combination of them, and it adds a dissipative term $2\xi\omega\dot{\mathbf{\Phi}}$ that should be taken into account. Moreover, neither this method nor the classical approach can deal with systems having redundant constraints, and they need to integrate $n+m$ equations. In order to eliminate this problem, Bayo et al. (1988) proposed the penalty formulation, that approximates the Lagrange multipliers as the Baumgarte stabilization term, multiplied by a penalty factor $\alpha$, so that the force associated to the constraints is proportional to the violation of the constraints and their derivatives,

$$\boldsymbol{\lambda} = \alpha\left(\ddot{\mathbf{\Phi}} + 2\xi\omega\dot{\mathbf{\Phi}} + \omega^2\mathbf{\Phi}\right) \tag{2.161}$$

The penalty factor takes large values, ranging from $10^7$ to $10^{10}$, in such a way that a small violation of a constraint introduces a large opposing force. The resulting system is, after substituting the approximation of $\boldsymbol{\lambda}$ into the equations of motion

$$\left(\mathbf{M} + \mathbf{\Phi}_{\mathbf{z}}^{\mathsf{T}}\alpha\mathbf{\Phi}_{\mathbf{z}}\right)\ddot{\mathbf{z}} = \mathbf{Q} - \mathbf{\Phi}_{z}^{\mathsf{T}}\alpha\left(\dot{\mathbf{\Phi}}_{\mathbf{z}}\dot{\mathbf{z}} + 2\xi\omega\dot{\mathbf{\Phi}} + \omega^2\mathbf{\Phi}\right) \tag{2.162}$$

where the leading matrix is only $n \times n$. An alternative to this approach, that obtains the exact values of the Lagrange's multipliers, but retaining the advantages of the penalty formulation such as the integration of only $n$ equations and the possibility of dealing with redundant constraints, is the augmented Lagrangian formulation (García de Jalón and Bayo, 1994), which had been already used by Vanderplaats (1984) in optimization problems. This formulation consists of the penalty formulation, with an added iteration to calculate the exact value of the Lagrange multipliers. The equations of motion are stated as follows

$$\mathbf{M}\ddot{\mathbf{z}} + \mathbf{\Phi}_{\mathbf{z}}^{\mathsf{T}}\alpha\left(\dot{\mathbf{\Phi}}_{\mathbf{z}}\dot{\mathbf{z}} + 2\xi\omega\dot{\mathbf{\Phi}} + \omega^2\mathbf{\Phi}\right) + \mathbf{\Phi}_{\mathbf{z}}^{\mathsf{T}}\boldsymbol{\lambda}^* = \mathbf{Q} \tag{2.163}$$

where $\boldsymbol{\lambda}^*$ is the Lagrange multipliers vector, obtained from an iteration process carried out within each time step,

$$\boldsymbol{\lambda}^*_{i+1} = \boldsymbol{\lambda}^*_i + \alpha \left( \ddot{\boldsymbol{\Phi}} + 2\xi\omega\dot{\boldsymbol{\Phi}} + \omega^2\boldsymbol{\Phi} \right) \quad i = 0, 1, 2, \ldots \tag{2.164}$$

which starts with $\boldsymbol{\lambda}^*_0$ equal to zero, or to the value of $\boldsymbol{\lambda}^*$ obtained in the previous time–step.

Other methods for turning the DAE system into an ODE are those based on coordinate partitioning (Wehage and Haug, 1982), which separate the dependent coordinates into a set of independent and another set of dependent coordinates, and integrate only the independent coordinates. One of these methods, proposed by García de Jalón and Bayo (1994), uses a velocity transformation to obtain the dependent coordinates as a function of the independent ones. These methods integrate a minimum set of variables, but there exist some disadvantages. On the one side, the velocity transformation must be carried out at every time–step, leading in this case to a double velocity transformation, first from Cartesian to dependent relative, and then from dependent relative to independent relative coordinates. On the other side, the chosen set of independent coordinates may not be valid for all the positions of the mechanism, so that a checking must be carried out at every time–step, and in case the set is not valid, a new transformation must be defined, which implies the integration of a different set of variables and, therefore, restarting the integration.

The actual formulation used is a variation of the augmented Lagrangian formulation, but stating it as an index–3 instead of an index–1 formulation. As opposed to the augmented Lagrangian, the formulation used in this thesis applies the penalty only to the constraints vector, so that their fulfillment is totally assured. The final equations of motion are as follows:

$$\mathbf{M}\ddot{\mathbf{z}} + \boldsymbol{\Phi}_{\mathbf{z}}^{\mathsf{T}}\alpha\boldsymbol{\Phi} + \boldsymbol{\Phi}_{\mathbf{z}}^{\mathsf{T}}\boldsymbol{\lambda}^* = \mathbf{Q} \tag{2.165}$$

$$\boldsymbol{\lambda}^*_{i+1} = \boldsymbol{\lambda}^*_i + \alpha\boldsymbol{\Phi} \quad i = 1, 2, \ldots \tag{2.166}$$

Since this method only enforces the fulfillment of the constraints at position level, the velocities and accelerations are later projected in order to enforce them to fulfill the constraints at velocity and acceleration level, using mass–orthogonal projections as proposed by Bayo and Ledesma (1996). As it will be seen in the numerical integration section, the projections actually used in this formulation are modified in order to improve the efficiency (Cuadrado et al., 2000), so that they are no longer mass–orthogonal.

The full procedure for the assembly of the equations of motion is schematized in Figure 2.19. Starting from known values of the relative positions and velocities, the first step is to perform the forward position and velocity analyses, in order to obtain the positions and velocities of the bodies and joints in natural coordinates. This allows for calculating, on the one side, the kinematic and inertia terms, expressed in the Cartesian coordinates **Z** and, on the other side, the kinematic constraints and their Jacobian matrix. The kinematic and inertia terms are used to calculate the projection

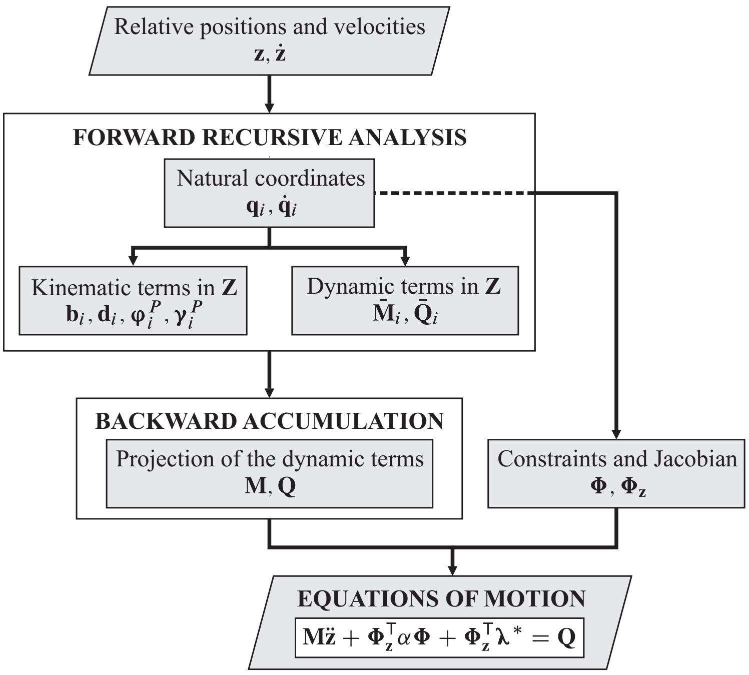

Relative positions and velocities
$\mathbf{z}, \dot{\mathbf{z}}$
FORWARD RECURSIVE ANALYSIS
Natural coordinates
$\mathbf{q}_i, \dot{\mathbf{q}}_i$
Kinematic terms in $\mathbf{Z}$
$\mathbf{b}_i, \mathbf{d}_i, \boldsymbol{\varphi}_i^P, \boldsymbol{\gamma}_i^P$
Dynamic terms in $\mathbf{Z}$
$\bar{\mathbf{M}}_i, \bar{\mathbf{Q}}_i$
BACKWARD ACCUMULATION
Projection of the dynamic terms
$\mathbf{M}, \mathbf{Q}$
Constraints and Jacobian
$\boldsymbol{\Phi}, \boldsymbol{\Phi}_\mathbf{z}$
EQUATIONS OF MOTION
$\mathbf{M}\ddot{\mathbf{z}} + \boldsymbol{\Phi}_\mathbf{z}^\mathsf{T}\alpha\boldsymbol{\Phi} + \boldsymbol{\Phi}_\mathbf{z}^\mathsf{T}\boldsymbol{\lambda}^* = \mathbf{Q}$

Figure 2.19: Procedure for building the equations of motion.

of the dynamic terms into the relative coordinates, in a recursive backward process, which is one of the keys of the formulation. Once the projection has been performed, all the terms of the equations of motion are available in relative coordinates.

## 2.8.2 Time integration

Once the equations of motion have been transformed into a second order ODE system, the choice of numerical integrators is considerably widened. The structural integrators, originally developed for their use in the field of structural dynamics (Bathe, 1995), have been adapted and successfully used by many authors like García de Jalón and Bayo (1994), Géradin and Cardona (2001), or Cuadrado et al. (2000), for the integration of the equations of motion in multibody systems. In this thesis, the Newmark dissipative integrator (Newmark, 1959), an implicit integrator from the structural family, has been chosen for the time integration of the equations of motion due to the excellent results obtained by using it in rigid multibody systems, as shown by the work of Cuadrado et al. (2004b) and Dopico (2004). This integrator can introduce a variable amount of numerical damping, based on the choice of a parameter $\xi$ and, in the limit case when there is no damping, the integrator becomes the well–known trapezoidal rule, which is the form actually used in this work. The numerical damping is later introduced in order to test the efficiency that can be reached, although all the examples are integrated without problems with the trapezoidal rule.

**Combination of the equations of motion with the numerical integrator**

The difference equations of a general Newmark integrator, which yield the position and velocity at the time–step $n+1$ by using the accelerations as primary variables, are as follows for a time–step $h$:

$$\mathbf{z}_{n+1} = \mathbf{z}_n + h\dot{\mathbf{z}}_n + \frac{h^2}{2}\left[(1-2\beta)\,\ddot{\mathbf{z}}_n + 2\beta\ddot{\mathbf{z}}_{n+1}\right] \tag{2.167}$$

$$\dot{\mathbf{z}}_{n+1} = \dot{\mathbf{z}}_n + h\left[(1-\gamma)\,\ddot{\mathbf{z}}_n + \gamma\ddot{\mathbf{z}}_{n+1}\right] \tag{2.168}$$

In the dissipative subfamily of the Newmark integrators, the parameters $\beta$ and $\gamma$ are allowed to take the following values, with $\xi \leq 0$, for keeping the integrator in its unconditionally stable zone,

$$\beta = \frac{(1-\xi)^2}{4}; \quad \gamma = \frac{1-2\xi}{2} \tag{2.169}$$

For $\xi = 0$, $\beta$ is equal to 0.25 and $\gamma$ to 0.5, so that the integrator becomes the trapezoidal rule, which adds no numerical damping and shows second order precision. Lowering the value of $\xi$ increases the amount of numerical damping, thus improving the stability although at the price of becoming an integrator of only first order precision.

In order to combine the equations of the integrator with the equations of motion stated according to the described index–3 augmented Lagrangian formulation, the positions are used as primary variables, so that the equations of the integrator turn into

$$\dot{\mathbf{z}}_{n+1} = \frac{\gamma}{\beta h}\mathbf{z}_{n+1} - \hat{\dot{\mathbf{z}}}_n \tag{2.170}$$

$$\ddot{\mathbf{z}}_{n+1} = \frac{1}{\beta h^2}\mathbf{z}_{n+1} - \hat{\ddot{\mathbf{z}}}_n \tag{2.171}$$

where

$$\hat{\dot{\mathbf{z}}}_n = \frac{\gamma}{\beta h}\mathbf{z}_n + \left(\frac{\gamma}{\beta} - 1\right)\dot{\mathbf{z}}_n + \left(\frac{\gamma}{2\beta} - 1\right)h\ddot{\mathbf{z}}_n \tag{2.172}$$

$$\hat{\ddot{\mathbf{z}}}_n = \frac{1}{\beta h^2}\mathbf{z}_n + \frac{1}{\beta h}\dot{\mathbf{z}}_n + \left(\frac{1}{2\beta} - 1\right)\ddot{\mathbf{z}}_n \tag{2.173}$$

If dynamic equilibrium is imposed at step $n+1$ by combining the equations of the integrator (2.170) and (2.171) with the equations of motion (2.165), a nonlinear system of algebraic equations $\mathbf{f}\,(\mathbf{z}_{n+1}) = 0$ must be solved for the relative positions. This system can be solved by using the Newton–Raphson iteration with the following approximated tangent matrix and residual vector:

$$\mathbf{f}_{\mathbf{z}} \approx \mathbf{M} + \gamma h\mathbf{C} + \beta h^2\left(\mathbf{\Phi}_{\mathbf{z}}^{\mathsf{T}}\alpha\mathbf{\Phi}_{\mathbf{z}} + \mathbf{K}\right) \tag{2.174}$$

$$\mathbf{f} = \beta h^2\left(\mathbf{M}\ddot{\mathbf{q}} + \mathbf{\Phi}_{\mathbf{z}}^{\mathsf{T}}\alpha\mathbf{\Phi} + \mathbf{\Phi}_{\mathbf{z}}^{\mathsf{T}}\boldsymbol{\lambda}^* - \mathbf{Q}\right) \tag{2.175}$$

being **K** and **C** the already described generalized stiffness and damping matrices

$$\mathbf{K} = -\frac{\partial \mathbf{Q}}{\partial \mathbf{z}}; \quad \mathbf{C} = -\frac{\partial \mathbf{Q}}{\partial \dot{\mathbf{z}}} \tag{2.176}$$

The procedure for obtaining the position at step $n+1$ starts by calculating firstly an initial guess

$$\mathbf{z}_{n+1}^{0} = \mathbf{z}_n + h\dot{\mathbf{z}}_n + \frac{h^2}{2}\ddot{\mathbf{z}}_n \tag{2.177}$$

then obtaining the corresponding velocity and acceleration at $n+1$ from the equations of the integrator (2.170) and (2.171). Then, the Newton–Raphson iteration is performed, by calculating the terms of the tangent matrix and the residual, obtaining the correction in positions, and obtaining the new velocities and accelerations again from the equations of the integrator, until the norm of the correction or the residual goes under a specified tolerance. The iteration used in the augmented Lagrangian formulation for updating the Lagrange multipliers of Eq. (2.166) can be performed along with the position correction loop.

## Projection of velocities and accelerations

The solution of $\mathbf{f}(\mathbf{z}_{n+1}) = 0$ obtained after the convergence of the corrector yields a position vector that fulfills the dynamic equilibrium equations, along with the kinematic constraints at position level ($\mathbf{\Phi} = 0$). However, the velocities and accelerations thus obtained are not guaranteed to satisfy the time derivatives of the constraints, since they have not been imposed. Bayo and Ledesma (1996) proposed a solution that consists of projecting the velocities and accelerations, in such a way that the new values fulfill the first and second time derivatives of the constraints. The projection of velocities is performed by solving the following minimization problem,

$$\begin{aligned} &\min V = \frac{1}{2}\left(\dot{\mathbf{z}} - \dot{\mathbf{z}}^*\right)^{\mathsf{T}} \mathbf{M}\left(\dot{\mathbf{z}} - \dot{\mathbf{z}}^*\right) \\ &\text{s.t.} \quad \dot{\mathbf{\Phi}} = 0 \end{aligned} \tag{2.178}$$

where $\dot{\mathbf{z}}^*$ are the velocities obtained after the convergence of the Newton–Raphson iteration, and $\dot{\mathbf{z}}$ are the updated values obtained after solving the minimization problem. This problem consists then in finding the velocities that fulfill the constraints at velocity level, while producing the smallest possible deviation from the original values, weighted by the mass matrix. This problem can be solved by means of the Lagrange multipliers method, although, in order to avoid an iterative process, a penalty approach is more convenient. This leads to solving the following linear system for $\dot{\mathbf{z}}$:

$$\left(\mathbf{M} + \mathbf{\Phi}_{\mathbf{z}}^{\mathsf{T}} \alpha \mathbf{\Phi}_{\mathbf{z}}\right) \dot{\mathbf{z}} = \mathbf{M}\dot{\mathbf{z}}^* - \mathbf{\Phi}_{\mathbf{z}}^{\mathsf{T}} \alpha \mathbf{\Phi}_t \tag{2.179}$$

A similar procedure can be used for projecting the accelerations. The minimization problem of Eq. (2.178) can be applied to them, in this case subject to the restriction

$\ddot{\mathbf{\Phi}} = 0$, leading to a linear system with the same leading matrix and a different right–hand–side

$$\left(\mathbf{M} + \mathbf{\Phi}_{\mathbf{z}}^{\mathsf{T}} \alpha \mathbf{\Phi}_{\mathbf{z}}\right) \ddot{\mathbf{z}} = \mathbf{M}\ddot{\mathbf{z}}^{*} - \mathbf{\Phi}_{\mathbf{z}}^{\mathsf{T}} \alpha \left(\dot{\mathbf{\Phi}}_{\mathbf{z}} \dot{\mathbf{z}} + \dot{\mathbf{\Phi}}_{t}\right) \tag{2.180}$$

where the $\dot{\mathbf{\Phi}}_{\mathbf{z}} \dot{\mathbf{z}} + \dot{\mathbf{\Phi}}_{t}$ terms might be previously updated by using the already projected velocities.

Instead of using the mass matrix as the weighting matrix of the objective function, Cuadrado et al. (2000) proposed to use a different one that makes the leading matrix of Eqs. (2.179) and (2.180) become the tangent matrix (2.174), already assembled and factorized when calculating the positions in the Newton–Raphson iteration. The proposed weighting matrix is defined as

$$\mathbf{W} = \mathbf{M} + \gamma h \mathbf{C} + \beta h^{2} \mathbf{K} \tag{2.181}$$

so that if the restrictions of both the velocity and acceleration minimization problems are scaled by a factor of $\beta h^2$, the linear systems to be solved in order to perform the projections become,

$$\mathbf{f}_{\mathbf{z}} \dot{\mathbf{z}} = \mathbf{W}\dot{\mathbf{z}}^{*} - \beta h^{2} \mathbf{\Phi}_{\mathbf{z}}^{\mathsf{T}} \alpha \mathbf{\Phi}_{t} \tag{2.182}$$

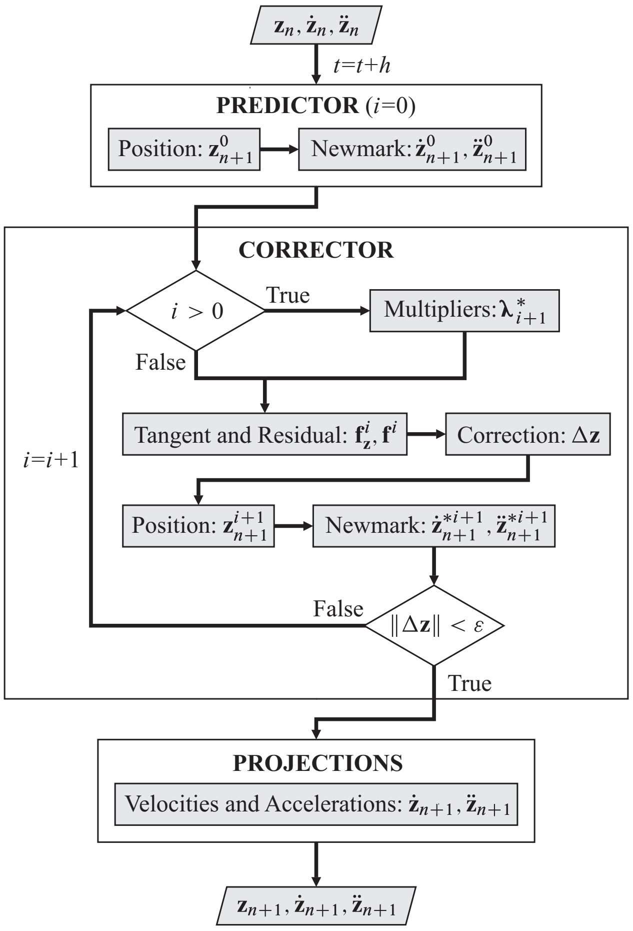
$$\mathbf{f}_{\mathbf{z}} \ddot{\mathbf{z}} = \mathbf{W}\ddot{\mathbf{z}}^{*} - \beta h^{2} \mathbf{\Phi}_{\mathbf{z}}^{\mathsf{T}} \alpha \left(\dot{\mathbf{\Phi}}_{\mathbf{z}} \dot{\mathbf{z}} + \dot{\mathbf{\Phi}}_{t}\right) \tag{2.183}$$

This allows to make use of the last factorization of the tangent matrix performed during the Newton–Raphson iteration for evaluating the projections, thus reducing their impact on the required CPU–time.

### Algorithm

In order to initiate the integration process, the index–3 augmented Lagrangian formulation combined with a Newmark integrator here described needs the positions, velocities and accelerations at the first time–step. The positions and velocities can be obtained from those of the degrees of freedom, that must be known at $t = 0$, by performing a standard kinematic analysis. The positions are obtained by means of a Newton–Raphson iteration in order to solve the nonlinear system $\mathbf{\Phi} = 0$ for the unknown variables. The unknown velocities, on the other side, must satisfy the equation $\mathbf{\Phi}_{\mathbf{z}} \dot{\mathbf{z}} = 0$, in which the Jacobian appears again. Once the positions and velocities are known, the accelerations at the initial instant can be obtained by means of any of the methods previously described, such as the penalty method, described in Eq. (2.162).

Then, the procedure for obtaining the positions, velocities and accelerations at step $n + 1$ from those at step $n$ can be seen in Figure 2.20. This procedure is divided into three main steps: first, an initial guess is calculated for step $n + 1$ by means of the predictor; then, a corrector loop calculates the positions, velocities and accelerations that satisfy the equations of motion; and finally, the velocities and accelerations are projected in order to minimize the violation of the constraints at velocity and acceleration level. The resulting algorithm for obtaining the positions, velocities and accelerations



$\mathbf{z}_n, \dot{\mathbf{z}}_n, \ddot{\mathbf{z}}_n$
$t{=}t{+}h$
PREDICTOR ($i{=}0$)
Position: $\mathbf{z}^0_{n+1}$
Newmark: $\dot{\mathbf{z}}^0_{n+1}, \ddot{\mathbf{z}}^0_{n+1}$
CORRECTOR
$i > 0$
True
Multipliers: $\boldsymbol{\lambda}^*_{i+1}$
False
Tangent and Residual: $\mathbf{f}^i_{\mathbf{z}}, \mathbf{f}^i$
Correction: $\Delta\mathbf{z}$
$i{=}i{+}1$
Position: $\mathbf{z}^{i+1}_{n+1}$
Newmark: $\dot{\mathbf{z}}^{*i+1}_{n+1}, \ddot{\mathbf{z}}^{*i+1}_{n+1}$
False
$\|\Delta\mathbf{z}\| < \varepsilon$
True
PROJECTIONS
Velocities and Accelerations: $\dot{\mathbf{z}}_{n+1}, \ddot{\mathbf{z}}_{n+1}$
$\mathbf{z}_{n+1}, \dot{\mathbf{z}}_{n+1}, \ddot{\mathbf{z}}_{n+1}$

Figure 2.20: Time integration of the equations of motion.

at $n + 1$ is as follows:

1. Increase time: $t = t + h$.
2. Predictor: $i = 0$
   (a) Positions $\mathbf{z}^0_{n+1}$ from Eq. (2.177).
   (b) Velocities $\dot{\mathbf{z}}^0_{n+1}$ and accelerations $\ddot{\mathbf{z}}^0_{n+1}$ from Eqs. (2.170) and (2.171).
3. Corrector:
   (a) If $i > 0$, update Lagrange's multipliers (Eq. (2.166)).
   (b) Evaluate tangent matrix $\mathbf{f}^i_{\mathbf{z}}$ and residual $\mathbf{f}^i$ by means of Eqs. (2.174) and (2.175).
   (c) Solve the linear system $\mathbf{f}^i_{\mathbf{z}}\Delta\mathbf{z} = \mathbf{f}^i$ for $\Delta\mathbf{z}$.
   (d) Calculate the corrected $\mathbf{z}^{i+1}_{n+1} = \mathbf{z}^i_{n+1} + \Delta\mathbf{z}$.

   (e) Update velocities and accelerations with Eqs. (2.170) and (2.171).
   (f) If the error $\|\Delta\mathbf{z}\|$ is larger than the tolerance $\varepsilon$, $i = i + 1$; go to step 3.

4. Projections:
   (a) Projection of relative velocities: Eq. (2.182).
   (b) Velocity analysis for Cartesian coordinates, update $\dot{\mathbf{\Phi}}_{\mathbf{z}}\dot{\mathbf{z}}$ and $\dot{\mathbf{\Phi}}_t$.
   (c) Projection of relative accelerations: Eq. (2.183).

In order to evaluate all the terms needed at the steps 3(a) and 3(b), the procedure described in Figure 2.19 for obtaining $\mathbf{M}$, $\mathbf{Q}$, $\mathbf{\Phi}$, $\mathbf{\Phi}_{\mathbf{z}}$, $\dot{\mathbf{\Phi}}_{\mathbf{z}}\dot{\mathbf{z}}$ and $\dot{\mathbf{\Phi}}_t$ must be used. The $\mathbf{K}$ and $\mathbf{C}$ matrices are directly obtained by following the procedure described in their corresponding sections.

## 2.9 Numerical examples

Three examples, already used for a natural vs. relative coordinates comparison in rigid multibody systems (Cuadrado et al., 2004a; Dopico, 2004), have been implemented in the flexible case through both the formulation in natural coordinates described in Gutiérrez (2003), Cuadrado et al. (2004c), and that in relative coordinates here developed. The first one is a planar double four–bar mechanism formed by five identical bars, the second one is the front left suspension of the Bombardier Iltis vehicle (Frik et al., 1993), and the third one is the full Iltis vehicle. Performance measurements have been carried out with different numbers of flexible elements, in order to evaluate the influence of such parameter in each formulation. The first two examples were implemented in MATLAB, so the CPU–times should not be considered as a reference for the efficiency, but only for comparison between formulations. The Iltis vehicle is programmed in FORTRAN, obtaining faster simulations despite of being a much larger system.

### 2.9.1 Double four–bar mechanism

The system consists of five identical steel bars, as seen in Figure 2.21. Each of them has unit length and mass, and they are all connected by revolute joints. All bars can be considered individually as rigid or flexible, modeled in the flexible case by 10 beam elements, with one axial static, one bending static, and two bending dynamic modes. The closed loops have been cut as done in the double four–bar mechanism shown in the example in Figure 2.11.

The number of coordinates increases as more flexible bars are considered in the system, being this increment different in the absolute (natural coordinates) and relative formulations, because the latter does not include as coordinates the unit vectors of the local frames. The number of coordinates for each formulation and number of flexible bodies is given in Table 2.1. This number tends to be double in natural coordinates when more flexible bodies are considered, because each flexible body adds four modal

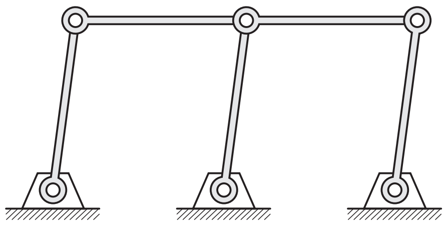


Figure 2.21: Double four–bar mechanism.

amplitudes plus four unit vector components, while, in relative coordinates, each body adds only the four modal amplitudes.

Table 2.1: Number of system coordinates in the first example.

| # flexible bars | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| Absolute | 6 | 13 | 20 | 27 | 34 | 41 |
| Relative | 5 | 8 | 11 | 14 | 17 | 20 |

The system is subject to gravity, and its leftmost ground–attached bar receives an initial velocity of 1 rad/s in clockwise direction. Motion is integrated during 5 s –the time to approximately complete 2.7 revolutions– by using the trapezoidal rule ($\xi = 0$), with a time–step of 10 ms. The CPU–times required for the integration are those provided in Table 2.2. As it may be seen in the Table, the CPU–times reduce their difference when more bodies are considered flexible. In the rigid case, the absolute formulation is five times faster, while, in the fully flexible model, the relative formulation needs only 37% more time for integration, probably due to the proportionally lower number of coordinates mainly.

Table 2.2: CPU–times (s) in the first example.

| # flexible bars | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| Absolute | 0.91 | 3.30 | 6.24 | 9.61 | 11.51 | 15.22 |
| Relative | 4.85 | 9.11 | 12.62 | 15.74 | 17.74 | 20.92 |

The energy conservation has been checked to assess the precision of both formulations. In this example, the total energy fluctuates each time the system passes through the singular position, so instead of the energy loss at the end of the simulation, its mean value along the whole five seconds has been measured. The relative method, whose mean energy loss ranges from 0.004 J in the rigid case to 0.013 J in the fully flexible case, has showed a higher precision than the absolute one, which loses a mean of 0.021 J in both cases.

## 2.9.2 Iltis suspension

The Bombardier Iltis vehicle (Frik et al., 1993) is a well–known benchmark system for simulation software. The second example chosen for the comparison of formulations is its left front suspension, shown in Figure 2.22. The model is formed by a total of five bodies, having three of them the possibility of being flexible or not. The bodies that can be considered as flexible, namely the A–arm, the upper link and the track rod, appear in dashed lines in the figure. The A–arm is connected to the chassis through a revolute joint, and to the lower side of the stub axle by means of a spherical joint. The stub axle is considered as a rigid body, and it is connected to both the upper link and the track rod by spherical joints. The upper link is connected to the chassis by a revolute joint, whereas the track rod uses a spherical one for that purpose. Three force elements appear in this system: a shock absorber, acting between the A–arm and the chassis; a spring, which connects the upper joint of the stub axle to the chassis; and the wheel, which is modeled as a simple spring dependent on the distance to the floor.

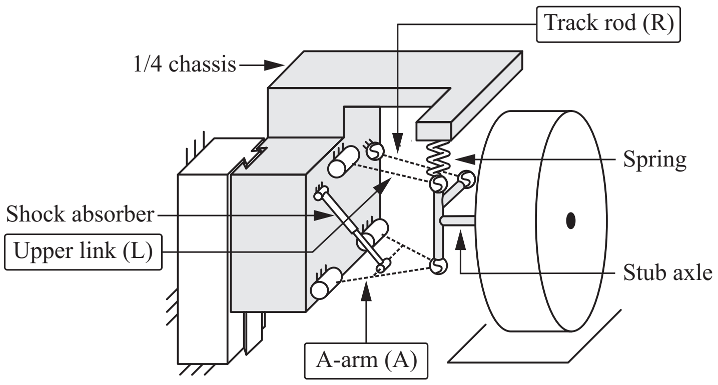


Figure 2.22: Sketch of the Iltis left front suspension.

The modeling used in relative coordinates is described here. The system is again a closed loop and, in order to transform it into an open–loop system, the two upper spherical joints at the top of the stub axle, i.e. the connections to the upper link and the track rod, are cut. Then, the topology of the mechanism is as follows:

- A translational joint $z_1$, defined in the vertical direction, connects the chassis to the inertial system.
- The A–arm is connected to the chassis by means of a revolute joint $z_2$. Then, three revolute joints $z_3$, $z_4$ and $z_5$ are used to model the spherical joint between the A–arm and the stub axle. This completes the first branch that starts from the chassis.
- The second branch consists of the upper link, attached to the chassis by means of a revolute joint $z_6$.

- The track rod is pinned to the chassis, having two degrees of freedom $z_7$ and $z_8$. This is the third and last branch of the mechanism.

Six constraints are established to impose the closure of the cut joints, since both are of the spherical type.

In this case, modal damping has been added to the flexible elements, using a modal damping matrix equal to 1% of their modal stiffness matrix. The flexible bodies are modeled by using a tangent frame at the input joint, which is, for the three of them, placed at the connections to the chassis. Then, the A–arm has two off–plane static modes, defined at the connections to the shock absorber and to the stub axle, plus the first two dynamic modes to complete the deformation field. It has been modeled by means of a finite element model, using 3D beam elements, and is formed by two bars, discretized into 10 elements each, that converge at the connection to the stub axle, plus an additional rigid element that models the connection to the shock absorber. The other two elements are simple beams, discretized into 10 elements, and they both use the same modal reduction, consisting of two transversal static modes defined at their tips, plus the first four dynamic modes.

The number of coordinates needed in all the possible combinations of formulation and flexible bodies are shown in Table 2.3. Being this a three–dimensional system, the consideration of a body as flexible in natural coordinates can add up to 9 coordinates in addition to the modal amplitudes, since a local reference frame needs three unit vectors, although in practice some of the vectors can be shared between neighbor elements thus reducing the total number of variables. In this case the number of coordinates in the absolute model is around three times larger, and, unlike the previous example, this relation remains almost constant with the number of flexible bodies.

Table 2.3: Number of system coordinates in the second example.

| Flexible elements | None | A | L | R | A+L | A+R | L+R | All |
|---|---|---|---|---|---|---|---|---|
| Absolute | 35 | 45 | 47 | 50 | 57 | 60 | 62 | 72 |
| Relative | 8 | 12 | 14 | 14 | 18 | 18 | 20 | 24 |

The suspension reaches equilibrium and then runs down a 0.2 m step at $t$=2 s. The integration, performed again with the trapezoidal rule with a time step of 10 ms, is carried out for 5 seconds until the suspension reaches equilibrium again. The time history of the vertical coordinate of the chassis, as well as that of the wheel center, with all possible flexible elements, are plotted in Figure 2.23, showing a very good agreement between the two formulations.

The CPU–times required to carry out the simulation are displayed in Table 2.4, as well as in Figure 2.24. In order to more clearly show the influence of the number of bodies, Figure 2.25 shows the CPU–times for the cases of one or two flexible bodies obtained as the mean values of the three different combinations. It is observed that the method in relative coordinates is now faster than that in absolute coordinates, and the difference grows as the number of flexible bodies is increased. However, the improvement is not very significant, taking into account that the implementation of the

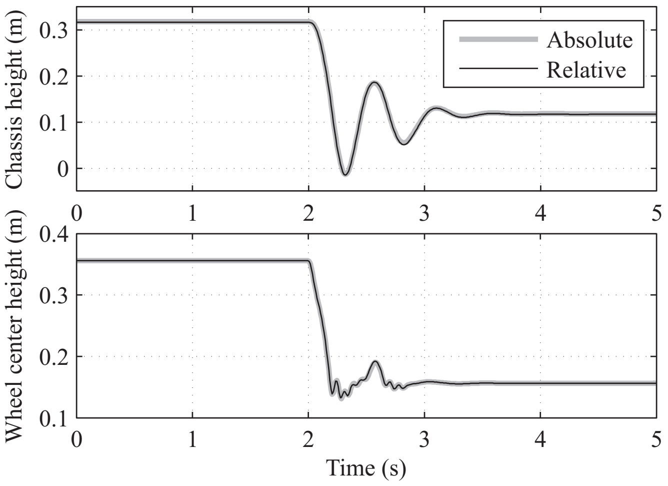


Figure 2.23: Iltis suspension simulation results.

Table 2.4: CPU–times (ms) in the second example.

| Flexible elements | None | A | L | R | A+L | A+R | L+R | All |
|---|---|---|---|---|---|---|---|---|
| Absolute | 45 | 155 | 120 | 91 | 234 | 231 | 175 | 298 |
| Relative | 31 | 125 | 52 | 67 | 169 | 167 | 98 | 216 |

method in relative coordinates is much more involved. The energy loss has been measured for this example as well in order to compare the precision of the two methods. The total energy at the end of the simulation must be equal to the total energy at the beginning, minus the energy dissipated in the damper, and the difference between this theoretical value and the actual energy at the end of the simulation is taken as a precision measurement. The results are very similar although slightly better in the relative formulation, which loses 14.07 J in the rigid case and 15.08 J in the fully flexible one, as opposed to 15.73 J and 15.45 J respectively for the absolute method.

### 2.9.3 Iltis vehicle

The third example chosen is the full Iltis vehicle, which is a good example of a large system. The four suspensions of the Iltis vehicle are identical to the one described in the previous example.

In the model in relative coordinates, the connection of the chassis to the inertial frame is made by using a floating joint, and the closed loops have been cut in all the suspensions at the same joints, thus having 12 branches starting from the chassis. When performing the forward position and velocity analyses for the open–loop system, as well as the backward mass and force accumulation, the computation of the four suspensions can be parallelized, since they are independent.

As done in the single suspension problem, all possible combinations of rigid and

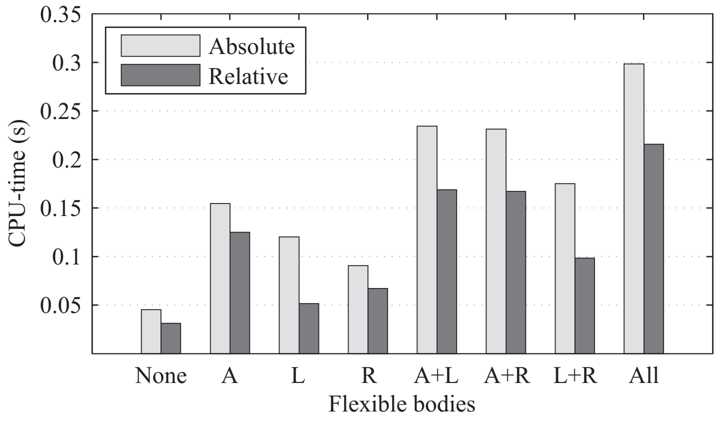


Figure 2.24: CPU–times in the second example.

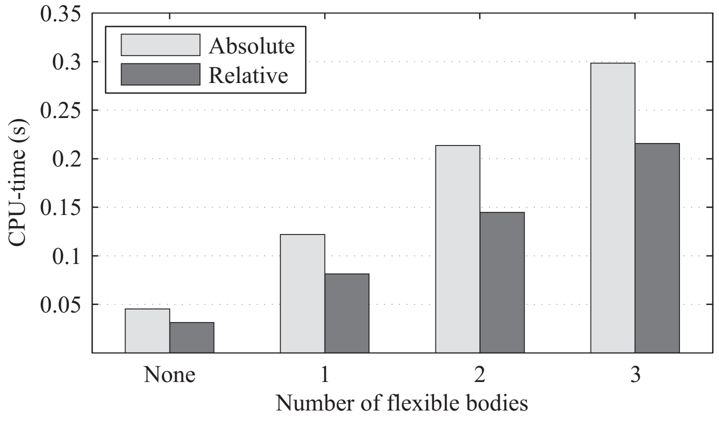


Figure 2.25: CPU–times vs. number of flexible bodies in the second example.

flexible bodies have been tested, using both formulations. The combinations have been chosen keeping the modeling of the four suspensions identical, i.e. if one element (A, L or R) is considered as flexible, it is done in all four suspensions. The number of coordinates obtained for each combination is shown in Table 2.5. As can be observed, the coordinate numbers tends to be three times lower in the relative case, as it happened in the previous example. But the relative formulation, due to the modal amplitudes, no longer has the very low number of coordinates common for such formulations, reaching a total of 98 coordinates in case of the highest number of flexible bodies.

Table 2.5: Number of system coordinates in the third example.

| Flexible elements | None | A | L | R | A+L | A+R | L+R | All |
|---|---|---|---|---|---|---|---|---|
| Absolute | 168 | 196 | 216 | 228 | 244 | 256 | 276 | 304 |
| Relative | 34 | 50 | 58 | 58 | 74 | 74 | 82 | 98 |

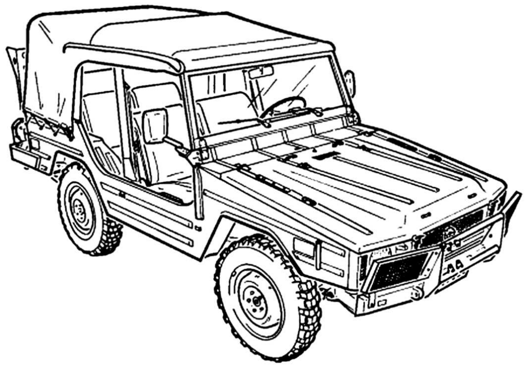


Figure 2.26: Iltis vehicle.

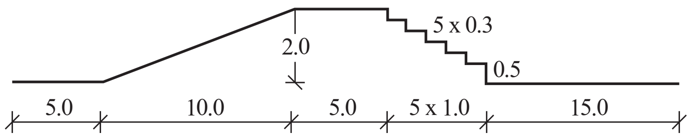


Figure 2.27: Road profile for the Iltis vehicle simulation.

The vehicle runs during 8 s over the road profile shown in Figure 2.27, going from left to right with an initial velocity of 5 m/s. The integration is carried out with a time–step of 10 ms, using the Newmark integrator with $\xi = 0$, i.e. the trapezoidal rule. The time histories of the height of both the chassis origin and the center of the front left wheel, when all the elements are considered as flexible, are plotted in Figs. 2.28 and 2.29, showing a very good agreement between the different methods. It is observed that this is a rather violent maneuver, since the car bounces several times when running down the steps. The good agreement between both formulations is also obtained in the elastic coordinates, as can be seen in Figure 2.30, where the deflections of the tip of the front left A–arm obtained by using both formulations are plotted. The CPU–times obtained can be seen in Table 2.6 and Figure 2.31, including all combinations of flexible and rigid bodies. Figure 2.32 shows the CPU–times vs. the number of flexible bodies, in order to more clearly show its influence. In this figure, the times for four and eight flexible bodies (one and two per suspension) are the mean values of all the corresponding different cases, as was done in the second example.

As it happened in the rigid case, the relative method is faster than the absolute one for large systems, but not to the same extent. As can be seen in Table 2.6, the relative method is roughly 5.5 times faster in the rigid case, but only 3 times in the fully flexible case. This happens because the computation of the flexible mass matrices takes most of the computing time.

The error with respect to a reference solution, calculated with a time–step of $10^{-4}$ s, is measured in order to evaluate the precision of both methods. This error is defined as the mean deviation of the height of the chassis from the reference solution,

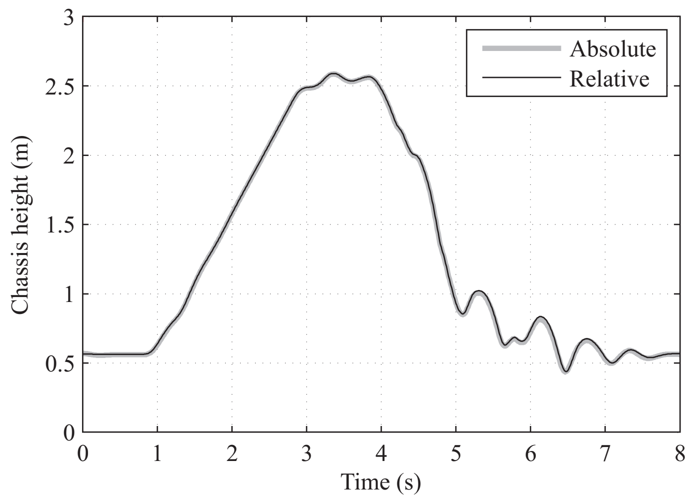


Figure 2.28: Time–history of the origin of the chassis.

Table 2.6: CPU–times (s) in the third example.

| Flexible elements | None | A | L | R | A+L | A+R | L+R | All |
|---|---|---|---|---|---|---|---|---|
| Absolute | 1.25 | 2.33 | 2.35 | 2.41 | 3.35 | 3.44 | 3.31 | 4.48 |
| Relative | 0.23 | 0.74 | 0.57 | 0.58 | 1.22 | 1.21 | 0.93 | 1.48 |

measured along the last three seconds,

$$e = \frac{1}{n_s + 1} \sum_{i=0}^{n_s} \left| z_i - z_i^* \right| \tag{2.184}$$

where $z_i$ and $z_i^*$ are the calculated and reference chassis heights respectively, and $n_s$ is the number of time–steps. Only the last three seconds of the simulation, where the most violent bounces occur, have been taken into account to make the error more significant. It can be observed in Table 2.7 that the relative formulation, following the trend observed in the previous two examples, is also slightly more accurate than the absolute one for the time–step of 0.01 s.

In order to compare the robustness of the two methods, several simulations have been carried out increasing the time–step up to the maximum possible. As it is shown in Table 2.7, the absolute method can run with a maximum time–step of 14 ms in the flexible case, obtaining a CPU–time reduction of about 15%. The relative formulation reaches a time–step of 44 ms, with a significant improvement in the CPU–time, reaching a real–time ratio of more than 13 times in a system with 12 flexible bodies. Obviously, such high time–steps can only be obtained at the expense of introducing significant errors, especially in the case of the relative method where the time–step is three times higher. Going even further, Table 2.7 shows also the fastest results obtained by adding numerical damping to the Newmark integrator, keeping the time–step

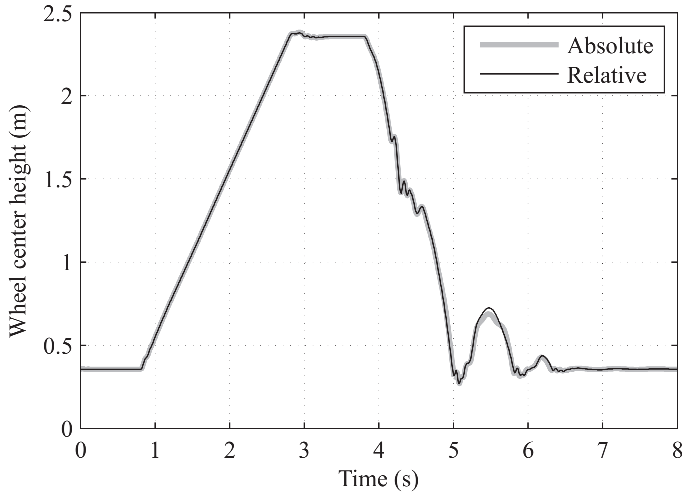


Figure 2.29: Time–history of the height of the front left wheel.

below 50 ms. All these results have a mean error of almost 0.1 m, which may be not admissible for certain applications, but they serve to show how robust the relative formulation is, reaching a time–step of 50 ms in the flexible case, where the absolute method cannot go further than 22 ms.

## 2.10 Conclusions and criteria of use

As expected, the formulation in relative coordinates obtains lower performance than that in natural coordinates for small systems, and higher performance for medium and large systems, being somewhat more accurate in both cases. The higher time–steps reached by the relative method in the last example, especially with the trapezoidal rule, show that it is not only faster for large systems, but also more robust.

It must be pointed out that the first example is implemented in MATLAB, so that the results are not completely reliable. In the rigid case, the absolute method should keep the advantage since the relative method only reduces the coordinates from 6 to 5, but the results obtained when more flexible bodies are added might vary if a compiled language is used.

The introduction of finite element models through the corotational approximation is very easy but can have a high impact on performance. Profiling shows that the $\mathbf{B}^*$ matrix calculation and mass matrix projection takes most of the total integration time. The impact of these operations obviously grows with the number of flexible bodies, reaching, for the Iltis vehicle with the twelve flexible elements, 82% of the total time in the relative formulation and 72% in the absolute one, despite of the coarse finite element meshes used. In order to avoid this problem, the implementation of a different method for evaluating the inertia terms, based on a preprocessing stage for extracting constant mass matrix terms, instead of keeping the size of the underlying

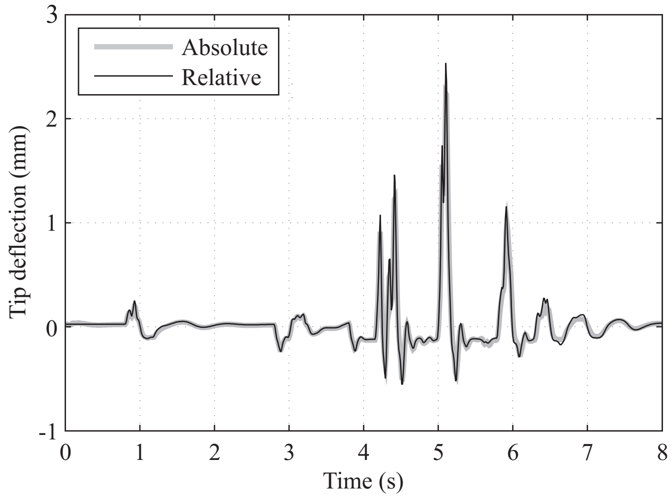


Figure 2.30: Time–history of the tip deflection of the front left A–arm.

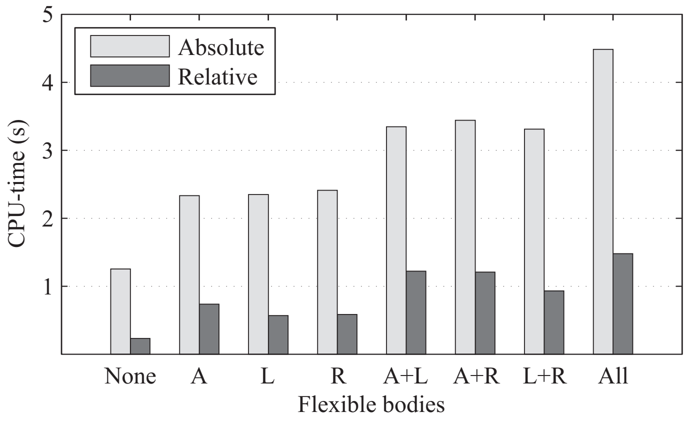


Figure 2.31: CPU–times in the third example.

finite element model –as it happens with matrix $\mathbf{B}^*$– is addressed in the next chapter. These operations ($\mathbf{B}^*$ calculation and mass projection) are performed faster in the relative method, due to the fact that in natural coordinates a flexible body has 12 rigid body variables, as opposed to only 6 in the method in relative coordinates. Each of these coordinates adds a column to the $\mathbf{B}^*$ matrix, which affects very significantly its size in systems with few modes per flexible body.

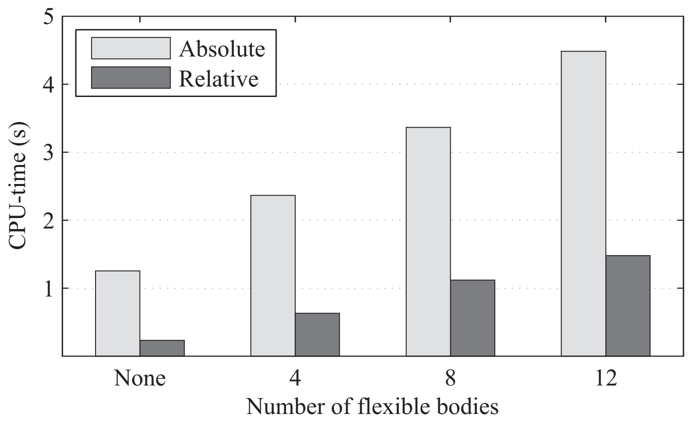


Figure 2.32: CPU–times vs. number of flexible bodies in the third example.

Table 2.7: Efficiency and precision in the third example.

| | Rigid | | Flexible | |
|---|---|---|---|---|
| | Absolute | Relative | Absolute | Relative |
| Trapezoidal rule, time–step 0.01 s | | | | |
| CPU–time (s) | 1.255 | 0.231 | 4.483 | 1.478 |
| Error ($\times 10^{-2}$ m) | 2.472 | 1.934 | 2.224 | 2.175 |
| Real–time ratio | 6.37 | 34.63 | 1.78 | 5.41 |
| Trapezoidal rule, highest time–step below 0.05 s | | | | |
| Time–step (s) | 0.034 | 0.044 | 0.014 | 0.044 |
| CPU–time (s) | 0.610 | 0.094 | 3.859 | 0.609 |
| Error ($\times 10^{-2}$ m) | 7.297 | 7.303 | 3.031 | 8.550 |
| Real–time ratio | 13.11 | 85.11 | 2.07 | 13.14 |
| Newmark, highest time–step below 0.05 s | | | | |
| $\xi$ parameter | -0.8 | -0.8 | -0.8 | -0.8 |
| Time–step (s) | 0.050 | 0.050 | 0.022 | 0.050 |
| CPU–time (s) | 0.359 | 0.063 | 2.062 | 0.375 |
| Error ($\times 10^{-2}$ m) | 9.301 | 9.479 | 9.989 | 9.517 |
| Real–time ratio | 22.28 | 126.98 | 3.88 | 21.33 |

# Chapter 3

# Inertia Shape Integrals

## 3.1 Introduction

The projection of the inertia terms into the body coordinates by means of a full $\mathbf{B}^*$ matrix, the *projection method* in what follows, was first introduced into a FFR formulation in natural coordinates by Avello (1995). This method is very simple to implement, as can be seen in the previous chapter, and if the sparsity of the finite element matrix is taken advantage of, it is reasonably efficient for small to medium size meshes. If large finite element models are required, however, the method can become unusable due to the dependency on the size of the finite element mesh. The two formulations compared in Chapter 2 show a very good performance, but their practical application, due to the use of this method for calculating the inertia terms, can be restricted for this reason.

This chapter is focused on the implementation and efficiency of a different method for the calculation of the inertia terms, the *preprocessing method,* based on the use of the inertia shape integrals or *invariants* (Shabana, 1991; Sugiyama et al., 2006). These integrals were already used by Cuadrado et al. (1996) in a FFR formulation in natural coordinates, similar to that used in the present work for comparison purposes (Cuadrado et al., 2004c; Gutiérrez, 2003). That formulation used a different modeling for the flexible bodies, not considering the static modes as system variables, but writing them in terms of the points coordinates and unit vectors components, a difference that completely modified the resulting implementation.

The inertia shape integrals are a set of invariant matrices which are obtained at a preprocessing stage, by integrating the deformation modes, the undeformed positions, and some certain products between them, over the whole volume of the body. The use of inertia shape integrals for obtaining the inertia terms leads to operations involving only matrices the size of the reduced model. This means that their use eliminates the size of the finite element mesh from the system, taking full advantage of the modal reduction, so that the CPU–time will only depend on the size of the reduced model, i.e. the number of deformation modes selected, and the mesh can be refined as much as needed without introducing any penalty to the simulation time. Wallrapp (1994) included these integrals in a proposal for standardization of the input data, externally generated, required by a multibody code to model flexible bodies.

In this chapter, the implementation of the inertia shape integrals in both the absolute and the relative formulations is discussed, and an efficiency comparison to the original full $\mathbf{B}^*$ matrix projection method is performed, in order to provide some practical criteria of use. All the development is carried out for isoparametric elements, for the sake of simplicity, although everything is easily generalizable to the non–isoparametric case. First, the main idea of the method, common to both formulations, is described. Then, the implementation of the method in both absolute and relative coordinates is detailed. A method for efficiently calculating the inertia shape integrals, based on simple matrix products, is described in the next section, and then detailed for three–dimensional beams. Finally, the results obtained with both the projection and the preprocessing methods are presented and discussed.

## 3.2 General description

The velocity of an arbitrary point of a flexible body can be expressed as a linear function of the generalized velocities $\dot{\mathbf{q}}$, by means of a projection matrix $\mathbf{B}$,

$$\dot{\mathbf{r}} = \mathbf{B}\,(\mathbf{q})\,\dot{\mathbf{q}} \tag{3.1}$$

This is applicable to both the velocities in natural coordinates, $\dot{\mathbf{q}}$, or in the intermediate Cartesian velocities used in the previous chapter, $\mathbf{Z}$, being the $\mathbf{B}$ matrix obviously different for each coordinate set. The mass matrix and the velocity dependent inertia forces vector must be expressed in terms of the $\dot{\mathbf{q}}$ or $\mathbf{Z}$ coordinates in order to include them in the equations of motion. In the methods described so far, this has been done by assembling a full $\mathbf{B}^*$ matrix including all the finite element nodes, and then performing the projection of the inertia terms at every iteration of the integrator. This procedure, as pointed out at the beginning of this chapter, is very simple to implement, but if the finite element model becomes too large, being the number of DOFs several orders of magnitude larger than the number of modes, a different method should be used in order to keep the efficiency requirements. The main idea is to eliminate the size of the finite element model, making the CPU–time dependent only on the number of modes chosen for the model reduction, which is actually the objective of using a modal reduction.

### 3.2.1 Derivation of the inertia terms

Three steps might be distinguished when deriving the mass matrix from the kinetic energy expression: the finite element discretization, the transformation between absolute and generalized velocities, and the mass integration. The key difference between the projection method and the preprocessing method addressed in this chapter is the order in which these three operations are performed. In the previous method, the finite element discretization was first introduced, then the integration of the interpolation matrices led to the constant finite element mass matrix $\mathbf{M}^*$, which could be finally projected into the generalized velocities $\dot{\mathbf{q}}$ by means of the full transformation matrix $\mathbf{B}^*$ at every time step:

$$T = \frac{1}{2}\int_V \dot{\mathbf{r}}^\mathsf{T}\dot{\mathbf{r}}\,dm \approx \frac{1}{2}\dot{\mathbf{r}}^{*\mathsf{T}}\mathbf{M}^*\dot{\mathbf{r}}^* \implies \mathbf{M} \approx \mathbf{B}^{*\mathsf{T}}\mathbf{M}^*\mathbf{B}^* \tag{3.2}$$

In the method addressed in this chapter, the velocity projection for a generic point is applied first, and after taking $\dot{\mathbf{q}}$ out of the integral, the kinetic energy is obtained as,

$$T = \frac{1}{2}\int_V \dot{\mathbf{r}}^\mathsf{T}\dot{\mathbf{r}}\,dm = \frac{1}{2}\dot{\mathbf{q}}^\mathsf{T}\left(\int_V \mathbf{B}^\mathsf{T}\mathbf{B}\,dm\right)\dot{\mathbf{q}} \tag{3.3}$$

which means that the mass matrix expressed in the body coordinates $\mathbf{q}$ is

$$\mathbf{M} = \int_V \mathbf{B}^\mathsf{T}\mathbf{B}\,dm \tag{3.4}$$

and in order to calculate this integral, the finite element discretization is introduced. The results obtained are then the same as in the projection method, since only the order in which operations are performed is changed. In the case of beams, this integral can be done by using the analytical functions of the deformation modes, thus eliminating the inconsistency and obtaining their exact values.

The definition of the kinetic energy in Eq. (3.3) can be introduced into the Lagrangian, then the Lagrange equations can be applied to obtain the expression of the velocity dependent inertia forces vector, as done in the projection method,

$$\mathbf{Q}_v = -\left(\int_V \mathbf{B}^\mathsf{T}\dot{\mathbf{B}}\,dm\right)\dot{\mathbf{q}} \tag{3.5}$$

## 3.2.2 Definition of the inertia shape integrals

As mentioned before, the integrals needed for obtaining the mass matrix and the velocity dependent forces vector can be efficiently calculated by matrix and vector operations if some invariant matrices are extracted. The complete set needed consists of 16 mass integrals, including the undeformed positions, the mode shapes, and several combinations of products between them. These integrals can be divided into two sets. The first set consists of three mass integrals, where only undeformed positions appear, so that they lead to inertia terms associated to the undeformed body motion. The inertia shape integrals are the remaining 13 ones, which include the mode shapes, hence they are used to obtain the variation of the inertia properties produced by deformation.

The first three integrals can be obtained from the undeformed geometry of the body, and include the mass of the body $m$, the static moment $\bar{\mathbf{m}}_u$, and the planar inertia tensor $\bar{\mathbf{P}}_u$, all of them calculated in the local frame,

$$m = \int_V dm; \qquad \bar{\mathbf{m}}_u = \int_V \bar{\mathbf{r}}_u\,dm; \qquad \bar{\mathbf{P}}_u = \int_V \bar{\mathbf{r}}_u\bar{\mathbf{r}}_u^\mathsf{T}\,dm \tag{3.6}$$

The first integral does not need to be calculated unless the mass of the body is unknown, and has been included in the set only for completeness. The static moment $\bar{\mathbf{m}}_u$ can also be directly obtained without integration, if the mass and the undeformed position of the center of gravity $\bar{\mathbf{r}}_u^G$ are known,

$$\bar{\mathbf{m}}_u = \int_V \bar{\mathbf{r}}_u \, dm = m\bar{\mathbf{r}}_u^G \tag{3.7}$$

And the planar inertia tensor can be derived from the undeformed inertia tensor $\bar{\mathbf{J}}_u$ in case it is available,

$$\bar{\mathbf{P}}_u = \int_V \bar{\mathbf{r}}_u \bar{\mathbf{r}}_u^{\mathsf{T}} \, dm = \frac{1}{2} \sum_{i=1}^{3} \left( \bar{J}_u \right)_{ii} \mathbf{I}_3 - \bar{\mathbf{J}}_u = \bar{J}_u \mathbf{I}_3 - \bar{\mathbf{J}}_u \tag{3.8}$$

being $\bar{J}_u$ the moment of inertia of the undeformed body with respect to the origin of the local frame of reference.

All the remaining integrals involve the mode shapes $\mathbf{X}$; therefore, they will be used to obtain the variable part of the inertia terms. Three kinds of these integrals can be defined, generating a total of 13 constant matrices. The first type is the integral of the mode shapes themselves, which results in a $3 \times n_m$ matrix, being $n_m$ the number of columns in $\mathbf{X}$, i.e. the number of deformation modes chosen for the reduction of the finite element model,

$$\mathbf{S} = \int_V \mathbf{X} \, dm \tag{3.9}$$

If the mode shapes $\mathbf{X}$ are multiplied by the three components of the undeformed position and integrated, three more constant $3 \times n_m$ matrices are obtained,

$$\mathbf{S}^i = \int_V \bar{r}_{ui} \mathbf{X} \, dm, \quad i = 1, 2, 3 \tag{3.10}$$

And the remaining nine matrices, of size $n_m \times n_m$, include the integrals of the products between the three directions of the mode shapes,

$$\mathbf{S}^{ij} = \int_V \mathbf{X}_i^{\mathsf{T}} \mathbf{X}_j \, dm, \quad i, j = 1, 2, 3 \tag{3.11}$$

where $\mathbf{X}_i$ is the $i^{\text{th}}$ row of $\mathbf{X}$. It must be noted that only six of these integrals need to be calculated, since $\mathbf{S}^{ji}$ is equal to the transpose of $\mathbf{S}^{ij}$.

The complete set of undeformed geometry integrals and inertia shape integrals here defined, along with the generalized coordinates vector $\mathbf{q}$ and its time derivative $\dot{\mathbf{q}}$, contain all the necessary information required to calculate the mass matrix and the velocity dependent forces vector of a deformable body. In order to make the procedure clearer, the inertia terms can be considered as divided into blocks, according to the structure of the generalized coordinates vector,

$$\mathbf{M} = \begin{bmatrix} \mathbf{M}_{tt} & \mathbf{M}_{t\theta} & \mathbf{M}_{tf} \\ & \mathbf{M}_{\theta\theta} & \mathbf{M}_{\theta f} \\ sym. & & \mathbf{M}_{ff} \end{bmatrix}, \quad \mathbf{Q}_v = \begin{Bmatrix} \mathbf{Q}_{vt} \\ \mathbf{Q}_{v\theta} \\ \mathbf{Q}_{vf} \end{Bmatrix} \tag{3.12}$$

where the subindex $t$ refers to the inertia associated to the translation of the frame of reference, $\theta$ to that of its rotation, and $f$ to the inertia of the elastic coordinates.

## 3.3 Implementation in absolute coordinates

As pointed out in the previous section, the velocity of a point can be expressed as a linear function of the generalized velocities $\dot{\mathbf{q}}$, by means of a variable transformation matrix $\mathbf{B}$. On the formulation in natural coordinates, the generalized velocities are

$$\dot{\mathbf{q}}^{\mathsf{T}} = \begin{Bmatrix} \dot{\mathbf{r}}_0^{\mathsf{T}} & \dot{\mathbf{u}}^{\mathsf{T}} & \dot{\mathbf{v}}^{\mathsf{T}} & \dot{\mathbf{w}}^{\mathsf{T}} & \dot{\mathbf{y}}^{\mathsf{T}} \end{Bmatrix} \tag{3.13}$$

where $\mathbf{r}_0$ is the position of the origin of the local frame, $\mathbf{u}$, $\mathbf{v}$ and $\mathbf{w}$ are its three unit vectors expressed in global coordinates, which can be grouped into a rotation matrix $\mathbf{A}$, and $\dot{\mathbf{y}}$ is the vector of elastic coordinates. In order to obtain an expression for the $\mathbf{B}$ matrix, the velocity of a point can be obtained by differentiating the position given by Eqs. (2.1) and (2.3), thus leading to

$$\dot{\mathbf{r}} = \dot{\mathbf{r}}_0 + \dot{\mathbf{A}}\bar{\mathbf{r}} + \mathbf{A}\dot{\bar{\mathbf{r}}} = \dot{\mathbf{r}}_0 + \dot{\mathbf{A}}\bar{\mathbf{r}} + \mathbf{A}\mathbf{X}\dot{\mathbf{y}} \tag{3.14}$$

which is a linear relationship between the velocity of the point and the generalized velocities $\dot{\mathbf{q}}$. This can be expressed in matrix form, as in Eq. (3.1), thus obtaining an expression for the $\mathbf{B}$ matrix of a generic point in natural coordinates,

$$\mathbf{B} = \begin{bmatrix} \mathbf{I}_3 & \bar{r}_1\mathbf{I}_3 & \bar{r}_2\mathbf{I}_3 & \bar{r}_3\mathbf{I}_3 & \mathbf{A}\mathbf{X} \end{bmatrix} \tag{3.15}$$

The components of the local deformed position $\bar{\mathbf{r}}$, provided that $\bar{\mathbf{r}} = \bar{\mathbf{r}}_u + \mathbf{X}\mathbf{y}$, are,

$$\bar{r}_i = \bar{r}_{ui} + \mathbf{X}_i\mathbf{y}, \quad i = 1, 2, 3 \tag{3.16}$$

### 3.3.1 Mass matrix

In order to obtain the mass matrix, the $\mathbf{B}^{\mathsf{T}}\mathbf{B}$ product must be developed

$$\mathbf{M} = \int_V \mathbf{B}^{\mathsf{T}}\mathbf{B}\,dm = \int_V \left[\begin{array}{c|ccc|c} \mathbf{I}_3 & \bar{r}_1\mathbf{I}_3 & \bar{r}_2\mathbf{I}_3 & \bar{r}_3\mathbf{I}_3 & \mathbf{A}\mathbf{X} \\ \hline & \bar{r}_1^2\mathbf{I}_3 & \bar{r}_1\bar{r}_2\mathbf{I}_3 & \bar{r}_1\bar{r}_3\mathbf{I}_3 & \bar{r}_1\mathbf{A}\mathbf{X} \\ & & \bar{r}_2^2\mathbf{I}_3 & \bar{r}_2\bar{r}_3\mathbf{I}_3 & \bar{r}_2\mathbf{A}\mathbf{X} \\ & sym. & & \bar{r}_3^2\mathbf{I}_3 & \bar{r}_3\mathbf{A}\mathbf{X} \\ \hline & & & & \mathbf{X}^{\mathsf{T}}\mathbf{X} \end{array}\right] dm \tag{3.17}$$

In what follows, the derivation of the different blocks in terms of the inertia shape integrals will be addressed in detail.

**Mass terms associated to the reference coordinates**

The first three blocks $\mathbf{M}_{tt}$, $\mathbf{M}_{t\theta}$ and $\mathbf{M}_{\theta\theta}$ contain the inertia terms related to the motion of the frame of reference. They have the same physical meaning as in rigid body dynamics, although most of their terms are now variable.

The first block $\mathbf{M}_{tt}$ is a constant $3 \times 3$ diagonal matrix, representing the translational inertia of the body,

$$\mathbf{M}_{tt} = \int_V \mathbf{I}_3 \, dm = m\mathbf{I}_3 \tag{3.18}$$

The second block $\mathbf{M}_{t\theta}$ contains the mass terms that couple the translational and rotational inertia of the reference frame,

$$\mathbf{M}_{t\theta} = \int_V \begin{bmatrix} \bar{r}_1\mathbf{I}_3 & \bar{r}_2\mathbf{I}_3 & \bar{r}_3\mathbf{I}_3 \end{bmatrix} dm \tag{3.19}$$

and its calculation requires the integration of the three components of the deformed local position $\bar{\mathbf{r}}$. The integral of $\bar{\mathbf{r}}$ is by definition the static moment $\bar{\mathbf{m}}$, in the deformed configuration, and it can be considered as divided into a constant and a variable part,

$$\bar{\mathbf{m}} = \int_V \bar{\mathbf{r}} \, dm = \int_V (\bar{\mathbf{r}}_u + \mathbf{X}\mathbf{y}) \, dm \tag{3.20}$$

The integral of the undeformed position $\bar{\mathbf{r}}_u$ is already known, since it is the undeformed static moment $\bar{\mathbf{m}}_u$. The second term represents the variation introduced by the deformation, and it can be easily calculated by taking the modal amplitudes vector $\mathbf{y}$ out of the integral, so that the remaining integral is nothing but the integral of the mode shapes $\mathbf{S}$. The static moment is finally obtained as,

$$\bar{\mathbf{m}} = \bar{\mathbf{m}}_u + \mathbf{S}\mathbf{y} \tag{3.21}$$

and its three components are the respective diagonals of the three $3 \times 3$ blocks that form the $\mathbf{M}_{t\theta}$ submatrix,

$$\mathbf{M}_{t\theta} = \begin{bmatrix} \bar{m}_1\mathbf{I}_3 & \bar{m}_2\mathbf{I}_3 & \bar{m}_3\mathbf{I}_3 \end{bmatrix} \tag{3.22}$$

In the center of the mass matrix the block $\mathbf{M}_{\theta\theta}$ is found. It contains the rotational inertia of the frame of reference, and it is itself formed by nine diagonal $3 \times 3$ blocks,

$$\mathbf{M}_{\theta\theta} = \int_V \begin{bmatrix} \bar{r}_1^2\mathbf{I}_3 & \bar{r}_1\bar{r}_2\mathbf{I}_3 & \bar{r}_1\bar{r}_3\mathbf{I}_3 \\ & \bar{r}_2^2\mathbf{I}_3 & \bar{r}_2\bar{r}_3\mathbf{I}_3 \\ sym. & & \bar{r}_3^2\mathbf{I}_3 \end{bmatrix} dm \tag{3.23}$$

In this case, the terms to be integrated are the components of the deformed planar inertia tensor $\bar{\mathbf{P}}$,

$$\bar{P}_{ij} = \int_V \bar{r}_i \bar{r}_j \, dm, \quad i,j = 1,2,3 \tag{3.24}$$

Each $\bar{r}_i \bar{r}_j$ product can be calculated by first decomposing $\bar{r}_i$ and $\bar{r}_j$ into their constant and variable parts,

$$\int_V \bar{r}_i \bar{r}_j \, dm = \int_V (\bar{r}_{ui} + \mathbf{X}_i \mathbf{y}) \left(\bar{r}_{uj} + \mathbf{X}_j \mathbf{y}\right) dm, \quad i,j = 1,2,3 \tag{3.25}$$

and then developing the product,

$$\int_V \left(\bar{r}_{ui}\bar{r}_{uj} + \bar{r}_{ui}\mathbf{X}_j\mathbf{y} + \bar{r}_{uj}\mathbf{X}_i\mathbf{y} + \mathbf{X}_i\mathbf{y}\mathbf{X}_j\mathbf{y}\right) dm, \quad i,j = 1,2,3 \tag{3.26}$$

Each $\bar{P}_{ij}$ needs the evaluation of four integrals. The first one is the integral of $\bar{r}_{ui}\bar{r}_{uj}$, which is constant and is recognised as the element $ij$ of the undeformed planar inertia tensor. The remaining three terms depend all on $\mathbf{y}$, so that they represent the variation produced by deformation. Two of them are linear in $\mathbf{y}$, and they include $\mathbf{S}^i_j$ and $\mathbf{S}^j_i$, which are the $j^{\text{th}}$ row of $\mathbf{S}^i$ and the $i^{\text{th}}$ row of $\mathbf{S}^j$ respectively. The last term is equal to the integral of $\mathbf{y}^{\mathsf{T}}\mathbf{X}_i^{\mathsf{T}}\mathbf{X}_j\mathbf{y}$, and it leads to a quadratic expression in $\mathbf{y}$. Finally, each element $\bar{P}_{ij}$ of the planar inertia tensor can be found to be

$$\bar{P}_{ij} = \left(\bar{P}_u\right)_{ij} + \left(\mathbf{S}^i_j + \mathbf{S}^j_i\right)\mathbf{y} + \mathbf{y}^{\mathsf{T}}\mathbf{S}^{ij}\mathbf{y}, \quad i,j = 1,2,3 \tag{3.27}$$

and used to assemble the rotational inertia submatrix,

$$\mathbf{M}_{\theta\theta} = \begin{bmatrix} \bar{P}_{11}\mathbf{I}_3 & \bar{P}_{12}\mathbf{I}_3 & \bar{P}_{13}\mathbf{I}_3 \\ & \bar{P}_{22}\mathbf{I}_3 & \bar{P}_{23}\mathbf{I}_3 \\ sym. & & \bar{P}_{33}\mathbf{I}_3 \end{bmatrix} \tag{3.28}$$

It is observed that, if only the constant terms $\bar{\mathbf{m}}_u$ and $\bar{\mathbf{P}}_u$ are used for obtaining these blocks, the resulting matrix is the mass matrix of the body, considered as rigid in its undeformed state.

### Mass terms coupling the reference coordinates to the elastic coordinates

The first four blocks of the last column, all of them of size $3 \times n_m$, represent the inertia coupling between the reference and the elastic coordinates.

The first one, which will only depend on the orientation but not on the deformation state, will couple the translational inertia to the elastic deformation, and is easily obtained by taking $\mathbf{A}$ out of the integral,

$$\mathbf{M}_{rf} = \int_V \mathbf{A}\mathbf{X} \, dm = \mathbf{A}\mathbf{S} \tag{3.29}$$

The second, third and fourth blocks couple the rotational and deformation inertias:

$$\mathbf{M}_{\theta f} = \int_V \begin{bmatrix} \bar{r}_1 \mathbf{AX} \\ \bar{r}_2 \mathbf{AX} \\ \bar{r}_3 \mathbf{AX} \end{bmatrix} dm \tag{3.30}$$

These are the only terms which depend both on the orientation and on the deformation state, and are the most involved of the mass matrix. If the rotation matrix $\mathbf{A}$ is taken out of the integral in each block, the remaining integrals are those of $\bar{r}_i \mathbf{X}$. It can be observed that these integrals are analogous to the inertia shape integrals $\mathbf{S}^i$, but, in this case, the factors that multiply the mode shapes are the components of the deformed local position, which may be named $\mathbf{S}_d^i$,

$$\mathbf{S}_d^i = \int_V \bar{r}_i \mathbf{X}\, dm, \quad i = 1, 2, 3 \tag{3.31}$$

In order to evaluate them, the local deformed position is decomposed as usual:

$$\int_V \bar{r}_i \mathbf{X}\, dm = \int_V (\bar{r}_{ui} + \mathbf{X}_i \mathbf{y})\, \mathbf{X}\, dm, \quad i = 1, 2, 3 \tag{3.32}$$

The first term of the integral is the inertia shape integral $\mathbf{S}^i$, and in order to obtain the second, some manipulations are needed. First, the scalar product $\mathbf{X}_i \mathbf{y}$ can be transposed, but $\mathbf{y}^{\mathsf{T}}$ can not yet be taken out of the integral since the remaining product $\mathbf{X}_i^{\mathsf{T}} \mathbf{X}$ is not compatible. To avoid this problem, $\mathbf{X}$ can be divided into its three rows, yielding

$$\int_V \mathbf{X}_i \mathbf{y} \mathbf{X}\, dm = \int_V \mathbf{y}^{\mathsf{T}} \mathbf{X}_i^{\mathsf{T}} \mathbf{X}\, dm = \int_V \begin{bmatrix} \mathbf{y}^{\mathsf{T}} \mathbf{X}_i^{\mathsf{T}} \mathbf{X}_1 \\ \mathbf{y}^{\mathsf{T}} \mathbf{X}_i^{\mathsf{T}} \mathbf{X}_2 \\ \mathbf{y}^{\mathsf{T}} \mathbf{X}_i^{\mathsf{T}} \mathbf{X}_3 \end{bmatrix} dm, \quad i = 1, 2, 3 \tag{3.33}$$

where the three inertia shape integrals $\mathbf{S}^{i1}$, $\mathbf{S}^{i2}$ and $\mathbf{S}^{i3}$ are recognized, leading to

$$\mathbf{S}_d^i = \mathbf{S}^i + \begin{bmatrix} \mathbf{y}^{\mathsf{T}} \mathbf{S}^{i1} \\ \mathbf{y}^{\mathsf{T}} \mathbf{S}^{i2} \\ \mathbf{y}^{\mathsf{T}} \mathbf{S}^{i3} \end{bmatrix}, \quad i = 1, 2, 3 \tag{3.34}$$

The $\mathbf{M}_{\theta f}$ block of the mass matrix is finally obtained by rotating the $\mathbf{S}_d^i$ integrals:

$$\mathbf{M}_{\theta f} = \begin{bmatrix} \mathbf{AS}_d^1 \\ \mathbf{AS}_d^2 \\ \mathbf{AS}_d^3 \end{bmatrix} \tag{3.35}$$

**Mass terms associated to the elastic coordinates**

The last $n_m \times n_m$ block of the mass matrix is constant, as it happened to the mass inertia associated to the origin of the reference frame $\mathbf{r}_0$. The value of this block, according to

Eq. (3.4), is equal to $\mathbf{X}^{\mathsf{T}}\mathbf{A}^{\mathsf{T}}\mathbf{A}\mathbf{X}$. Since the vectors of the frame of reference **u**, **v** and **w** have been defined as unit orthogonal vectors, the rotation matrix **A** is orthogonal, and this makes the product $\mathbf{A}^{\mathsf{T}}\mathbf{A}$ identically equal to $\mathbf{I}_3$ independently on the orientation of the frame, thus making this block constant. This block, once eliminated the orientation dependency, is nothing but the well–known modal mass matrix, widely used in the structural dynamics field and, as will be seen later, is the same for the method in relative coordinates. It can be obtained as,

$$\int_V \mathbf{X}^{\mathsf{T}}\mathbf{X}\,dm = \mathbf{S}^{11} + \mathbf{S}^{22} + \mathbf{S}^{33} \tag{3.36}$$

## 3.3.2 Velocity dependent inertia forces

The velocity dependent inertia forces are obtained by means of Eq. (3.5), so that the first step to be performed is the **B** matrix differentiation, a straightforward operation if the structure of the local position components, shown in Eq. (3.16), is taken into account,

$$\dot{\mathbf{B}} = \begin{bmatrix} 0 & \mathbf{X}_1\dot{\mathbf{y}}\mathbf{I}_3 & \mathbf{X}_2\dot{\mathbf{y}}\mathbf{I}_3 & \mathbf{X}_3\dot{\mathbf{y}}\mathbf{I}_3 & \dot{\mathbf{A}}\mathbf{X} \end{bmatrix} \tag{3.37}$$

If the $\dot{\mathbf{B}}\dot{\mathbf{q}}$ product is evaluated, not forgetting that $\dot{\mathbf{A}}$ is formed by the derivatives of the unit vectors $\dot{\mathbf{u}}$, $\dot{\mathbf{v}}$ and $\dot{\mathbf{w}}$ as columns, it can be easily found that

$$\dot{\mathbf{B}}\dot{\mathbf{q}} = \mathbf{X}_1\dot{\mathbf{y}}\dot{\mathbf{u}} + \mathbf{X}_2\dot{\mathbf{y}}\dot{\mathbf{v}} + \mathbf{X}_3\dot{\mathbf{y}}\dot{\mathbf{w}} + \dot{\mathbf{A}}\mathbf{X}\dot{\mathbf{y}} = 2\dot{\mathbf{A}}\mathbf{X}\dot{\mathbf{y}} \tag{3.38}$$

It the derivative of the orientation matrix is expressed in terms of the angular velocity $\boldsymbol{\omega}$, it can be observed that this expression coincides with the Coriolis acceleration,

$$\dot{\mathbf{B}}\dot{\mathbf{q}} = 2\boldsymbol{\omega} \times \mathbf{A}\dot{\bar{\mathbf{r}}} \tag{3.39}$$

The different blocks of the velocity dependent forces vector, defined according to the partition shown in Eq. (3.12), can be obtained after evaluating the product $\mathbf{B}^{\mathsf{T}}\dot{\mathbf{B}}\dot{\mathbf{q}}$ and performing the integration,

$$\mathbf{Q}_{vt} = -2\int_V \dot{\mathbf{A}}\mathbf{X}\dot{\mathbf{y}}\,dm \tag{3.40}$$

$$\mathbf{Q}_{v\theta} = -2\int_V \begin{bmatrix} \dot{\mathbf{A}}\bar{r}_1\mathbf{X} \\ \dot{\mathbf{A}}\bar{r}_2\mathbf{X} \\ \dot{\mathbf{A}}\bar{r}_3\mathbf{X} \end{bmatrix} \dot{\mathbf{y}}\,dm \tag{3.41}$$

$$\mathbf{Q}_{vf} = -2\int_V \mathbf{X}^{\mathsf{T}}\mathbf{A}^{\mathsf{T}}\dot{\mathbf{A}}\mathbf{X}\dot{\mathbf{y}}\,dm \tag{3.42}$$

The forces associated to the reference coordinates are straightforward if the inertia shape integrals are already known. The first block includes the integral of the mode shapes, so that it is immediately obtained by taking the rotation matrix out of the integral

$$\mathbf{Q}_{vt} = -2\dot{\mathbf{A}}\mathbf{S}\dot{\mathbf{y}} \tag{3.43}$$

The rotational forces require the integrals of the deformed local position components times the mode shapes, which have been already calculated

$$\mathbf{Q}_{v\theta} = -2\begin{bmatrix} \dot{\mathbf{A}}\mathbf{S}_d^1 \\ \dot{\mathbf{A}}\mathbf{S}_d^2 \\ \dot{\mathbf{A}}\mathbf{S}_d^3 \end{bmatrix}\dot{\mathbf{y}} \tag{3.44}$$

The only problem left is calculating the integral of $\mathbf{X}^{\mathsf{T}}\mathbf{A}^{\mathsf{T}}\dot{\mathbf{A}}\mathbf{X}$, needed to obtain the $\mathbf{Q}_{vf}$ forces. In order to do this, firstly the $\mathbf{A}^{\mathsf{T}}\dot{\mathbf{A}}$ product is studied,

$$\mathbf{A}^{\mathsf{T}}\dot{\mathbf{A}} = \begin{bmatrix} \mathbf{u}^{\mathsf{T}} \\ \mathbf{v}^{\mathsf{T}} \\ \mathbf{w}^{\mathsf{T}} \end{bmatrix}\begin{bmatrix} \dot{\mathbf{u}} & \dot{\mathbf{v}} & \dot{\mathbf{w}} \end{bmatrix} = \begin{bmatrix} \mathbf{u}^{\mathsf{T}}\dot{\mathbf{u}} & \mathbf{u}^{\mathsf{T}}\dot{\mathbf{v}} & \mathbf{u}^{\mathsf{T}}\dot{\mathbf{w}} \\ \mathbf{v}^{\mathsf{T}}\dot{\mathbf{u}} & \mathbf{v}^{\mathsf{T}}\dot{\mathbf{v}} & \mathbf{v}^{\mathsf{T}}\dot{\mathbf{w}} \\ \mathbf{w}^{\mathsf{T}}\dot{\mathbf{u}} & \mathbf{w}^{\mathsf{T}}\dot{\mathbf{v}} & \mathbf{w}^{\mathsf{T}}\dot{\mathbf{w}} \end{bmatrix} \tag{3.45}$$

This resulting matrix is skew–symmetric, since $\dot{\mathbf{A}} = \tilde{\boldsymbol{\omega}}\mathbf{A}$ and $\tilde{\boldsymbol{\omega}}$ is skew–symmetric by definition. If the whole $\mathbf{X}^{\mathsf{T}}\mathbf{A}^{\mathsf{T}}\dot{\mathbf{A}}\mathbf{X}$ product is rewritten, taking into account the skew–symmetry of $\mathbf{A}^{\mathsf{T}}\dot{\mathbf{A}}$ and dividing $\mathbf{X}$ into its three rows,

$$\mathbf{X}^{\mathsf{T}}\mathbf{A}^{\mathsf{T}}\dot{\mathbf{A}}\mathbf{X} = \begin{bmatrix} \mathbf{X}_1^{\mathsf{T}} & \mathbf{X}_2^{\mathsf{T}} & \mathbf{X}_3^{\mathsf{T}} \end{bmatrix}\begin{bmatrix} 0 & \mathbf{u}^{\mathsf{T}}\dot{\mathbf{v}} & \mathbf{u}^{\mathsf{T}}\dot{\mathbf{w}} \\ -\mathbf{u}^{\mathsf{T}}\dot{\mathbf{v}} & 0 & \mathbf{v}^{\mathsf{T}}\dot{\mathbf{w}} \\ -\mathbf{u}^{\mathsf{T}}\dot{\mathbf{w}} & -\mathbf{v}^{\mathsf{T}}\dot{\mathbf{w}} & 0 \end{bmatrix}\begin{bmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \\ \mathbf{X}_3 \end{bmatrix} \tag{3.46}$$

after developing this expression and integrating, the following result for $\mathbf{Q}_{vf}$ is obtained:

$$\mathbf{Q}_{vf} = -2\left[\mathbf{u}^{\mathsf{T}}\dot{\mathbf{v}}\left(\mathbf{S}^{12} - \mathbf{S}^{21}\right) + \mathbf{u}^{\mathsf{T}}\dot{\mathbf{w}}\left(\mathbf{S}^{13} - \mathbf{S}^{31}\right) + \mathbf{v}^{\mathsf{T}}\dot{\mathbf{w}}\left(\mathbf{S}^{23} - \mathbf{S}^{32}\right)\right]\dot{\mathbf{y}} \tag{3.47}$$

The terms within parentheses are constant, so that they can be calculated in the preprocessing stage and stored before starting the simulation.

## 3.4 Implementation in relative coordinates

In the relative method, the velocity of a point of the solid can be expressed, using the intermediate Cartesian coordinates $\mathbf{Z}$, as,

$$\dot{\mathbf{r}} = \dot{\mathbf{s}} + \boldsymbol{\omega} \times \mathbf{r} + \mathbf{A}\dot{\bar{\mathbf{r}}} = \dot{\mathbf{s}} - \tilde{\mathbf{r}}\boldsymbol{\omega} + \mathbf{A}\mathbf{X}\dot{\mathbf{y}} \tag{3.48}$$

which leads to the following expression of the $\mathbf{B}$ matrix,

$$\mathbf{B} = \begin{bmatrix} \mathbf{I}_3 & -\tilde{\mathbf{r}} & \mathbf{A}\mathbf{X} \end{bmatrix} \tag{3.49}$$

### 3.4.1 Mass matrix

The procedure to obtain the mass matrix of a flexible body, expressed in the intermediate coordinates $\mathbf{Z}$, is essentially the same as that used in the absolute method. The first step is to develop the integral of $\mathbf{B}^{\mathsf{T}}\mathbf{B}$,

$$\bar{\mathbf{M}} = \int_V \mathbf{B}^{\mathsf{T}}\mathbf{B}\,dm = \int_V \begin{bmatrix} \mathbf{I}_3 & -\tilde{\mathbf{r}} & \mathbf{AX} \\ & -\tilde{\mathbf{r}}\tilde{\mathbf{r}} & \tilde{\mathbf{r}}\mathbf{AX} \\ sym. & & \mathbf{X}^{\mathsf{T}}\mathbf{X} \end{bmatrix} dm \tag{3.50}$$

There is a key difference between this expression of the mass matrix and the expression obtained for the absolute method: the integrals appearing here involve the absolute position, as opposed to Eq. (3.17), where only the local deformed position components appear in the integrals. In the relative method, the mass matrix contains fewer terms than in the absolute one, but these terms are somewhat more complicated to calculate.

#### Mass terms associated to the reference coordinates

The $3 \times 3$ block corresponding to $\dot{\mathbf{s}}$, as it happened in the absolute method, contains the translational inertia of the body, being a diagonal matrix containing the total mass of the body:

$$\bar{\mathbf{M}}_{tt} = \int_V \mathbf{I}_3\,dm = m\mathbf{I}_3 \tag{3.51}$$

The terms coupling $\dot{\mathbf{s}}$ and $\boldsymbol{\omega}$ contain the integral of $-\tilde{\mathbf{r}}$, which is nothing but the skew–symmetric matrix associated to the static moment of the deformed body with respect to the global origin of coordinates $\mathbf{m}$:

$$\bar{\mathbf{M}}_{t\theta} = \int_V -\tilde{\mathbf{r}}\,dm = -\tilde{\mathbf{m}} \tag{3.52}$$

This integral can be easily derived from the static moment of the deformed body in local coordinates $\bar{\mathbf{m}}$, which can be calculated by using Eq.(3.21), and then expressed in global coordinates by rotation and translation,

$$\mathbf{m} = m\mathbf{r}_0 + \mathbf{A}\bar{\mathbf{m}} \tag{3.53}$$

The integral of the terms related to rotation is the inertia tensor, expressed in global coordinates, and calculated with respect to the global origin,

$$\bar{\mathbf{M}}_{\theta\theta} = \int_V -\tilde{\mathbf{r}}\tilde{\mathbf{r}}\,dm = \mathbf{J} \tag{3.54}$$

If the $\tilde{\mathbf{r}}\tilde{\mathbf{r}}$ product is developed, the expressions obtained are very involved, being much easier to calculate the inertia tensor in local coordinates with respect to the local origin, then transforming it by means of the Steiner theorem and rotation. First the inertia tensor is obtained from the planar inertia tensor described in the absolute method, by using Eq. (3.8), which is also applicable to the deformed configuration. Then it is translated to the center of mass by means of the Huygens' formula,

$$\bar{\mathbf{J}}^G = \bar{\mathbf{J}}^0 + m\tilde{\bar{\mathbf{r}}}^G\tilde{\bar{\mathbf{r}}}^G \tag{3.55}$$

The next step is rotating it in order to express it in global coordinates,

$$\mathbf{J}^G = \mathbf{A}\bar{\mathbf{J}}^G\mathbf{A}^\mathsf{T} \tag{3.56}$$

and finally translating it again from the center of mass to the global origin,

$$\mathbf{J} = \mathbf{J}^G - m\tilde{\mathbf{r}}^G\tilde{\mathbf{r}}^G \tag{3.57}$$

The rotational inertia of the deformed body, with respect to the global origin, results

$$\bar{\mathbf{M}}_{\theta\theta} = \mathbf{A}\left(\bar{\mathbf{J}}^0 + m\tilde{\bar{\mathbf{r}}}^G\tilde{\bar{\mathbf{r}}}^G\right)\mathbf{A}^\mathsf{T} - m\tilde{\mathbf{r}}^G\tilde{\mathbf{r}}^G \tag{3.58}$$

In short, in order to obtain the mass matrix terms corresponding to $\boldsymbol{\omega}$, the planar inertia tensor in local coordinates, for the deformed configuration and with respect to the body local frame origin, must be calculated as described for the absolute method, then transformed into the inertia tensor $\bar{\mathbf{J}}^0$, and finally converted into the global inertia tensor by applying Eq. (3.58).

By using the undeformed static moment $\bar{\mathbf{m}}_u$ and inertia tensor $\bar{\mathbf{J}}_u^0$ instead of their deformed counterparts, the mass matrix of the body in the undeformed configuration, is the result obtained for the reference blocks. Therefore, the mass matrix of any rigid body can be calculated by using these expressions.

## Mass terms coupling the reference coordinates to the elastic coordinates

The block containing the inertia coupling the translation to the elastic coordinates is exactly the same as in the absolute method, and can be obtained directly by rotating the inertia shape integral $\mathbf{S}$, as shown in Eq. (3.29).

The coupling between the rotation and the elastic deformation is the most complicated term. If the rotation matrix $\mathbf{A}$ is divided into its three rows $\mathbf{A}_1$, $\mathbf{A}_2$ and $\mathbf{A}_3$, and the product is developed, it can be found, after some manipulation, that,

$$\bar{\mathbf{M}}_{\theta f} = \int_V \tilde{\bar{\mathbf{r}}}\mathbf{A}\mathbf{X}\,dm = \int_V \begin{bmatrix} \mathbf{A}_3 r_2\mathbf{X} - \mathbf{A}_2 r_3\mathbf{X} \\ \mathbf{A}_1 r_3\mathbf{X} - \mathbf{A}_3 r_1\mathbf{X} \\ \mathbf{A}_2 r_1\mathbf{X} - \mathbf{A}_1 r_2\mathbf{X} \end{bmatrix} dm \tag{3.59}$$

This means that the integrals of $r_1\mathbf{X}$, $r_2\mathbf{X}$ and $r_3\mathbf{X}$ are needed. Analogously to what has been done to the integrals of the components of the deformed local position times the mode shapes, these integrals can be named $\mathbf{S}_a^1$, $\mathbf{S}_a^2$ and $\mathbf{S}_a^3$ respectively, since they are the integrals of the components of the absolute position times the mode shapes.

$$\mathbf{S}_a^i = \int_V r_i \mathbf{X}\, dm = \int_V (r_{0i} + \mathbf{A}_i \bar{\mathbf{r}})\, \mathbf{X}\, dm, \quad i = 1,2,3 \tag{3.60}$$

After developing this expression, the following expression for the three integrals can be obtained,

$$\int_V r_i \mathbf{X}\, dm = r_{0i}\mathbf{S} + \sum_{j=1}^{3} \mathbf{A}_{ij}\mathbf{S}_d^j\, dm, \quad i = 1,2,3 \tag{3.61}$$

which includes the variable integrals $\mathbf{S}_d^j$, the same as those already needed for obtaining the planar inertia tensor by means of Eq. (3.34). The integral of $\tilde{\mathbf{r}}\mathbf{A}\mathbf{X}$ is, in short, obtained by following two steps: first, the $\mathbf{S}_a^i$ integrals are calculated by means of Eq. (3.61) using the already stored $\mathbf{S}_d^i$ integrals obtained from Eq. (3.34); then, the results can be substituted into Eq. (3.59) to obtain the mass terms coupling the rotation to the elastic deformation:

$$\bar{\mathbf{M}}_{\theta f} = \begin{bmatrix} \mathbf{A}_3\mathbf{S}_a^2 - \mathbf{A}_2\mathbf{S}_a^3 \\ \mathbf{A}_1\mathbf{S}_a^3 - \mathbf{A}_3\mathbf{S}_a^1 \\ \mathbf{A}_2\mathbf{S}_a^1 - \mathbf{A}_1\mathbf{S}_a^2 \end{bmatrix} \tag{3.62}$$

**Mass terms associated to the elastic coordinates**

As happened to the translational inertia and the coupling between translation and elastic deformation, the mass associated to the elastic coordinates is the same as in the absolute formulation, being again the constant modal mass matrix already shown in Eq. (3.36).

## 3.4.2 Velocity dependent inertia forces

The velocity dependent inertia forces vector in the intermediate Cartesian coordinate system $\bar{\mathbf{Q}}_v$ is obtained, as previously described, from

$$\bar{\mathbf{Q}}_v = -\int_V \mathbf{B}^{\mathrm{T}}\dot{\mathbf{B}}\mathbf{Z}\, dm \tag{3.63}$$

where, as seen in Chapter 2, the product $\dot{\mathbf{B}}\mathbf{Z}$ is equal to

$$\dot{\mathbf{B}}\mathbf{Z} = \tilde{\boldsymbol{\omega}}\tilde{\boldsymbol{\omega}}\mathbf{r} + 2\dot{\mathbf{A}}\mathbf{X}\dot{\mathbf{y}} \tag{3.64}$$

After several manipulations, the following expressions for the three sections of $\bar{\mathbf{Q}}_v$ can be obtained:

$$\bar{\mathbf{Q}}_{vt} = \int_V \left(\tilde{\boldsymbol{\omega}}\tilde{\mathbf{r}}\boldsymbol{\omega} - 2\dot{\mathbf{A}}\mathbf{X}\dot{\mathbf{y}}\right) dm \tag{3.65}$$

$$\bar{\mathbf{Q}}_{v\theta} = \int_V \left(\tilde{\boldsymbol{\omega}}\tilde{\mathbf{r}}\tilde{\mathbf{r}}\boldsymbol{\omega} - 2\tilde{\mathbf{r}}\dot{\mathbf{A}}\mathbf{X}\dot{\mathbf{y}}\right) dm \tag{3.66}$$

$$\bar{\mathbf{Q}}_{vf} = \int_V \left[\left(\tilde{\mathbf{r}}\dot{\mathbf{A}}\mathbf{X}\right)^{\mathsf{T}} \boldsymbol{\omega} - 2\mathbf{X}^{\mathsf{T}}\mathbf{A}^{\mathsf{T}}\dot{\mathbf{A}}\mathbf{X}\dot{\mathbf{y}}\right] dm \tag{3.67}$$

The integral of the first section corresponds to the translational forces. The first term includes the integral of the skew–symmetric matrix associated to the absolute position $\tilde{\mathbf{r}}$, which is equal to the skew–symmetric matrix associated to the static moment. The second term is directly obtained from the integral of the modes $\mathbf{S}$. Therefore, the translational forces result

$$\bar{\mathbf{Q}}_{vt} = \tilde{\boldsymbol{\omega}}\tilde{\mathbf{m}}\boldsymbol{\omega} - 2\dot{\mathbf{A}}\mathbf{S}\dot{\mathbf{y}} \tag{3.68}$$

The second section are the rotational inertia forces, where the deformed inertia tensor in absolute coordinates is recognised in the first term,

$$\bar{\mathbf{Q}}_{v\theta} = -\tilde{\boldsymbol{\omega}}\mathbf{J}\boldsymbol{\omega} - 2\left(\int_V \tilde{\mathbf{r}}\dot{\mathbf{A}}\mathbf{X}\, dm\right)\dot{\mathbf{y}} \tag{3.69}$$

and the second one is calculated as done for the integral of $\tilde{\mathbf{r}}\mathbf{A}\mathbf{X}$ in Eq. (3.59), but using $\dot{\mathbf{A}}$ instead of $\mathbf{A}$:

$$\int_V \tilde{\mathbf{r}}\dot{\mathbf{A}}\mathbf{X}\, dm = \begin{bmatrix} \dot{\mathbf{A}}_3\mathbf{S}_a^2 - \dot{\mathbf{A}}_2\mathbf{S}_a^3 \\ \dot{\mathbf{A}}_1\mathbf{S}_a^3 - \dot{\mathbf{A}}_3\mathbf{S}_a^1 \\ \dot{\mathbf{A}}_2\mathbf{S}_a^1 - \dot{\mathbf{A}}_1\mathbf{S}_a^2 \end{bmatrix} \tag{3.70}$$

The inertia forces of a rigid body can also be obtained from these expressions, as happened to the mass matrix, if the constant undeformed static moment and inertia tensor are used, and the terms coming from the modal amplitudes are eliminated. The last section of $\mathbf{Q}_v$ is related to the elastic coordinates,

$$\bar{\mathbf{Q}}_{vf} = \left(\int_V \tilde{\mathbf{r}}\dot{\mathbf{A}}\mathbf{X}\, dm\right)^{\mathsf{T}} \boldsymbol{\omega} - 2\left(\int_V \mathbf{X}^{\mathsf{T}}\mathbf{A}^{\mathsf{T}}\dot{\mathbf{A}}\mathbf{X}\, dm\right)\dot{\mathbf{y}} \tag{3.71}$$

and it can be directly calculated from the already used integral of $\tilde{\mathbf{r}}\dot{\mathbf{A}}\mathbf{X}$, and from that of $\mathbf{X}^{\mathsf{T}}\mathbf{A}^{\mathsf{T}}\dot{\mathbf{A}}\mathbf{X}$, which is obtained, as in the absolute formulation, by means of Eq. (3.47).

## 3.5 Efficient calculation of the inertia shape integrals

The calculation of the inertia shape integrals might appear a priori as a cumbersome task, but it is actually much faster than the calculation of the deformation modes themselves. This is because the integrals are calculated by using interpolation functions, which can be integrated independently of the nodal displacements and rotations, therefore allowing for their calculation by means of simple matrix products.

The method for calculating the inertia shape integrals by means of matrix products will be described for three–dimensional isoparametric 2–node beam elements, although it can be generalized for any other type of isoparametric or non–isoparametric finite elements. All the positions and displacements appearing here will be expressed in the local frame of the body, since the interpolation matrices of isoparametric elements are invariant to rotation. In case structural elements are used, the individual orientation of each element within the model must be taken into account by applying the corresponding transformations to the interpolation matrices.

When isoparametric finite elements are used, the same interpolation is applied to geometry and elastic displacements. The position $\bar{\mathbf{r}}$ or the elastic displacement $\bar{\mathbf{r}}_f$ of any point within a beam element $e$ can be therefore interpolated among its values at the end nodes, represented by the nodal coordinates vector $\mathbf{q}^e$ or the nodal displacements vector $\mathbf{q}_f^e$, by means of the same interpolation matrix $\mathbf{N}$ (Bathe, 1995),

$$\bar{\mathbf{r}} = \mathbf{N}\mathbf{q}^e; \qquad \bar{\mathbf{r}}_f = \mathbf{N}\mathbf{q}_f^e \tag{3.72}$$

The interpolation matrix $\mathbf{N}$, according to the type of finite elements used, will depend on one or more parameters or material coordinates. In the particular case of beam elements, only one parameter $\ell$ can be used, corresponding to the position along the undeformed neutral axis, meaning that the same interpolation functions are used for position and displacement, in the $x$, $y$ and $z$ directions.

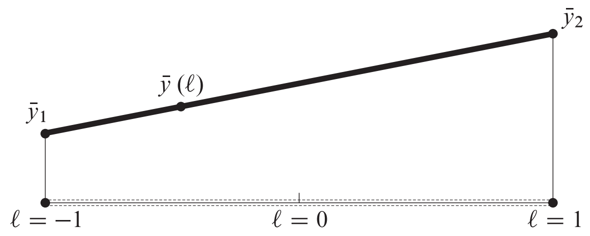


Figure 3.1: Interpolation in a 2–node beam element.

In a general isoparametric beam element of $n_n$ nodes, any of the three components of position or displacement can be calculated at any point as an interpolation among its nodal values. The interpolation is performed by means of $n_n$ interpolation functions $n_i(\ell)$, such that each $n_i$ evaluates to unity at node $i$ and to zero at all other nodes. For the components of the position,

$$\bar{x}(\ell) = \sum_{i=1}^{n_n} \bar{x}_i^e n_i(\ell); \quad \bar{y}(\ell) = \sum_{i=1}^{n_n} \bar{y}_i^e n_i(\ell); \quad \bar{z}(\ell) = \sum_{i=1}^{n_n} \bar{z}_i^e n_i(\ell) \tag{3.73}$$

In the case of a 2–node beam element, if the parametrization is chosen so that the parameter $\ell$ can take values between -1 (first node) and 1 (second node), as shown for the $\bar{y}$ component in Figure 3.1, the $n_i$ functions result,

$$n_1(\ell) = \frac{1}{2}(1-\ell); \qquad n_2(\ell) = \frac{1}{2}(1+\ell) \tag{3.74}$$

If the interpolation described by Eqs.(3.73) and (3.74) is used for the position, provided that $\mathbf{q}^e$ is a vector containing the positions of nodes 1 and 2 of element $e$, it can be written,

$$\bar{\mathbf{r}} = \begin{bmatrix} n_1 & 0 & 0 & n_2 & 0 & 0 \\ 0 & n_1 & 0 & 0 & n_2 & 0 \\ 0 & 0 & n_1 & 0 & 0 & n_2 \end{bmatrix} \begin{Bmatrix} \bar{\mathbf{r}}_1^e \\ \bar{\mathbf{r}}_2^e \end{Bmatrix} = \mathbf{N}\mathbf{q}^e \tag{3.75}$$

leading to the following compact expression for the interpolation matrix $\mathbf{N}$,

$$\mathbf{N} = \begin{bmatrix} n_1\mathbf{I}_3 & n_2\mathbf{I}_3 \end{bmatrix} \tag{3.76}$$

which can be also used for interpolating elastic displacements.

Once the interpolation matrix is completely determined, it can be used for integrating positions or displacements along the finite element. The integral of the position $\bar{\mathbf{r}}$ over the whole volume of an element $V_e$ is,

$$\int_{V_e} \bar{\mathbf{r}}\, dm = \int_{V_e} \mathbf{N}\mathbf{q}^e\, dm = \mathbf{N}^e\mathbf{q}^e \tag{3.77}$$

being $\mathbf{N}^e$ the integral of the interpolation matrix over the volume of the finite element. Considering the density $\rho$ and the cross–sectional area $A$ to be constant along the element,

$$\mathbf{N}^e = \int_{V_e} \mathbf{N}\, dm = \rho A \int_0^{L_e} \begin{bmatrix} n_1\mathbf{I}_3 & n_2\mathbf{I}_3 \end{bmatrix} ds \tag{3.78}$$

being $ds$ the differential of length of arch, and $L_e$ the length of the finite element, measured along its neutral axis. This integral can be evaluated by substitution, since $n_1$ and $n_2$ are functions of $\ell$,

$$ds = \frac{\partial s}{\partial \ell}\, d\ell \tag{3.79}$$

where,

$$\frac{\partial s}{\partial \ell} = \sqrt{\left(\frac{\partial \bar{x}}{\partial \ell}\right)^2 + \left(\frac{\partial \bar{y}}{\partial \ell}\right)^2 + \left(\frac{\partial \bar{z}}{\partial \ell}\right)^2} \tag{3.80}$$

Substituting the components of the local position $\bar{x}$, $\bar{y}$ and $\bar{z}$ by their interpolations, as shown in Eq.(3.73), and differentiating with respect to $\ell$, it is easily demonstrable that,

$$\frac{\partial s}{\partial \ell} = \frac{1}{2}\sqrt{(\bar{x}_2-\bar{x}_1)^2+(\bar{y}_2-\bar{y}_1)^2+(\bar{z}_2-\bar{z}_1)^2} = \frac{L_e}{2} \tag{3.81}$$

The integral results, after the substitution,

$$\rho A\int_0^{L_e}\begin{bmatrix} n_1\mathbf{I}_3 & n_2\mathbf{I}_3\end{bmatrix} ds = \rho A\frac{L_e}{2}\int_{-1}^{1}\begin{bmatrix} n_1\mathbf{I}_3 & n_2\mathbf{I}_3\end{bmatrix} d\ell \tag{3.82}$$

Evaluation of the integral, taking into account that the product $\rho A L_e$ is the mass of the element, $m_e$, yields,

$$\mathbf{N}^e = \frac{m_e}{2}\begin{bmatrix}\mathbf{I}_3 & \mathbf{I}_3\end{bmatrix} \tag{3.83}$$

which means that the integral of the interpolation matrix only depends on the mass of the element.

The same procedure can be employed to integrate the square of the position, the square of the displacement, or the product between them. For instance, the integral of the position times the elastic displacement is,

$$\int_{V_e}\bar{\mathbf{r}}^{\mathsf{T}}\bar{\mathbf{r}}_f\,dm = \int_{V_e}\mathbf{q}^{e\mathsf{T}}\mathbf{N}^{\mathsf{T}}\mathbf{N}\mathbf{q}_f^e\,dm = \mathbf{q}^{e\mathsf{T}}\mathbf{M}^e\mathbf{q}_f^e \tag{3.84}$$

where in this case $\mathbf{M}^e$, the integral of $\mathbf{N}^{\mathsf{T}}\mathbf{N}$, is the mass matrix of the finite element, and it is easy to demonstrate, by following a similar procedure to that used for integrating $\mathbf{N}$, that,

$$\mathbf{M}^e = \frac{m_e}{6}\begin{bmatrix}2\mathbf{I}_3 & \mathbf{I}_3\\ \mathbf{I}_3 & 2\mathbf{I}_3\end{bmatrix} \tag{3.85}$$

The integration of scalar magnitudes or products between them appears in several inertia shape integrals, such as the product of one component of the undeformed position times another component of the elastic displacement. These integrals can be performed by using $\mathbf{N}_s^e$ and $\mathbf{M}_s^e$, the scalar versions of the previously defined integrals,

$$\mathbf{N}_s^e = \int_{V_e}\begin{Bmatrix}n_1 & n_2\end{Bmatrix} dV = \frac{m_e}{2}\begin{Bmatrix}1 & 1\end{Bmatrix} \tag{3.86}$$

$$\mathbf{M}_s^e = \int_{V_e}\begin{Bmatrix}n_1\\ n_2\end{Bmatrix}\begin{Bmatrix}n_1 & n_2\end{Bmatrix} dV = \frac{m_e}{6}\begin{bmatrix}2 & 1\\ 1 & 2\end{bmatrix} \tag{3.87}$$

With these four matrices, the mass integral of any scalar or vector magnitude and products between them over the volume of an element can be obtained by simple multiplication.

Both $\mathbf{N}^e$ and $\mathbf{M}^e$ matrices and their scalar versions are defined for a single element, so that integrals over the full body can be obtained as the sum of the integrals over all the finite elements. For example, the integrals of $\bar{\mathbf{r}}$ or $\bar{\mathbf{r}}^{\mathsf{T}}\bar{\mathbf{r}}$ over the total volume $V$ of a body discretized into $n_e$ elements, are,

$$\int_V \bar{\mathbf{r}}\, dm = \sum_{e=1}^{n_e} \mathbf{N}^e \mathbf{q}^e; \qquad \int_V \bar{\mathbf{r}}^{\mathsf{T}} \bar{\mathbf{r}}\, dm = \sum_{e=1}^{n_e} \mathbf{q}^{e\mathsf{T}} \mathbf{M}^e \mathbf{q}^e \tag{3.88}$$

If only one component is to be integrated, or the product of one component times another, the scalar versions must be used. For one component

$$\int_V \bar{r}_i\, dm = \sum_{e=1}^{n_e} \mathbf{N}_s^e \mathbf{q}_i^e, \quad i = 1, 2, 3 \tag{3.89}$$

and for two components

$$\int_V \bar{r}_i \bar{r}_j\, dm = \sum_{e=1}^{n_e} \mathbf{q}_i^{e\mathsf{T}} \mathbf{M}_s^e \mathbf{q}_j^e, \quad i, j = 1, 2, 3 \tag{3.90}$$

where each $\mathbf{q}_i^e$ and $\mathbf{q}_j^e$ are $2 \times 1$ vectors containing only the $i$ or $j$ components of the nodal coordinates $\mathbf{q}^e$.

The application of the described procedure for the calculation of the inertia shape integrals is very simple and efficient. First, the mass of all finite elements must be calculated, which is easy since the density, the cross–sectional area and the undeformed positions of the nodes are known. Then, any integral can be obtained as the sum of the integrals over all the finite elements.

Each column of **S** is the integral of a mode shape, which is an elastic displacement. The integrals of all modes can be performed simultaneously since the interpolation matrices, and consequently their integrals, are the same for all the modes. If a matrix $\mathbf{X}^e$ is defined for each finite element $e$, containing the modal displacements for the two nodes of element $e$, it can be written,

$$\mathbf{S} = \int_V \mathbf{X}\, dm = \sum_{e=1}^{n_e} \mathbf{N}^e \mathbf{X}^e \tag{3.91}$$

The integrals of the undeformed position times the mode shapes involve products of scalars and vectors, therefore they must be subdivided into three scalar by scalar products in order to perform the integration:

$$\mathbf{S}^i = \int_V \bar{r}_{ui} \mathbf{X}\, dm = \int_V \begin{bmatrix} \bar{r}_{ui} \mathbf{X}_1 \\ \bar{r}_{ui} \mathbf{X}_2 \\ \bar{r}_{ui} \mathbf{X}_3 \end{bmatrix} dm = \begin{bmatrix} \mathbf{S}_1^i \\ \mathbf{S}_2^i \\ \mathbf{S}_3^i \end{bmatrix}, \quad i = 1, 2, 3 \tag{3.92}$$

Then each of the three rows can be integrated by using the $\mathbf{M}_s^e$ matrices,

$$\mathbf{S}_j^i = \sum_{e=1}^{n_e} \mathbf{q}_{ui}^{e\mathsf{T}} \mathbf{M}_s^e \mathbf{X}_j^e, \quad i, j = 1, 2, 3 \tag{3.93}$$

The integrals of products between directions of the modes are also performed in the same way,

$$\mathbf{S}^{ij} = \sum_{e=1}^{n_e} \mathbf{X}_i^{e\mathsf{T}} \mathbf{M}_s^e \mathbf{X}_j^e, \quad i, j = 1, 2, 3 \tag{3.94}$$

Even the rigid body integrals such as the planar inertia tensor can be obtained by using this method, although their values are generally already known.

These sums can be calculated directly or by assembling the matrices by following the standard finite element procedure. As an example, if all the mass matrices $\mathbf{M}^e$ are assembled into a $\mathbf{M}^*$ matrix, and a $\mathbf{X}^*$ matrix is also created containing the modal displacements at all finite element nodes, Eq. (3.91) can be written as,

$$\mathbf{S} = \mathbf{M}^* \mathbf{X}^* \tag{3.95}$$

A similar procedure of assembly can be used for all the integrals.

## 3.6 Numerical example

The Iltis vehicle (Frik et al., 1993), the third example used in the previous chapter, is used also here as the base system for the tests. In this case, all the 12 possible flexible bodies are included, and instead of using a fixed mesh size, the flexible elements are discretized into a variable number $n_e$ of finite elements per bar. The A–arm, as previously pointed out, is modeled as two bars, coincident at the hub connection, and one additional element for the damper attachment, so that it has $2n_e + 1$ finite elements. This makes a total, for the 12 flexible bodies, of $16n_e + 4$ finite elements, and 64 modes.

In the test, the Iltis performs the same maneuver described in the previous chapter. The simulation is carried out by using both the absolute and the relative formulations, either with the projection method or the inertia shape integrals preprocessing, with a time–step of 10 ms, and with four different finite element discretizations (5, 10, 50 and 100 elements per bar). The full model in absolute coordinates, as pointed out in the previous chapter, has 304 variables, whereas the model in relative coordinates has a total of 98 variables.

The method used for calculating the inertia terms does not practically affect the results obtained from the simulations. As can be seen in Table 3.1 and Figure 3.2, the preprocessing (P) makes the CPU–time completely invariant with respect to the finite element mesh size. Both the absolute and the relative formulations benefit from this improvement, especially in the case of large finite element models, where the B matrix projection (B) takes a significantly larger amount of time.

Table 3.1: CPU–times for different finite element mesh sizes

| N° elements $n_e$ | 5 | 10 | 50 | 100 |
|---|---|---|---|---|
| Absolute, B | 4.19 | 4.48 | 7.58 | 12.59 |
| Absolute, P | 3.85 | 3.85 | 3.85 | 3.85 |
| Relative, B | 1.11 | 1.48 | 4.32 | 8.21 |
| Relative, P | 0.79 | 0.79 | 0.79 | 0.79 |

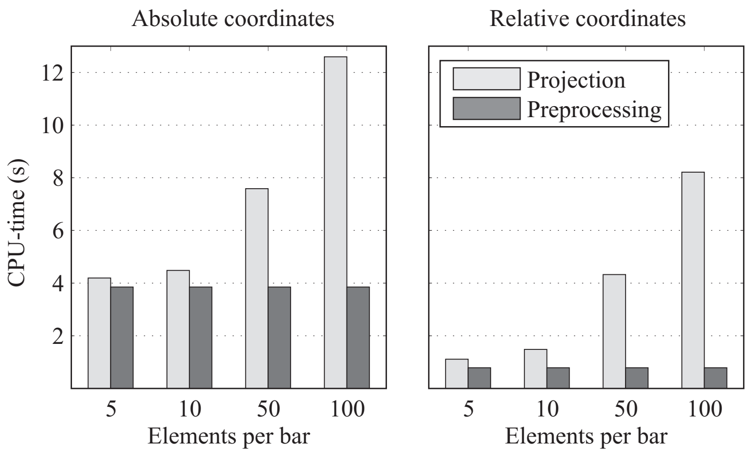


Figure 3.2: CPU–time vs. number of finite elements

## 3.7 Conclusions and criteria of use

From the obtained results, it can be said that the inertia shape integrals preprocessing very significantly improves the performance in all cases. When very large finite element models are used, the projection method can become unpractical, whereas the preprocessing one keeps the CPU–time dependent only on the number of modes. Apart from the higher difficulty of implementation, the only drawback of the method could be the preprocessing time but, in practice, it is negligible, especially if compared to the calculation of the mode shapes by solving the finite element system, since all the integrals can be obtained by direct matrix multiplication. In the present work, the preprocessing has been done in MATLAB, and it takes less than 0.02 s for an A–arm with 100 elements per bar (i.e. 201 elements).

The projection method, on the other hand, is much easier to implement, and the only input data it needs from finite element software are the mass and mode shapes matrices, along with the undeformed local positions of the nodes. This might make it more convenient for certain applications where the implementation time is more relevant, as long as the size of the finite element models is not too large.

In what respects the comparison between the formulations in absolute and relative coordinates, the latter seems to improve the advantage when using the preprocessing method, despite of its more involved inertia terms. It achieves a CPU–time around 5 times faster than the formulation in absolute coordinates, whereas in the case of projection with ten elements, as used in the previous chapter, the ratio is about 3. It is also observed that the improvement with respect to the preprocessing method is always more noticeable in the case of the formulation in relative coordinates; for instance, in the case of 100 elements per bar, the simulation in absolute coordinates is three times faster when using preprocessing, whereas in relative coordinates it becomes up to ten times faster.

# Chapter 4

# Geometric Stiffening

## 4.1 Introduction

All the methods addressed so far allow for simulating flexible multibody systems in a very efficient way. However, the use of component mode synthesis to reduce the size of the finite element models, due to the linearization of the elastic forces, limits their use to applications where the elastic deformations remain small. In order to accurately simulate systems with larger deformations, a possible solution could be the use of fully nonlinear methods, such as the absolute formulations or nonlinear finite element analysis, but none of these techniques is suitable for real–time simulation due to their elevated CPU cost.

In specific applications, involving beams under high rotational speeds, such as helicopter rotor or turbine blades, a stiffening effect appears due to the geometrical nonlinearity. This effect has been studied by many authors, like Kane et al. (1987), Mayo et al. (1995, 2004), Sharf (1996), Valembois et al. (1997), Zahariev (2000, 2002) or Shi et al. (2001). Helicopter rotor blades, a typical example, are bent by their own weight, but the rotation speed makes them rise toward the horizontal position, due to centrifugal forces, as if the bending stiffness is increasing. In a linear model, this effect is not captured due to the absence of coupling between axial and transversal deformation, which implies that rotational speed has no effect on bending, but only on the radial displacement.

There exist several techniques aiming for including this effect in beams, without resorting to fully nonlinear methods, thus allowing for extending the range of usability of the FFR formulations. The most general of these techniques is the substructuring method (Wu and Haug, 1988), which consists of dividing the beam into small pieces, being each of them a flexible body with its own frame of reference and mode shapes. Each substructure is clamped to the adjacent ones by means of the so–called *bracket joints*, sharing the points and frames of reference, thus making the full set behave as a whole beam. The main drawback of this method is the large number of coordinates needed, although this problem is reduced by using relative coordinates (Kim and Haug, 1988); moreover, when this technique is used, the number of deformation modes required per substructure remains small, since it is obvious that the higher frequency modes become unnecessary as long as the length of the divisions decreases.

Other less general methods are based on introducing nonlinearity into the elastic forces or into the modeling of the flexible body. In this chapter, the implementation of two of these techniques is addressed, along with substructuring in relative coordinates, and the results are compared to reference solutions obtained with fully nonlinear methods, such as the ANCF or the finite element method (FEM).

## 4.2 Substructuring

The substructuring technique allows for introducing nonlinear effects into FFR formulations in a completely general way, requiring no modifications to the original formulation. As can be seen in Figure 4.1, the beam is divided into several elements, which are interconnected by means of bracket joints, in such a way that the output frame of each element coincides with the input frame of the next one. If the formulation in natural coordinates is used, this means that the point and the three unit vectors defined at each bracket joint are shared between the two adjacent bodies. In relative coordinates, the only relative coordinates appearing at a bracket joint are the static modal amplitudes.

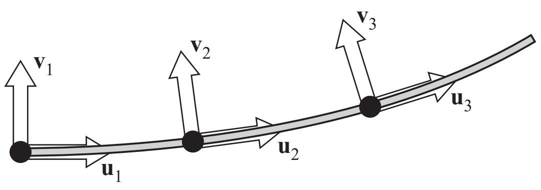


Figure 4.1: Beam divided into three substructures.

Since the substructures have a short length, they are modeled by using a small number of deformation modes. If substructures are modeled using $n_m$ modes, the use of natural coordinates implies that each additional substructure introduces $12 + n_m$ extra variables, along with 12 algebraic constraints, whereas in relative coordinates only $n_m$ variables are added; for this reason, only substructuring in relative coordinates is considered here.

## 4.3 Nonlinear stiffness matrix

The case of a planar flexible beam in natural coordinates is to be described for the sake of simplicity. In the case of a two dimensional beam, the floating frame of reference is described by a point and two vectors **u** and **v**, making a total of six reference coordinates. The position of an arbitrary point of the beam remains, as in the three dimensional case

$$\mathbf{r} = \mathbf{r}_0 + \mathbf{A}\left(\bar{\mathbf{r}}_u + \bar{\mathbf{r}}_f\right) \tag{4.1}$$

where $\mathbf{r}_0$ is the position of the local frame origin, $\mathbf{A}$ the rotation matrix defined now by the two orthogonal local unit vectors $\mathbf{u}$ and $\mathbf{v}$, $\bar{\mathbf{r}}_u$ the undeformed position in local coordinates, and $\bar{\mathbf{r}}_f$ the local elastic displacement (see Figure 4.2).

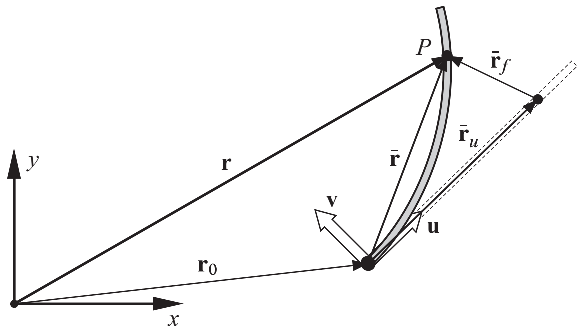


Figure 4.2: Deformed 2D beam.

The elastic displacements field $\bar{\mathbf{r}}_f\,(x, y)$ of an Euler–Bernoulli beam takes the following vector form (Sharf, 1996):

$$\bar{\mathbf{r}}_f\,(x, y) = \begin{Bmatrix} u \\ v \end{Bmatrix} = \begin{Bmatrix} u_0 - y v_0' \\ v_0 \end{Bmatrix} \tag{4.2}$$

where $u_0$ and $v_0$ are the axial and transversal displacements of the neutral axis, and the apostrophe indicates differentiation with respect to the $x$ coordinate. The nonlinear strain–displacement relationship in $x$ direction can be expressed as,

$$\varepsilon_{xx} = u' + \frac{1}{2}\left(u'^2 + v'^2\right) \cong u' + \frac{1}{2}v'^2 \tag{4.3}$$

where the term $u'^2$ is dropped since it is much smaller than $u'$. The elastic potential of the beam, after applying the stress–strain relation, is,

$$U = \frac{1}{2}\int_V E\varepsilon_{xx}^2\, dV \tag{4.4}$$

where $E$ is the Young modulus and $V$ is the volume of the beam. Introduction of the displacement field described by Eq. (4.2) in the strain–displacement relation, yields the deformation energy of the beam in terms of the deformed shape of the neutral axis (Sharf, 1996),

$$\begin{aligned} U = & \underbrace{\frac{1}{2}\int_0^L EAu_0'^2\, dx + \frac{1}{2}\int_0^L EIv_0''^2\, dx}_{Linear\ formulation} \\ & + \underbrace{\frac{1}{2}\int_0^L EAu_0' v_0'^2\, dx}_{First\ nonlinear} + \underbrace{\frac{1}{8}\int_0^L EAv_0'^4\, dx}_{Second\ nonlinear} \end{aligned} \tag{4.5}$$

with $A$ the cross–sectional area and $I$ the second moment of area with respect to the neutral axis. Different levels of approximation can be achieved depending on which terms of Eq. (4.5) are kept: they are discussed in the following subsections.

### 4.3.1 Linear formulation

The linear formulation includes only the first two terms of Eq. (4.5) in the elastic potential, neglecting the higher order ones. Introducing the finite element discretization into the equation and integrating the interpolation functions, the following expression can be obtained for the elastic potential in terms of the finite element coordinates,

$$U = \frac{1}{2}\mathbf{q}_f^{\mathsf{T}}\mathbf{K}_L^*\mathbf{q}_f \tag{4.6}$$

Here, $\mathbf{K}_L^*$ is the linear stiffness matrix, which is constant, and $\mathbf{q}_f$ is a vector containing the nodal displacements of the whole beam. This potential can be projected into the modal base by using matrix $\mathbf{X}$,

$$U = \frac{1}{2}\mathbf{y}^{\mathsf{T}}\mathbf{X}^{\mathsf{T}}\mathbf{K}_L^*\mathbf{X}\mathbf{y} = \frac{1}{2}\mathbf{y}^{\mathsf{T}}\mathbf{K}_L\mathbf{y} \tag{4.7}$$

By differentiation of the elastic potential, an expression for the elastic forces is obtained,

$$\mathbf{F}_{el} = -\left(\frac{\partial U}{\partial \mathbf{y}}\right)^{\mathsf{T}} = -\mathbf{K}_L\mathbf{y} \tag{4.8}$$

which is a linear relationship between the forces and the modal amplitudes.

A closer look to the elastic potential expression used in this formulation, constituted by the first two terms of Eq. (4.5), reveals the cause of its inability to capture the geometric stiffening effect: axial and transversal displacements separately contribute to the deformation energy. Only transversal forces can produce transversal displacements, therefore the axial forces introduced by the rotation have no effect on the deflection.

### 4.3.2 First nonlinear formulation

When the third term of Eq. (4.5) is considered too, the coupling between axial and transversal deformation is introduced through the integral of $u_0' {v_0'}^2$. This enables to capture the geometric stiffening effect, since it couples the longitudinal and the transversal displacements, but at the cost of a non–constant stiffness matrix.

The same steps as in the linear formulation must be carried out to obtain the elastic potential: the $u_0$ and $v_0$ derivatives are substituted by their finite element interpolations, and the integrals are evaluated; then, writing it in matrix form (Mayo et al., 1995, 2004),

$$U = \frac{1}{2}\mathbf{q}_f^{\mathsf{T}}\left(\mathbf{K}_L^* + \mathbf{K}_G^*\right)\mathbf{q}_f \tag{4.9}$$

The geometric stiffness matrix $\mathbf{K}_G^*$ is variable, and must be calculated at every time–step. In case that the axial displacement $u_0$ has a linear distribution, the strain is constant along the whole beam, and $\mathbf{K}_G^*$ can be expressed as the product of a scalar variable times a constant matrix. But in any other case, this is only applicable to each finite element, and the matrix must be assembled at every time–step, which is rather inefficient.

It is better to express $u_0$ and $v_0$ in terms of the mode shapes and then carry out the spatial integration. First, the neutral axis displacements are approximated by the modal superposition,

$$\begin{aligned} u_0(x) &= \sum_{i=1}^{ns} \phi_i^l(x)\eta_i + \sum_{j=1}^{nd} \psi_j^l(x)\xi_j \\ v_0(x) &= \sum_{i=1}^{ns} \phi_i^t(x)\eta_i + \sum_{j=1}^{nd} \psi_j^t(x)\xi_j \end{aligned} \tag{4.10}$$

where the superindices $l$ and $t$ indicate longitudinal or transversal component, respectively. These approximated displacements are then used to calculate the integral. The analytical functions of the mode shapes are usually known for a beam and, therefore, they can be directly integrated. In the case that the modes are finite element displacement vectors, the integrals must be calculated by using the interpolation functions. The geometric stiffness matrix, already projected into the modal subspace, takes the following linear combination form, with the modal amplitudes as coefficients,

$$\mathbf{K}_G = \sum_{i=1}^{ns} \eta_i \mathbf{K}_{Gi} + \sum_{j=1}^{nd} \xi_j \mathbf{K}_{Gj} \tag{4.11}$$

where all the $\mathbf{K}_{Gi}$ and $\mathbf{K}_{Gj}$ matrices are constant, and have the form,

$$\mathbf{K}_{Gi} = \int_0^L EA\phi_i'^l \begin{Bmatrix} \phi_1'^t \\ \vdots \\ \phi_{ns}'^t \\ \psi_1'^t \\ \vdots \\ \psi_{nd}'^t \end{Bmatrix} \begin{Bmatrix} \phi_1'^t & \cdots & \phi_{ns}'^t & \psi_1'^t & \cdots & \psi_{nd}'^t \end{Bmatrix} dx \tag{4.12}$$

with $\psi_j'^l$ instead of $\phi_i'^l$ for $\mathbf{K}_{Gj}$. These matrices are non–zero for mode $i$ or $j$ only if the mode is longitudinal, so that there is one matrix for each axial mode. According to this, in order to obtain a nonzero $\mathbf{K}_G$ matrix, this method needs to incorporate at least one axial mode.

Differentiation of the elastic potential with respect to $\mathbf{y}$, neglecting the term which contains the derivative of $\mathbf{K}_G$, yields the elastic forces vector,

$$\mathbf{F}_{el} = -\left(\frac{\partial U}{\partial \mathbf{y}}\right)^{\mathsf{T}} = -\left(\mathbf{K}_L + \mathbf{K}_G\right)\mathbf{y} \tag{4.13}$$

The modifications with respect to the linear formulation are minimal. All the integrals of Eq. (4.12) must be calculated in a preprocessing stage, thus obtaining one constant matrix for each axial mode. Since the stiffness matrix is no longer constant, it must be calculated at every integrator iteration by adding the variable $\mathbf{K}_G$, obtained from Eq. (4.11), to the linear stiffness matrix $\mathbf{K}_L$.

### 4.3.3 Second nonlinear formulation

In this formulation, the four terms of the elastic energy in Eq. (4.5) are considered, being the most suitable for severe deformation conditions but, logically, at the cost of a higher computational effort.

$$U = \frac{1}{2}\mathbf{q}_f^{\mathsf{T}}\left(\mathbf{K}_L^* + \mathbf{K}_G^* + \mathbf{K}_H^*\right)\mathbf{q}_f \tag{4.14}$$

The inclusion of the higher order term adds a second–order nonlinear matrix $\mathbf{K}_H^*$, and the elastic forces are obtained by differentiation,

$$\mathbf{F}_{el} = -\left(\frac{\partial U}{\partial \mathbf{q}_f}\right)^{\mathsf{T}} = -\left(\mathbf{K}_L^* + \mathbf{K}_G^* + \mathbf{K}_H^*\right)\mathbf{q}_f + \mathbf{Q}_g \tag{4.15}$$

where all the terms depending on the derivatives of the variable $\mathbf{K}$ matrices are grouped into the generalized nonlinear forces vector $\mathbf{Q}_g$. The main problem of this formulation is that it needs a high number of axial modes to obtain accurate results (Mayo et al., 1995, 2004), making its use inefficient.

## 4.4 Foreshortening formulation

The axial shortening of a beam due to its deflection is known as foreshortening (Figure 4.3). This effect cannot be captured by using the linear or first nonlinear formula-

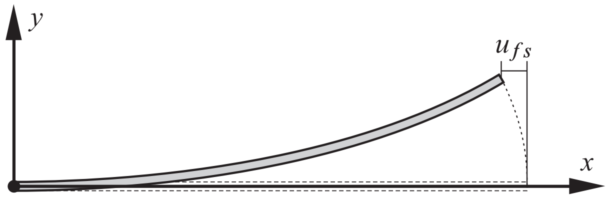


Figure 4.3: Foreshortening produced by deflection.

tions. The explicit inclusion of the foreshortening effect in the model leads to a simpler and more efficient method (Mayo et al., 1995, 2004), and provides the same level of accuracy as the second nonlinear formulation.

## 4.4.1 Calculation of the foreshortening

The longitudinal displacement of any point of the neutral axis can be divided into the axial deformation produced by the actual axial forces, $s$, and the shortening produced by the deflection $u_{fs}$,

$$u_0 = s + u_{fs} \tag{4.16}$$

The foreshortening of a curve infinitesimal $ds$ can be obtained, as shown in Figure 4.4, from the projection of $ds - dx$ into the undeformed neutral axis,

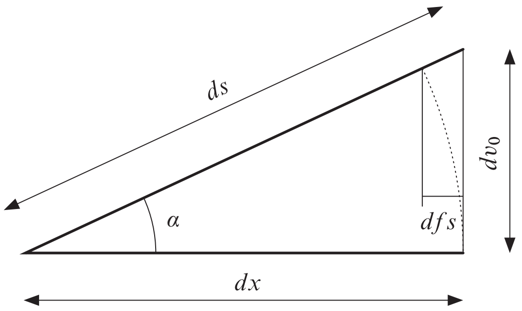


Figure 4.4: Foreshortening of a curve infinitesimal.

$$dfs = (ds - dx)\cos\alpha = \left(1 - \frac{dx}{ds}\right)dx = \left(1 - \frac{1}{\sqrt{1 + {v_0'}^2}}\right)dx \tag{4.17}$$

This expression can be simplified for small values of $v_0'$, by developing it into a Taylor series up to the second order,

$$dfs \approx \frac{1}{2}{v_0'}^2 dx \tag{4.18}$$

The total shortening accumulated from a reference point $x_0$, which has zero axial displacement, is then obtained by integration,

$$u_{fs}(x) = -\frac{1}{2}\int_{x_0}^{x} {v_0'}^2\, dx \tag{4.19}$$

Substituting the longitudinal displacement of Eq. (4.16) into Eq. (4.5), yields the following expression for the elastic potential,

$$U = \frac{1}{2}\int_0^L EAs'^2\, dx + \frac{1}{2}\int_0^L EIv_0''^2\, dx \tag{4.20}$$

It is observed that the elastic energy has the same form as in the linear formulation, although the meaning is different. The stiffness matrix is the same as the one used for the linear case $\mathbf{K}_L$, and so happens with the elastic forces. Therefore, the stiffening effect does not appear now in the elastic forces: it is translated to the inertia and constraint forces, since the foreshortening is introduced at kinematic level.

Using the finite element method to discretize the beam with 2D beam elements, the neutral axis displacement within a finite element $e$, $\bar{\mathbf{r}}^e_{f0}$, can be interpolated from its nodal displacements, $\mathbf{q}^e_f$, by means of the interpolation matrix, $\mathbf{N}$, which can be split into longitudinal and transversal interpolation submatrices, $\mathbf{N}_l$ and $\mathbf{N}_t$,

$$\bar{\mathbf{r}}^e_{f0} = \begin{Bmatrix} u_0 \\ v_0 \end{Bmatrix} = \mathbf{N}\mathbf{q}^e_f = \begin{bmatrix} \mathbf{N}_l \\ \mathbf{N}_t \end{bmatrix} \mathbf{q}^e_f \tag{4.21}$$

where $u_0$ and $v_0$ are the local components of $\bar{\mathbf{r}}^e_{f0}$. In order to calculate the total foreshortening on a finite element, the nodal displacement must be modified so that,

$$\bar{\mathbf{r}}^e_{f0} = \begin{Bmatrix} u_0 \\ v_0 \end{Bmatrix} = \begin{bmatrix} \mathbf{N}_l \\ \mathbf{N}_t \end{bmatrix} \mathbf{q}^e_f + \begin{Bmatrix} u^e_{fs} \\ 0 \end{Bmatrix} \tag{4.22}$$

where $u^e_{fs}$ is the foreshortening produced in that finite element by its own deflection, and can be calculated by applying Eq. (4.19) over the whole length of the element, $L^e$. Substituting $v_0'$ by its interpolation,

$$u^e_{fs} = -\frac{1}{2}\int_0^{L^e} \mathbf{q}^{e\mathsf{T}}_f \mathbf{N}'^{\mathsf{T}}_t \mathbf{N}'_t \mathbf{q}^e_f \, dx = -\frac{1}{2}\mathbf{q}^{e\mathsf{T}}_f \mathbf{H}^e \mathbf{q}^e_f \tag{4.23}$$

The shortening suffered by one element is then a quadratic function of the nodal coordinates, where $\mathbf{H}^e$ is a constant matrix depending only on the transversal interpolation functions and the length of the element, defined as

$$\mathbf{H}^e = \int_0^{L^e} \mathbf{N}'^{\mathsf{T}}_t \mathbf{N}'_t \, dx \tag{4.24}$$

The total shortening accumulated by the finite elements located between the reference node (with zero axial displacement) and the finite element $n$, itself included, is the sum of all the element–level shortenings, as shown in Figure 4.5.

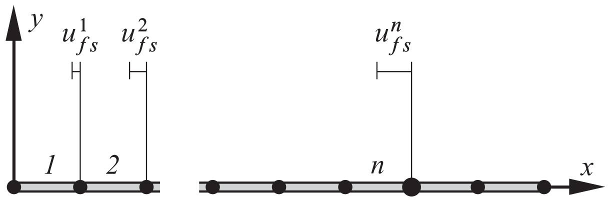


Figure 4.5: Accumulated foreshortening at element $n$.

$$u_{fs}^{n} = -\frac{1}{2}\sum_{e=1}^{n} \mathbf{q}_f^{eT}\mathbf{H}^e\mathbf{q}_f^e \tag{4.25}$$

This expression can be written in matrix form for each element, assembled for all the finite element coordinates of the beam, and then projected into the modal subspace,

$$u_{fs}^{n} = -\frac{1}{2}\mathbf{q}_f^{\mathsf{T}}\mathbf{H}_a^n\mathbf{q}_f = -\frac{1}{2}\mathbf{y}^{\mathsf{T}}\mathbf{X}^{\mathsf{T}}\mathbf{H}_a^n\mathbf{X}\mathbf{y} = -\frac{1}{2}\mathbf{y}^{\mathsf{T}}\mathbf{G}^n\mathbf{y} \tag{4.26}$$

If analytical functions are available for the mode shapes, these $\mathbf{G}^n$ matrices can be directly calculated by using the second expression of Eq. (4.10) to evaluate Eq. (4.19),

$$\mathbf{G}^n = \int_0^{L^n} \begin{Bmatrix} \phi_1'^t \\ \vdots \\ \phi_{ns}'^t \\ \psi_1'^t \\ \vdots \\ \psi_{nd}'^t \end{Bmatrix} \begin{Bmatrix} \phi_1'^t & \cdots & \phi_{ns}'^t & \psi_1'^t & \cdots & \psi_{nd}'^t \end{Bmatrix} dx \tag{4.27}$$

where $L^n$ is the length of the beam from the reference point to the end node (node $i$) of finite element $n$. If the modes are finite element displacement vectors, the integrals must be calculated by using the interpolation functions.

## 4.4.2 Inertia terms

In the previous chapter, the use of inertia shape integrals is demonstrated to be the most efficient way to calculate the inertia terms, especially when the finite element models are very large. However, the shape integrals are no longer constant if foreshortening is considered, making the method much more involved. Moreover, in the case of beams, the finite element models are usually small, in such a way that the projection method is efficient enough to achieve real–time performance. For these reasons, the inertia terms are here calculated by using the $\mathbf{B}^*$ matrix projection method. The foreshortening is introduced at the calculation of the $\mathbf{B}^*$ matrix and the $\dot{\mathbf{B}}^*\dot{\mathbf{q}}$ or $\dot{\mathbf{B}}^*\mathbf{Z}$ vector, which are later used to obtain the mass matrix and the inertia forces vector as described in the second chapter, by using Eq. (2.80) and Eq. (2.81).

The expression previously obtained for the accumulated foreshortening at a finite element $n$ can be used to obtain the elastic displacement of its end node $i$

$$\bar{\mathbf{r}}_{f0}^{i} = \mathbf{q}_f^i + \begin{Bmatrix} u_{fs}^n \\ 0 \end{Bmatrix} = \mathbf{X}^i\mathbf{y} + \begin{Bmatrix} u_{fs}^n \\ 0 \end{Bmatrix} = \mathbf{X}^i\mathbf{y} - \frac{1}{2}\begin{Bmatrix} 1 \\ 0 \end{Bmatrix}\mathbf{y}^{\mathsf{T}}\mathbf{G}^n\mathbf{y} \tag{4.28}$$

which can be rewritten in terms of a new variable $\mathbf{X}_{fs}^{i}$ matrix, which depends linearly on the modal amplitudes **y**,

$$\mathbf{X}^i_{fs} = \mathbf{X}^i - \frac{1}{2}\begin{Bmatrix}1\\0\end{Bmatrix}\mathbf{y}^{\mathsf{T}}\mathbf{G}^n \implies \bar{\mathbf{r}}^i_{f0} = \mathbf{X}^i_{fs}\mathbf{y} \tag{4.29}$$

Substituting the elastic displacement given by this expression into Eq. (4.1) yields the new nodal velocity

$$\dot{\mathbf{r}}^{i*} = \dot{\mathbf{r}}_0 + \dot{\mathbf{A}}\left(\bar{\mathbf{r}}^i_u + \mathbf{X}^i_{fs}\mathbf{y}\right) + \mathbf{A}\left(\mathbf{X}^i_{fs}\dot{\mathbf{y}} + \dot{\mathbf{X}}^i_{fs}\mathbf{y}\right) \tag{4.30}$$

The term within the last parentheses can be found to be, if the symmetry of the $\mathbf{G}^n$ matrices is taken into account,

$$\mathbf{X}^i_{fs}\dot{\mathbf{y}} + \dot{\mathbf{X}}^i_{fs}\mathbf{y} = \left(\mathbf{X}^i - \begin{Bmatrix}1\\0\end{Bmatrix}\mathbf{y}^{\mathsf{T}}\mathbf{G}^n\right)\dot{\mathbf{y}} = \bar{\mathbf{X}}^i_{fs}\dot{\mathbf{y}} \tag{4.31}$$

This result finally enables to define the $\mathbf{B}^i$ matrix of a node for the foreshortening formulation, either in natural coordinates:

$$\mathbf{B}^i = \begin{bmatrix}\mathbf{I}_2 & \bar{r}^i_1\mathbf{I}_2 & \bar{r}^i_2\mathbf{I}_2 & \mathbf{A}\bar{\mathbf{X}}^i_{fs}\end{bmatrix} \tag{4.32}$$

or in relative coordinates:

$$\mathbf{B}^i = \begin{bmatrix}\mathbf{I}_2 & -\tilde{\mathbf{r}}^i & \mathbf{A}\bar{\mathbf{X}}^i_{fs}\end{bmatrix} \tag{4.33}$$

where two differences with respect to the original matrices are found. Firstly, the axial component of the local position $\bar{r}^i_1$ is modified by the foreshortening, thus affecting the global position $\mathbf{r}^i$. Secondly, the mode shapes matrix in the last block is a modified version of **X**, given by Eq. (4.31).

The calculation of the centrifugal and Coriolis forces vector is straightforward. In natural coordinates, the $\dot{\mathbf{B}}^i\dot{\mathbf{q}}$ product is evaluated at all the nodes and assembled,

$$\dot{\mathbf{B}}^i\dot{\mathbf{q}} = 2\left(\dot{\mathbf{A}}\bar{\mathbf{X}}^i_{fs} + \mathbf{A}\dot{\mathbf{X}}^i_{fs}\right)\dot{\mathbf{y}} \tag{4.34}$$

and if relative coordinates are chosen, the calculation of $\dot{\mathbf{B}}^i\mathbf{Z}$ is completely analogous

$$\dot{\mathbf{B}}^i\mathbf{Z} = \boldsymbol{\omega} \times \left(\boldsymbol{\omega} \times \mathbf{r}^i\right) + 2\left(\dot{\mathbf{A}}\bar{\mathbf{X}}^i_{fs} + \mathbf{A}\dot{\mathbf{X}}^i_{fs}\right)\dot{\mathbf{y}} \tag{4.35}$$

These changes affect the mass matrix, the velocity–dependent inertia forces, and the applied forces as well, since they depend in turn on the **B** matrix at the point of application. Moreover, those constraints involving nodes undergoing foreshortening must also be modified, since transversal modes affect the beam length. Therefore, the geometric stiffening effect is considered now through inertia and constraint forces, instead of through the elastic forces, as happened in the first and second nonlinear formulations. If relative coordinates are used, it must be taken into account the fact that the bending modes introduce an axial displacement, which affects the kinematic relations at the joints, and also in the Jacobian of the constraints vector, if any cut joint is placed at a point including foreshortening.

## 4.5 Examples and results

The example system is a typical case of geometric stiffening, studied by many authors, such as Kane et al. (1987), Mayo et al. (1995, 2004), Sharf (1996), Valembois et al. (1997), Zahariev (2000, 2002), or Shi et al. (2001): a beam pinned at one of its ends, as shown in Figure 4.6, which rotates an angle $\theta(t)$ about the origin,

$$\theta(t) = \begin{cases} \dfrac{\omega_s}{T_s}\left(\dfrac{t^2}{2} + \left(\dfrac{T_s}{2\pi}\right)^2 \left[\cos\left(\dfrac{2\pi t}{T_s}\right) - 1\right]\right) & 0 \le t < T_s \\ \omega_s\left(t - \dfrac{T_s}{2}\right) & T_s \le t \end{cases} \tag{4.36}$$

The characteristics of the beam are the following: length $L$=10 m, cross–sectional area $A$=4 · 10$^{-4}$ m$^2$, second moments of area $I_{y,z}$=2 · 10$^{-7}$ m$^4$, density $\rho$=3000 Kg/m$^3$ and Young modulus $E$=7 · 10$^{10}$ N/m$^2$.

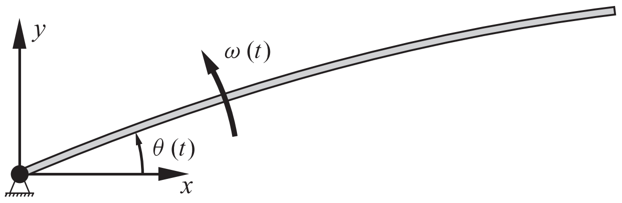


Figure 4.6: Spin–up beam.

### 4.5.1 Two–dimensional case

In all the cited previous works, this example is treated as a 2D problem, studying the behavior of the beam in the $xy$ plane in absence of gravity acceleration. By using the first nonlinear and foreshortening formulations, both of them in either absolute or relative coordinates, the in–plane tip deflection is obtained for $T_s$=15 s and $\omega_s$=6 rad/s. The beam is discretized into ten finite elements, approximating the elastic displacements by using two transversal modes, one static and another dynamic, defined in the local $xy$ plane. The results are compared to a reference solution calculated by means of the ANCF, a fully nonlinear formulation that automatically captures the geometric stiffening effect. In order to obtain the reference solution, the beam is discretized into 15 elements, using the ANCF–based 2D beam element developed by Omar and Shabana (2001). Since the tip deflections obtained are nearly the same regardless of the type of coordinates used, only the plots for absolute coordinates will be shown here.

The results obtained for the linear formulation reveal that, as expected, it cannot account for the geometric stiffening effect. As can be seen in Figure 4.7, the tip deflection becomes too large, crashing the simulation before its end.

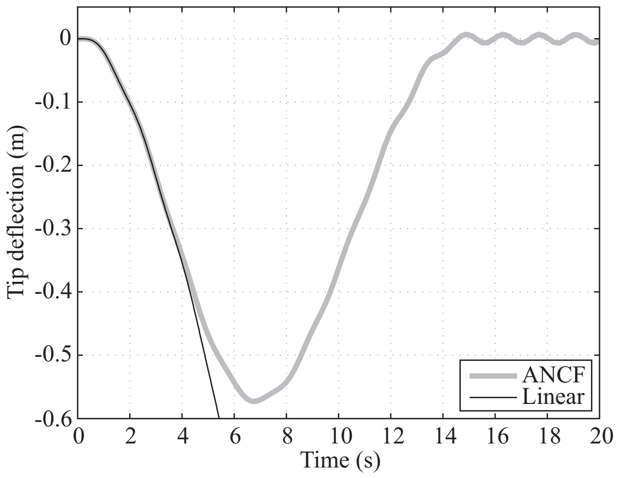


Figure 4.7: Linear formulation vs. ANCF in the first example.

The first nonlinear formulation needs to include at least one axial mode, as the geometric stiffness matrix depends on the axial deformation. In the example, the axial displacement, caused by centrifugal forces, has a nonlinear distribution, so that the first dynamic axial mode is required to achieve reasonable accuracy. Figure 4.8 shows that using only one linear static mode (FNL1 curve) yields unacceptable results, increasing the stiffness excessively. Therefore, two axial modes are at least needed to correctly simulate the motion of the beam (FNL2 curve). A more efficient and accu-

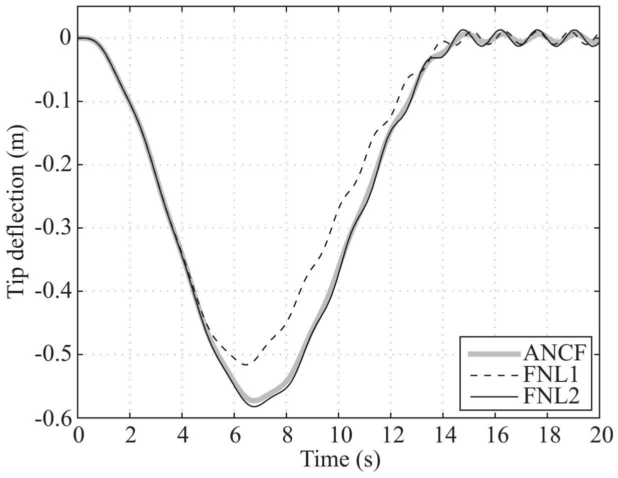


Figure 4.8: First nonlinear formulation vs. ANCF in the first example.

rate, although less general alternative to the use of a combination of static and dynamic axial modes, would be the use of a single *acceleration mode*. The idea is to introduce a deformation mode specifically designed for fitting the axial displacement field produced by a variable centrifugal acceleration, such as that produced by rotation.

The foreshortening formulation (FS curve in Figure 4.9) achieves the best results, despite the absence of axial modes. The quality of the correlation becomes more obvious at the steady–state stage, where the first nonlinear formulation shows a higher oscillation amplitude.

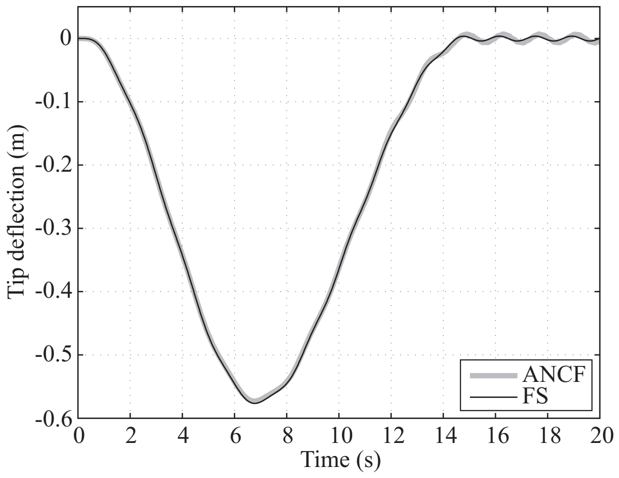


Figure 4.9: Foreshortening formulation vs. ANCF in the first example.

Table 4.1 shows the CPU–times for all the simulations, run with the same integrator and parameters, with a time–step of 0.01 s. The reference solution takes more than one hour to be computed, but this should not be taken as a reference for the performance of the ANCF, since the used formulation is one of the earliest ANCF developments and, moreover, it has been implemented in MATLAB. The FFR formulations are sorted according to their accuracy, from the lowest to the highest: first nonlinear with one axial mode (FNL1), first nonlinear with two axial modes (FNL2), and foreshortening (FS0). For a fixed number of modes, the foreshortening method is slower than the first nonlinear formulation, since it recalculates the mode shapes at every iteration; however, in order to achieve a similar accuracy, the first nonlinear formulation needs two additional axial modes, making it less efficient in practice than the foreshortening formulation.

Table 4.1: CPU–times (s) in the 2D spin–up beam.

| Formulation | FNL1 | FNL2 | FS0 |
|---|---|---|---|
| Absolute | 0.266 | 0.297 | 0.266 |
| Relative | 0.094 | 0.125 | 0.094 |

Moreover, it is observed that the performance is greatly increased when using relative coordinates. According to the results obtained in the second chapter, the use of relative coordinates should not be advantageous in such a small system, but in this particular case it is, due to the extremely simple topology. Since the system consists of only one body with one relative coordinate, the forward position and velocity analyses are trivial, and the same happens to the backward accumulation of forces and inertias, so that the reduction of the number of variables is completely taken advantage of.

## 4.5.2 Three–dimensional case

In this example, a gravity acceleration of 9.81 m/s$^2$ in the negative direction of the $z$ axis is added to the previously studied case, leading to a three–dimensional problem. The beam is simulated by using substructuring in relative coordinates, along with the same four methods compared in the two–dimensional case. In this example, a commercial nonlinear finite element code (COSMOS/M) is used for obtaining the reference solution. The reference model is discretized into 20 elements, and the numerical integration is carried out by using, as well as in the FFR formulations, the trapezoidal rule with a time–step of 10 ms. In order to obtain a numerical value for the error, the position of the beam tip (in the local frame) is compared with that of the reference solution. For the $x$ direction,

$$\Delta x = \frac{1}{n_s + 1} \sum_{i=0}^{n_s} \left| x_i - x_i^* \right| \tag{4.37}$$

where $x_i$ and $x_i^*$ are the calculated and reference values respectively, and $n_s$ is the number of time–steps. The same is done for $y$ and $z$ directions. In all the simulations, the beam is let reach its equilibrium position prior to starting the spin–up maneuver.

### Substructuring

In the substructuring model, each one of the $n$ substructures is discretized into two finite elements, being their elastic deformation approximated by four transversal bending modes. The first $n - 1$ substructures use the static modes defined by unit displacements along the local $y$ and $z$ directions, along with unit rotations about the same axes, whereas the last substructure, since it has a free end, is modeled by using the first four free–end dynamic modes.

The displacements of the tip of the beam in the local $x$, $y$ and $z$ directions are shown in Figures 4.10 and 4.11, for three, five and ten substructures. In this example, due to the large deflections, the use of the infinitesimal rotation matrices defined in Eq. (2.12) for the rotation static modes introduces a significant error in the solution, especially in the axial direction. If three consecutive orthogonal rotations are used, as suggested by Kim and Haug (1988), the accuracy is significantly improved, although this is not fully consistent with the definition of the deformation modes, since the rotations are defined about the undeformed axes. The CPU–times and mean errors with respect to the reference solution, obtained by using 3, 5, 10 and 20 substructures, are shown in Table 4.2. The last column (20b) shows the results obtained if infinitesimal rotation matrices are used. The finite element model takes 19 seconds with 20 ele-

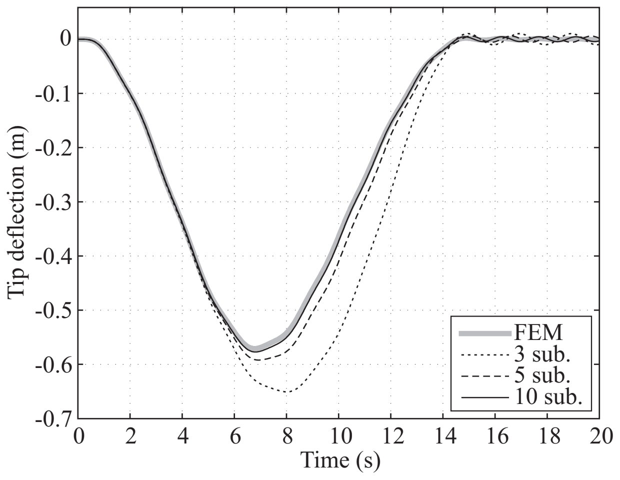


Figure 4.10: Substructuring vs. FEM in the second example ($y$).

ments, although it can be lowered to 14 seconds if only 10 elements are used, with no significant loss of accuracy.

Table 4.2: 3D spin–up beam results (substructuring).

| Substructures | 3 | 5 | 10 | 20 | 20b |
|---|---|---|---|---|---|
| CPU–time (s) | 0.375 | 0.609 | 1.312 | 4.578 | 4.562 |
| $\Delta x$ (mm) | 3.734 | 1.337 | 0.502 | 0.320 | 3.061 |
| $\Delta y$ (mm) | 47.681 | 13.128 | 3.193 | 1.481 | 1.757 |
| $\Delta z$ (mm) | 38.458 | 11.682 | 2.329 | 0.742 | 0.920 |

**First nonlinear formulation**

The FNL formulation does not need any modification for being extended to the 3D case. This formulation, whose results are shown in Figures 4.12 and 4.13, does not obtain acceptable results for the axial displacement, no matter how many axial modes are introduced, since the large deflection makes the foreshortening effect much more relevant than the actual beam shortening. At the equilibrium position, there is a tip displacement of more than 6 cm in the $x$ direction, which is not captured, and the error at the steady–state stage is about 3.5 mm with respect to the reference. It is observed that the use of free–end dynamic modes leads to better results than the combination of a static and a fixed–interface dynamic mode used in the first example. This makes sense because in this particular case the beam has a free end, so that this combination of modes represents better the actual reference conditions of the beam. However, two axial modes are still needed to reach a good accuracy in the $y$ direction. The CPU–times and mean errors obtained when using the first nonlinear formulation with both absolute and relative coordinates are shown in Table 4.3.

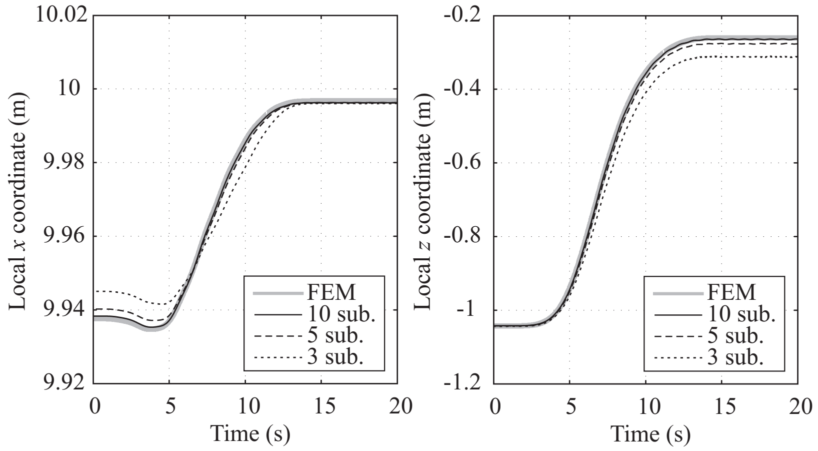


Figure 4.11: Substructuring vs. FEM in the second example ($x$ and $z$).

Table 4.3: 3D spin–up beam results (first nonlinear formulation).

| Formulation | Absolute | | Relative | |
|---|---|---|---|---|
| | FNL1 | FNL2 | FNL1 | FNL2 |
| CPU–time (s) | 0.271 | 0.286 | 0.105 | 0.125 |
| $\Delta x$ (mm) | 27.943 | 27.946 | 27.944 | 27.947 |
| $\Delta y$ (mm) | 11.046 | 5.937 | 10.313 | 5.576 |
| $\Delta z$ (mm) | 7.709 | 3.844 | 7.502 | 3.907 |

**Foreshortening formulation**

The extension of the FS formulation to the three–dimensional case is straightforward, since the effects in $y$ and $z$ directions can be considered independent. The foreshortening can be obtained from the following expression, where $w_0$ is the neutral axis displacement in the $z$ direction (Shi et al., 2001; Valembois et al., 1997),

$$u_{fs}(x) = -\frac{1}{2}\int_{x_0}^{x} \left( {v_0'}^2 + {w_0'}^2 \right) dx \tag{4.38}$$

In the case of the foreshortening method, as can be seen in Figures 4.14 and 4.15, the precision in the $x$ direction is significantly improved. In the vertical direction, the results are approximately the same obtained with the FNL2 formulation, despite using fewer deformation modes.

In Table 4.4 the CPU–times and deviations from the reference solution are shown for the foreshortening formulation. The results for none (FS0), one (FS1), and two (FS2) axial modes are included, in order to compare the efficiency to that of the first nonlinear formulation, although the plotted results correspond to the model with no axial modes. As it happened in the two–dimensional case, the FNL formulation is faster for the same number of modes, but when the efficiency is compared for a similar

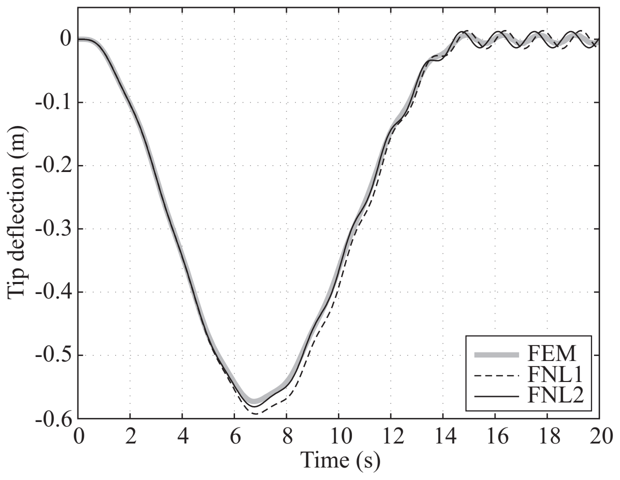


Figure 4.12: First nonlinear formulation vs. FEM in the second example ($y$).

level of accuracy in $y$ and $z$ directions, the FS method is slightly faster (i.e. FNL2 vs. FS0).

Table 4.4: 3D spin–up beam results (foreshortening formulation).

| Formulation | Absolute | | | Relative | | |
|---|---|---|---|---|---|---|
| | FS0 | FS1 | FS2 | FS0 | FS1 | FS2 |
| CPU–time (s) | 0.250 | 0.292 | 0.328 | 0.105 | 0.125 | 0.125 |
| $\Delta x$ (mm) | 0.503 | 0.360 | 0.363 | 0.496 | 0.363 | 0.366 |
| $\Delta y$ (mm) | 3.128 | 3.116 | 3.116 | 3.152 | 3.149 | 3.149 |
| $\Delta z$ (mm) | 4.780 | 4.786 | 4.786 | 4.811 | 4.818 | 4.818 |

## 4.6 Conclusions and criteria of use

In the present chapter, several methods for capturing the geometric stiffening effect in FFR formulations have been successfully implemented and compared. As the results obtained for the Kane's beam demonstrate, the FFR formulation with linearized elastic forces cannot capture this effect, at least if the beam is modeled as one single flexible body. If the beam is divided into smaller substructures, the effect is accurately captured and, if relative coordinates are used, the performance is very good if compared to ANCF or nonlinear FEM.

Among the methods that introduce modifications to the formulation, the first nonlinear formulation is the easiest to implement, obtaining very good results in a fraction of the time required when using substructures. However, it presents some problems, since only one axial mode is not sufficient to obtain accurate results, and the use of axial modes of high natural frequencies hinders the integration process. Moreover, if

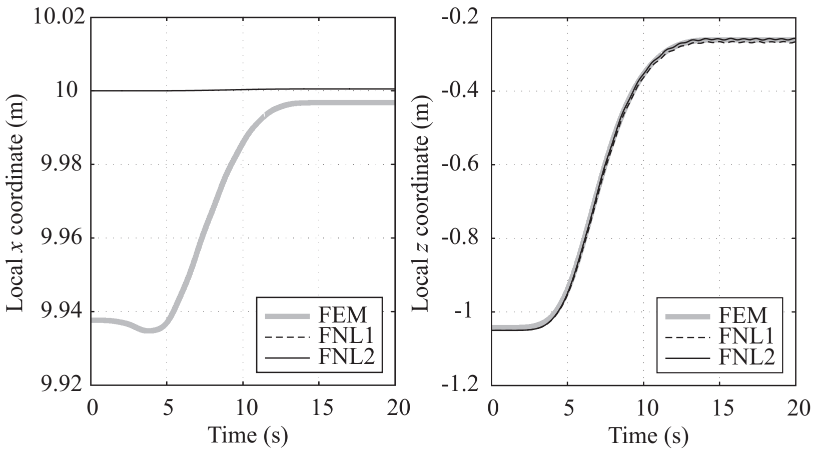


Figure 4.13: First nonlinear formulation vs. FEM in the second example ($x$ and $z$).

the deflections are large enough, this method fails also to accurately measure the axial displacements.

Finally, the foreshortening formulation has proven to have better accuracy and efficiency than the first nonlinear formulation, and almost the same accuracy as substructuring. It does not require axial modes for obtaining good results for the transversal deflections and, in case that axial stresses are needed, longitudinal modes can be added without problems. For the same number of modes, it is slightly slower since it involves more operations, but if the efficiency/accuracy ratio is considered, it is always advantageous to include the foreshortening in the kinematic modeling.

The use of relative coordinates dramatically improves the efficiency in the examples here addressed, despite being small systems. On the one hand, there is only one relative reference coordinate, namely the rotated angle, whereas the formulation in natural coordinates requires 12 additional variables for modeling the frame of reference. On the other hand, the time–consuming forward loops needed for calculating the positions and velocities, along with the backward projection of the mass matrix and the inertia forces vector, are completely trivial in this case.

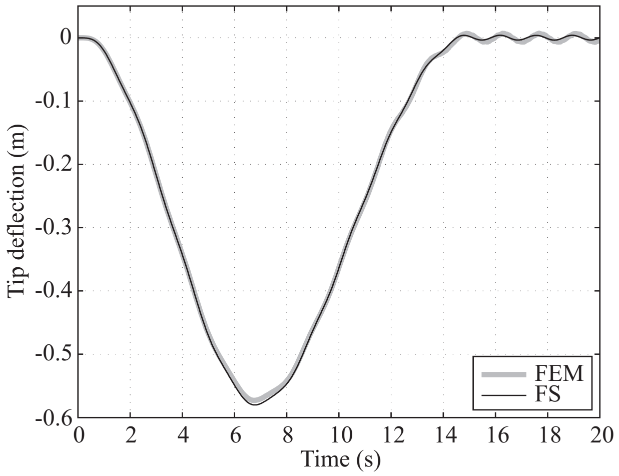


Figure 4.14: Foreshortening formulation vs. FEM in the second example ($y$).

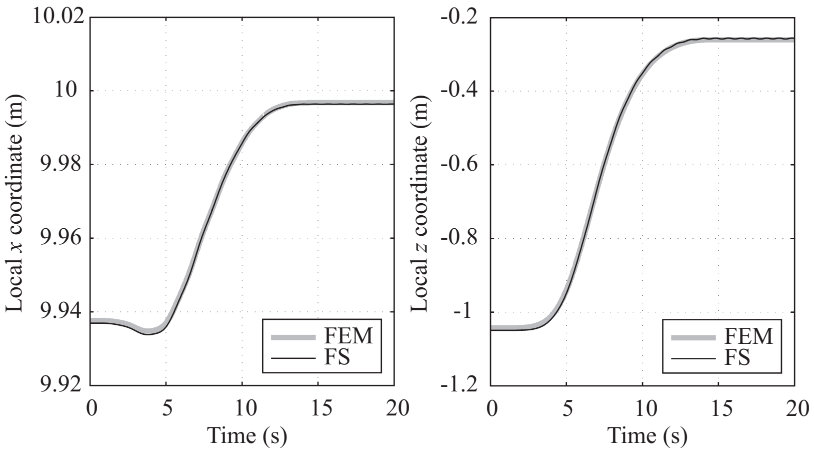


Figure 4.15: Foreshortening formulation vs. FEM in the second example ($x$ and $z$).

Chapter 5

# Conclusions and Future Research

## 5.1 Conclusions

All the different methods presented in this thesis are aimed at achieving real–time performance in the simulation of flexible multibody systems. The results obtained, including a full car simulated ten times faster than real–time, show that the objectives have been successfully achieved.

The main conclusions that can be extracted from this thesis are the following:

- A new semi–recursive $O\left(n^3\right)$ formulation in dependent relative coordinates for flexible multibody dynamics has been successfully implemented. The method has been used for the simulation of three different systems, including a full model of a car with twelve flexible elements, and compared in terms of efficiency and robustness to a method in natural coordinates which uses the same flexible body modeling. All the tested systems are closed–loop mechanisms, and two of them are three–dimensional problems performing violent maneuvers that are integrated without problems, even with very large time–steps, which demonstrates the robustness of the formulation.

- The method in relative coordinates, as it happens in the rigid case, presents a higher efficiency than its counterpart in natural coordinates when simulating large systems such as the Iltis vehicle, achieving a 500% performance improvement, something that can be determinant for real–time applications. The performance is also increased for medium–size problems, e.g. the single Iltis suspension, although not to the same extent (150% to 200%), so that in certain applications the performance increase may not be worth the extra implementation effort. In the case of very small systems, the method in relative coordinates is slower, since the reduction in the number of variables does not compensate the additional computational burden. However, it should be pointed out that the double four–bar example has been implemented in MATLAB, so that the results obtained are not concluding; moreover, in some specific cases, such as the Kane's beam studied in Chapter 4, the use of relative coordinates improves the performance despite being a very small system, due to its extremely simple topology.

As a rule of thumb, it can be said that the method in relative coordinates can improve performance in problems that, in the rigid case, have more than 25 variables when using natural coordinates, although significant improvement is to be expected only for large systems, above 50 variables.

- A further efficiency improvement has been achieved by implementing the calculation of the inertia terms by means of the inertia shape integrals. The method has been introduced into the new formulation, as well as into the original formulation in natural coordinates, leading to a very compact and systematic implementation. The use of the inertia shape integrals improves performance even for very small finite element models and, since it completely eliminates the mesh size from the problem, enables to use models of any size.

  However, the projection of the finite element mass matrix, used in the original method in natural coordinates, is much simpler to implement, and reasonably efficient for small finite element models, making it more recommendable for certain applications.

- In order to extend the range of applicability of FFR methods, three different methods for capturing the geometric stiffening effect in beams have been also implemented and compared. One of them, the substructuring technique, has been only tested in relative coordinates, since the number of variables otherwise required makes it less competitive for real–time applications. The remaining two methods have been implemented in both absolute and relative coordinates.

  The most accurate approach, at least regarding the transversal deflections, is the use of the substructuring technique. It enables to capture the nonlinear effects without introducing any modification to the formulation, at the cost of increasing the number of variables, a problem that can be minimized by using relative coordinates. By using substructures, the simulation of the Kane's beam can be performed about ten times faster than by using a commercial nonlinear finite–element code, with equivalent results.

  Among the methods that introduce modifications in the formulation in order to capture the geometric stiffening effect, the use of a nonlinear stiffness matrix is the simplest and most straightforward one, obtaining very good results if its extreme simplicity is taken into account, although the requirement of introducing axial modes may be a problem due to their high stiffness.

  The introduction of the axial foreshortening at the modeling stage yields the most accurate results in the axial direction and, although the accuracy in the transverse directions is slightly lower than that obtained by using substructures, the achieved real–time ratio is about 200, 15 times faster than substructuring and 150 times faster than nonlinear finite elements, making it the most adequate approach for real–time applications.

## 5.2 Future Research

The work presented in this thesis can be further developed in several directions, such as efficiency improvements or different applications.

- In order to further improve the efficiency, different methods for the reduction of the finite element model can be explored. In Koutsovasilis and Beitelschmidt (2008), a review of different existing methods is presented, being the Krylov subspaces (Lehner and Eberhard, 2006) a promising alternative to the classical static and dynamic modes.

- Another important issue that must be addressed is the optimal selection of the mode shapes. The choice of the mode shapes is in general left to the criterion of the analyst, and it is difficult to establish automated methods for that critical task. Several efforts have been carried out in that direction, such as the calculation of modal participation factors from the results of a preliminary rigid body simulation (Wallrapp and Wiedemann, 2002).

- The inertia terms obtained by using the preprocessing method can be analyzed in detail, in order to identify terms that can be neglected, thus reducing the number of operations. For instance, some of the terms are quadratic in the modal amplitudes, and since the deformations are asumed to be small, they can be probably neglected without any significant effect on the accuracy.

# Appendix

In this Appendix, the resulting matrices obtained for the example mechanism shown in Chapter 2, after the projection of the mass matrix into the relative coordinates are fully developed. The mechanism is shown in Figure 2.11, which is here repeated.

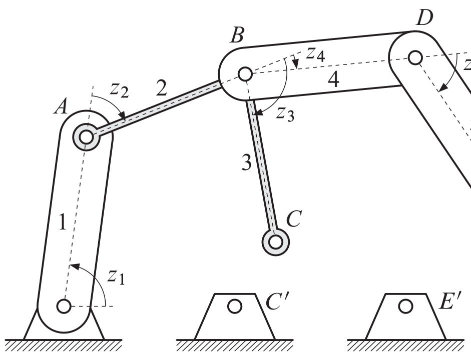


Figure A.1: Example mechanism of Chapter 2.

The mechanism has two flexible bodies, namely bodies 2 and 3. The first one has an input boundary at point $A$, and its output at point $B$. Point $B$ is in turn the input point of body 3, which has also an output boundary at point $C$. There exist two flexible bodies and four boundaries, so that there will be four sets of static modes, and two sets of dynamic modes. The specific sets of Cartesian and relative coordinates are, if both of them are sorted in such a way that reference coordinates, static modal amplitudes and dynamic modal amplitudes, are grouped,

$$\begin{aligned}\mathbf{Z} &= \left\{\mathbf{Z}_{r1}^{\top} \quad \mathbf{Z}_{r2}^{\top} \quad \mathbf{Z}_{r3}^{\top} \quad \mathbf{Z}_{r4}^{\top} \quad \mathbf{Z}_{r5}^{\top} \quad \dot{\boldsymbol{\eta}}_2^{A\top} \quad \dot{\boldsymbol{\eta}}_2^{B\top} \quad \dot{\boldsymbol{\eta}}_3^{B\top} \quad \dot{\boldsymbol{\eta}}_3^{C\top} \quad \dot{\boldsymbol{\xi}}_2^{\top} \quad \dot{\boldsymbol{\xi}}_3^{\top}\right\}^{\top} \\ \dot{\mathbf{z}} &= \left\{\dot{z}_1^{\top} \quad \dot{z}_2^{\top} \quad \dot{z}_3^{\top} \quad \dot{z}_4^{\top} \quad \dot{z}_5^{\top} \quad \dot{\boldsymbol{\eta}}_2^{A\top} \quad \dot{\boldsymbol{\eta}}_2^{B\top} \quad \dot{\boldsymbol{\eta}}_3^{B\top} \quad \dot{\boldsymbol{\eta}}_3^{C\top} \quad \dot{\boldsymbol{\xi}}_2^{\top} \quad \dot{\boldsymbol{\xi}}_3^{\top}\right\}^{\top}\end{aligned} \tag{A.1}$$

As pointed out in the second chapter, the mass matrix of each flexible body has the following structure:

$$\bar{\mathbf{M}}_i = \begin{bmatrix} \bar{\mathbf{M}}_{ri} & \bar{\mathbf{M}}_{r\eta i} & \bar{\mathbf{M}}_{r\xi i} \\ & \bar{\mathbf{M}}_{\eta i} & \bar{\mathbf{M}}_{\eta\xi i} \\ sym. & & \bar{\mathbf{M}}_{\xi i} \end{bmatrix} \tag{A.2}$$

If the mass matrices of the five bodies are assembled into a full matrix, the mass matrix of the mechanism in the Cartesian set of coordinates **Z** results, taking into account that the rigid bodies will only have a $\bar{\mathbf{M}}_r$ submatrix:

$$\bar{\mathbf{M}} = \left[\begin{array}{ccccc|cccc|cc}
\bar{\mathbf{M}}_{r1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & \bar{\mathbf{M}}_{r2} & 0 & 0 & 0 & \bar{\mathbf{M}}^{A}_{r\eta 2} & \bar{\mathbf{M}}^{B}_{r\eta 2} & 0 & 0 & \bar{\mathbf{M}}_{r\xi 2} & 0 \\
 & & \bar{\mathbf{M}}_{r3} & 0 & 0 & 0 & 0 & \bar{\mathbf{M}}^{B}_{r\eta 3} & \bar{\mathbf{M}}^{C}_{r\eta 3} & 0 & \bar{\mathbf{M}}_{r\xi 3} \\
 & & & \bar{\mathbf{M}}_{r4} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & & & & \bar{\mathbf{M}}_{r5} & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
 & & & & & \bar{\mathbf{M}}^{A}_{\eta 2} & \bar{\mathbf{M}}^{AB}_{\eta 2} & 0 & 0 & \bar{\mathbf{M}}^{A}_{\eta\xi 2} & 0 \\
 & & & & & & \bar{\mathbf{M}}^{B}_{\eta 2} & 0 & 0 & \bar{\mathbf{M}}^{B}_{\eta\xi 2} & 0 \\
 & & & & & & & \bar{\mathbf{M}}^{B}_{\eta 3} & \bar{\mathbf{M}}^{BC}_{\eta 3} & 0 & \bar{\mathbf{M}}^{B}_{\eta\xi 3} \\
 & & sym. & & & & & & \bar{\mathbf{M}}^{C}_{\eta 3} & 0 & \bar{\mathbf{M}}^{C}_{\eta\xi 3} \\
\hline
 & & & & & & & & & \bar{\mathbf{M}}_{\xi 2} & 0 \\
 & & & & & & & & & & \bar{\mathbf{M}}_{\xi 3}
\end{array}\right] \tag{A.3}$$

The assembly of the **R** matrix, according to Eqs. (2.133), (2.134) and (2.135), would be

$$\mathbf{R} = \left[\begin{array}{ccccc|cccc|cc}
\mathbf{b}_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\mathbf{b}_1 & \mathbf{b}_2 & 0 & 0 & 0 & \varphi_2^A & 0 & 0 & 0 & 0 & 0 \\
\mathbf{b}_1 & \mathbf{b}_2 & \mathbf{b}_3 & 0 & 0 & \varphi_2^A & \varphi_2^B & \varphi_3^B & 0 & 0 & 0 \\
\mathbf{b}_1 & \mathbf{b}_2 & 0 & \mathbf{b}_4 & 0 & \varphi_2^A & \varphi_2^B & 0 & 0 & 0 & 0 \\
\mathbf{b}_1 & \mathbf{b}_2 & 0 & \mathbf{b}_4 & \mathbf{b}_5 & \varphi_2^A & \varphi_2^B & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & \mathbf{I} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \mathbf{I} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{I} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{I} & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{I} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{I}
\end{array}\right] \tag{A.4}$$

As pointed out in Chapter 2, these two matrices are never actually assembled, but their terms are directly used to obtain the following results for the different terms of the mass matrix. The terms appearing in the projections of the rigid body or reference mass matrices, shown in Eq. (2.142), result as follows:

$$\mathbf{R}_r^{\mathsf{T}}\bar{\mathbf{M}}_r\mathbf{R}_r = \begin{bmatrix} \mathbf{b}_1^{\mathsf{T}}\mathbf{M}_{r1}\mathbf{b}_1 & \mathbf{b}_1^{\mathsf{T}}\mathbf{M}_{r2}\mathbf{b}_2 & \mathbf{b}_1^{\mathsf{T}}\mathbf{M}_{r3}\mathbf{b}_3 & \mathbf{b}_1^{\mathsf{T}}\mathbf{M}_{r4}\mathbf{b}_4 & \mathbf{b}_1^{\mathsf{T}}\mathbf{M}_{r5}\mathbf{b}_5 \\ & \mathbf{b}_2^{\mathsf{T}}\mathbf{M}_{r2}\mathbf{b}_2 & \mathbf{b}_2^{\mathsf{T}}\mathbf{M}_{r3}\mathbf{b}_3 & \mathbf{b}_2^{\mathsf{T}}\mathbf{M}_{r4}\mathbf{b}_4 & \mathbf{b}_2^{\mathsf{T}}\mathbf{M}_{r5}\mathbf{b}_5 \\ & & \mathbf{b}_3^{\mathsf{T}}\mathbf{M}_{r3}\mathbf{b}_3 & \mathbf{b}_3^{\mathsf{T}}\mathbf{M}_{r4}\mathbf{b}_4 & 0 \\ & sym. & & \mathbf{b}_4^{\mathsf{T}}\mathbf{M}_{r4}\mathbf{b}_4 & \mathbf{b}_4^{\mathsf{T}}\mathbf{M}_{r5}\mathbf{b}_5 \\ & & & & \mathbf{b}_5^{\mathsf{T}}\mathbf{M}_{r5}\mathbf{b}_5 \end{bmatrix} \tag{A.5}$$

$$\mathbf{R}_\eta^{\mathsf{T}}\bar{\mathbf{M}}_r\mathbf{R}_\eta = \begin{bmatrix} \varphi_2^{A\mathsf{T}}\mathbf{M}_{r2}^{A}\varphi_2^{A} & \varphi_2^{A\mathsf{T}}\mathbf{M}_{r2}^{B}\varphi_2^{B} & \varphi_2^{A\mathsf{T}}\mathbf{M}_{r3}^{B}\varphi_3^{B} & 0 \\ & \varphi_2^{B\mathsf{T}}\mathbf{M}_{r2}^{B}\varphi_2^{B} & \varphi_2^{B\mathsf{T}}\mathbf{M}_{r3}^{B}\varphi_3^{B} & 0 \\ & & \varphi_3^{B\mathsf{T}}\mathbf{M}_{r3}^{B}\varphi_3^{B} & 0 \\ sym. & & & 0 \end{bmatrix} \tag{A.6}$$

$$\mathbf{R}_r^{\mathsf{T}}\bar{\mathbf{M}}_r\mathbf{R}_\eta = \begin{bmatrix} \mathbf{b}_1^{\mathsf{T}}\mathbf{M}_{r2}^{A}\varphi_2^{A} & \mathbf{b}_1^{\mathsf{T}}\mathbf{M}_{r2}^{B}\varphi_2^{B} & \mathbf{b}_1^{\mathsf{T}}\mathbf{M}_{r3}^{B}\varphi_3^{B} & 0 \\ \mathbf{b}_2^{\mathsf{T}}\mathbf{M}_{r2}\varphi_2^{A} & \mathbf{b}_2^{\mathsf{T}}\mathbf{M}_{r2}^{B}\varphi_2^{B} & \mathbf{b}_2^{\mathsf{T}}\mathbf{M}_{r3}^{B}\varphi_3^{B} & 0 \\ \mathbf{b}_3^{\mathsf{T}}\mathbf{M}_{r3}\varphi_2^{A} & \mathbf{b}_3^{\mathsf{T}}\mathbf{M}_{r3}\varphi_2^{B} & \mathbf{b}_3^{\mathsf{T}}\mathbf{M}_{r3}^{B}\varphi_3^{B} & 0 \\ \mathbf{b}_4^{\mathsf{T}}\mathbf{M}_{r4}\varphi_2^{A} & \mathbf{b}_4^{\mathsf{T}}\mathbf{M}_{r4}\varphi_2^{B} & 0 & 0 \\ \mathbf{b}_5^{\mathsf{T}}\mathbf{M}_{r5}\varphi_2^{A} & \mathbf{b}_5^{\mathsf{T}}\mathbf{M}_{r5}\varphi_2^{B} & 0 & 0 \end{bmatrix} \tag{A.7}$$

where the accumulated mass matrices $\mathbf{M}_{ri}$ and $\mathbf{M}_{ri}^{P}$ are those defined in Eqs. (2.140) and (2.141). The four different terms appearing in Eq. (2.143) result

$$\mathbf{R}_r^{\mathsf{T}}\bar{\mathbf{M}}_{r\eta} = \begin{bmatrix} \mathbf{b}_1^{\mathsf{T}}\bar{\mathbf{M}}_{r\eta 2} & \mathbf{b}_1^{\mathsf{T}}\bar{\mathbf{M}}_{r\eta 3} \\ \mathbf{b}_2^{\mathsf{T}}\bar{\mathbf{M}}_{r\eta 2} & \mathbf{b}_2^{\mathsf{T}}\bar{\mathbf{M}}_{r\eta 3} \\ 0 & \mathbf{b}_3^{\mathsf{T}}\bar{\mathbf{M}}_{r\eta 3} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} ; \qquad \mathbf{R}_r^{\mathsf{T}}\bar{\mathbf{M}}_{r\xi} = \begin{bmatrix} \mathbf{b}_1^{\mathsf{T}}\bar{\mathbf{M}}_{r\xi 2} & \mathbf{b}_1^{\mathsf{T}}\bar{\mathbf{M}}_{r\xi 3} \\ \mathbf{b}_2^{\mathsf{T}}\bar{\mathbf{M}}_{r\xi 2} & \mathbf{b}_2^{\mathsf{T}}\bar{\mathbf{M}}_{r\xi 3} \\ 0 & \mathbf{b}_3^{\mathsf{T}}\bar{\mathbf{M}}_{r\xi 3} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{A.8}$$

$$\mathbf{R}_\eta^{\mathsf{T}}\bar{\mathbf{M}}_{r\eta} = \begin{bmatrix} \varphi_2^{A\mathsf{T}}\bar{\mathbf{M}}_{r\eta 2} & \varphi_2^{A\mathsf{T}}\bar{\mathbf{M}}_{r\eta 3} \\ \varphi_2^{B\mathsf{T}}\bar{\mathbf{M}}_{r\eta 2} & \varphi_2^{B\mathsf{T}}\bar{\mathbf{M}}_{r\eta 3} \\ 0 & \varphi_3^{B\mathsf{T}}\bar{\mathbf{M}}_{r\eta 3} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} ; \qquad \mathbf{R}_\eta^{\mathsf{T}}\bar{\mathbf{M}}_{r\xi} = \begin{bmatrix} \varphi_2^{A\mathsf{T}}\bar{\mathbf{M}}_{r\xi 2} & \varphi_2^{A\mathsf{T}}\bar{\mathbf{M}}_{r\xi 3} \\ \varphi_2^{B\mathsf{T}}\bar{\mathbf{M}}_{r\xi 2} & \varphi_2^{B\mathsf{T}}\bar{\mathbf{M}}_{r\xi 3} \\ 0 & \varphi_3^{B\mathsf{T}}\bar{\mathbf{M}}_{r\xi 3} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{A.9}$$

where the mass blocks in Cartesian coordinates $\bar{\mathbf{M}}_{r\eta i}$ and $\bar{\mathbf{M}}_{r\xi i}$ are considered as a whole for each flexible body, e.g.

$$\bar{\mathbf{M}}_{r\eta 2} = \begin{bmatrix} \bar{\mathbf{M}}_{r\eta 2}^{A} & \bar{\mathbf{M}}_{r\eta 2}^{B} \end{bmatrix} \tag{A.10}$$

# Bibliography

O. P. Agrawal and A. A. Shabana. Dynamic analysis of multibody systems using component modes. *Computers & Structures*, 21(6):1303–1312, 1985.

O. P. Agrawal and A. A. Shabana. Application of deformable–body mean axis to flexible multibody system dynamics. *Computer Methods in Applied Mechanics and Engineering*, 56:217–245, 1986.

F. M. L. Amirouche. *Computational Methods in Flexible Multibody Dynamics*. Prentice Hall, Englewood Cliffs, NJ, 1992.

A. Avello. *Dinámica de mecanismos flexibles con coordenadas Cartesianas y teoría de grandes deformaciones*. PhD thesis, University of Navarra, San Sebastián, Spain, 1990.

A. Avello. *Simulación dinámica interactiva de mecanismos flexibles con pequeñas deformaciones*. PhD thesis, University of Navarra, San Sebastián, Spain, 1995.

A. Avello, J. García de Jalón, and E. Bayo. Dynamics of flexible multibody systems using Cartesian co–ordinates and large displacement theory. *International Journal for Numerical Methods in Engineering*, 32(8):1543–1563, 1991.

D. S. Bae. Development of a new Multi-Flexible Body Dynamics (MFBD) platform: A relative nodal displacement method for finite element analysis. In *Proceedings of the ASME 2005 IDECT/CIE*, volume 6 C, pages 1821–1829, Long Beach, CA, 2005.

D. S. Bae and E. J. Haug. A recursive formulation for constrained mechanical system dynamics: Part I. Open loop systems. *Mechanics of Structures and Machines*, 15 (3):359–382, 1987.

D. S. Bae and E. J. Haug. A recursive formulation for constrained mechanical system dynamics: Part II. Closed loop systems. *Mechanics of Structures and Machines*, 15 (4):481–506, 1988.

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*, 50(8):1841–1859, 2001.

A. K. Banerjee and S. Nagarajan. Efficient simulation of large overall motion of beams undergoing large deflection. *Multibody System Dynamics*, 1(1):113–126, 1997.

K. J. Bathe. *Finite Element Procedures*. Prentice Hall, Englewood Cliffs, NJ, 1995.

J. Baumgarte. Stabilization of constraints and integrals of motion in dynamical systems. *Computer Methods in Applied Mechanics and Engineering*, 1(1):1–16, 1972.

E. Bayo and R. Ledesma. Augmented Lagrangian and mass–orthogonal projection methods for constrained multibody dynamics. *Nonlinear Dynamics*, 9(1–2):113–130, 1996.

E. Bayo, J. García de Jalón, and M. A. Serna. A modified Lagrangian formulation for the dynamic analysis of constrained mechanical systems. *Computer Methods in Applied Mechanics and Engineering*, 71(2):183–195, 1988.

T. Belytschko and B. J. Hsieh. Non–linear transient finite element analysis with convected co–ordinates. *International Journal for Numerical Methods in Engineering*, 7(3):255–271, 1973.

W. J. Book. A recursive Lagrangian formulation of manipulator dynamics and comparative study of dynamics formulation complexity. *International Journal of Robotics Research*, 3:87–101, 1984.

H. Bremer. On the dynamics of elastic multibody systems. *Applied Mechanics Reviews*, 52(9):275–296, 1999.

O. Brüls, P. Duysinx, and J. C. Golinval. The global modal parameterization for non–linear model–order reduction in flexible multibody dynamics. *International Journal for Numerical Methods in Engineering*, 69(5):948–977, 2007.

A. Cardona and M. Géradin. A beam finite element non–linear theory with finite rotations. *International Journal for Numerical Methods in Engineering*, 26:2403–2438, 1988.

A. Cardona and M. Géradin. Modeling of superelements in mechanism analysis. *International Journal for Numerical Methods in Engineering*, 32(8):1565–1593, 1991.

R. R. Craig and M. C. C. Bampton. Coupling of substructures for dynamic analyses. *AIAA Journal*, 6(7):1313–1319, 1968.

M. A. Crisfield. *Non–Linear Finite Element Analysis of Solids and Structures, Vol. 1 & 2*. John Wiley & Sons Inc., New York, NY, 1997.

J. Cuadrado. *Una nueva formulación en coordenadas naturales para el estudio de la flexibilidad en los mecanismos*. PhD thesis, University of Navarra, San Sebastián, Spain, 1993.

J. Cuadrado, J. Cardenal, and J. García de Jalón. Flexible mechanisms through natural coordinates and component synthesis: An approach fully compatible with the rigid case. *International Journal for Numerical Methods in Engineering*, 39(20):3535–3551, 1996.

J. Cuadrado, J. Cardenal, and E. Bayo. Modeling and solution methods for efficient real–time simulation of multibody dynamics. *Multibody System Dynamics*, 1(3): 259–280, 1997.

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*, 4(1):55–73, 2000.

J. Cuadrado, R. Gutiérrez, M. A. 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*, 51(9):1033–1052, 2001.

J. Cuadrado, D. Dopico, M. González, and M. A. Naya. A combined penalty and recursive real–time formulation for multibody dynamics. *Journal of Mechanical Design*, 126(4):602–608, 2004a.

J. Cuadrado, D. Dopico, M. A. Naya, and M. González. Penalty, semi–recursive and hybrid methods for MBS real–time dynamics in the context of structural integrators. *Multibody System Dynamics*, 12(2):117–132, 2004b.

J. Cuadrado, R. Gutiérrez, M. A. 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*, 11(2):147–166, 2004c.

O. N. Dmitrochenko and D. Y. Pogorelov. Generalization of plate finite elements for absolute nodal coordinate formulation. *Multibody System Dynamics*, 10(1):17–43, 2003.

D. Dopico. *Formulaciones semi–recursivas y de penalización para la dinámica en tiempo real de sistemas multicuerpo*. PhD thesis, University of La Coruña, Ferrol, Spain, 2004.

K. E. Dufva, J. T. Sopanen, and A. M. Mikkola. A two–dimensional shear deformable beam element based on the absolute nodal coordinate formulation. *Journal of Sound and Vibration*, 280(3–5):719–738, 2005.

A. G. Erdman and G. N. Sandor. Kineto–elastodynamics – A review of the state of the art and trends. *Mechanism and Machine Theory*, 7:19–33, 1972.

J. L. Escalona, H. A. Hussien, and A. A. Shabana. Application of the absolute nodal coordinate formulation to multibody system dynamics. *Journal of Sound and Vibration*, 214(5):833–851, 1998.

J. L. Escalona, J. M. Mayo, and J. Domínguez. Influence of reference conditions on the analysis of impact–induced elastic waves. *Multibody System Dynamics*, 7(2): 209–228, 2002.

R. Featherstone. Calculation of robot dynamics using articulated–body inertias. *International Journal of Robotics Research*, 2(1):13–30, 1983.

R. Featherstone. *Robot Dynamics Algorithm*. Kluwer Academic Publishers, Norwell, MA, 1987.

S. Frik, G. Leister, and W. Schwartz. Simulation of the IAVSD road vehicle benchmark bombardier Iltis with FASIM, MEDYNA, NEWEUL and SIMPACK. In *Multibody Computer Codes in Vehicle System Dynamics*, Amsterdam, 1993. Swets and Zeitlinger.

F. J. Funes, J. García de Jalón, F. de Ribera, and E. Álvarez. Solución de la dinámica de sistemas flexibles mediante formulaciones topológicas. In *Métodos Computacionais em Engenharia*, Lisbon, Portugal, 2004. APMTAC.

J. García de Jalón and E. Bayo. *Kinematic and Dynamic Simulation of Multibody Systems: The Real–Time Challenge*. Springer–Verlag, Berlin, 1994.

D. García-Vallejo. *Dinámica de sistemas multicuerpo rígido–flexibles en coordenadas absolutas*. PhD thesis, University of Sevilla, Sevilla, Spain, 2006.

D. García-Vallejo, J. L. Escalona, J. Mayo, and J. Domínguez. Describing rigid–flexible multibody systems using absolute coordinates. *Nonlinear Dynamics*, 34 (1-2):75–94, 2003.

M. Géradin and A. Cardona. *Flexible Multibody Dynamics: A Finite Element Approach*. John Wiley & Sons, New York, NY, 2001.

H. Goldstein. *Classical Mechanics*. Addison Wesley, Cambridge, MA, USA, 1950.

R. Gutiérrez. *Cálculo de tensiones en componentes de sistemas móviles mediante dinámica de sistemas multicuerpo flexibles*. PhD thesis, University of La Coruña, Ferrol, Spain, 2003.

R. S. Hartenberg and J. Denavit. *Kinematic Synthesis of Linkages*. McGraw–Hill, 1963.

B. He, X. Rui, and G. Wang. Riccati discrete time transfer matrix method for elastic beam undergoing large overall motion. *Multibody System Dynamics*, 18(4):579–598, 2007.

W. C. Hurty. Dynamic analysis of structural systems using component modes. *AIAA Journal*, 3(4):678–685, 1965.

R. L. Huston. Multi–body dynamics including the effect of flexibility and compliance. *Computers & Structures*, 14:443–451, 1981.

R. L. Huston. Computer methods in flexible multibody dynamics. *International Journal for Numerical Methods in Engineering*, 32(8):1657–1668, 1991.

J. M. Jiménez. *Formulaciones cinemáticas y dinámicas para la simulación en tiempo real de sistemas de sólidos rígidos*. PhD thesis, University of Navarra, San Sebastián, Spain, 1993.

B. Jonker. A finite element dynamic analysis of spatial mechanisms with flexible links. *Computer Methods in Applied Mechanics and Engineering*, 76:17–40, 1989.

T. Kane, R. Ryan, and A. Banerjee. Dynamics of a cantilever beam attached to a moving base. *AIAA Journal*, 10(2):131–151, 1987.

S. S. Kim and E. J. Haug. Recursive formulation for flexible multibody dynamics, part I: Open–loop systems. *Computer Methods in Applied Mechanics and Engineering*, 71(3):293–314, 1988.

S. S. Kim and E. J. Haug. Recursive formulation for flexible multibody dynamics, part II: Closed loop systems. *Computer Methods in Applied Mechanics and Engineering*, 74(3):251–269, 1989.

P. Koutsovasilis and M. Beitelschmidt. Comparison of model reduction techniques for large mechanical systems. *Multibody System Dynamics*, 20(2):111–128, 2008.

M. Lehner and P. Eberhard. On the use of moment–matching to build reduced order models in flexible multibody dynamics. *Multibody System Dynamics*, 16(2):191–211, 2006.

M. Lehner and P. Eberhard. A two–step approach for model reduction in flexible multibody dynamics. *Multibody System Dynamics*, 17(2–3):157–176, 2007.

G. G. Lowen and C. Chassapis. The elastic behavior of linkages: An update. *Mechanism and Machine Theory*, 21(1):33–42, 1986.

G. G. Lowen and W. G. Jandrasits. Survey of investigations into the dynamic behavior of mechanisms containing links with distributed mass and elasticity. *Mechanism and Machine Theory*, 7:13–17, 1972.

J. Mayo, J. Domínguez, and A. A. Shabana. Geometrically nonlinear formulations of beams in flexible multibody dynamics. *Journal of Vibration and Acoustics*, 117: 501–509, 1995.

J. Mayo, D. García-Vallejo, and J. Domínguez. Study of the geometric stiffening effect: comparison of different formulations. *Multibody System Dynamics*, 11(4): 321–341, 2004.

L. Meirovitch and M. K. Kwak. Convergence of the classical Rayleigh–Ritz method and the finite element method. *AIAA Journal*, 28(8):1509–1516, 1990.

A. M. Mikkola and A. A. Shabana. A non–incremental finite element procedure for the analysis of large deformation of plates and shells in mechanical system applications. *Multibody System Dynamics*, 9(3):283–309, 2003.

N. Newmark. A method of computation for structural dynamics. *Journal of the Engineering Mechanics Division, ASCE*, 85(EM3):67–94, 1959.

P. E. Nikravesh, I. Chung, and R. L. Bendict. Plastic hinge approach to vehicle crash simulation. *Computers & Structures*, 16:395–400, 1983.

M. Omar and A. A. Shabana. A two–dimensional shear deformable beam for large rotation and deformation problems. *Journal of Sound and Vibration*, 243:565–576, 2001.

X. Rui, B. He, Y. Lu, W. Lu, and G. Wang. Discrete time transfer matrix method for multibody system dynamics. *Multibody System Dynamics*, 14(3–4):317–344, 2005.

X. Rui, G. Wang, Y. Lu, and L. Yun. Transfer matrix method for linear multibody system. *Multibody System Dynamics*, 19(3):179–207, 2008.

R. Schwertassek, S. Dombrowski, and O. Wallrapp. Modal representation of stress in flexible multibody simulation. *Nonlinear Dynamics*, 20(4):381–399, 1999a.

R. Schwertassek, O. Wallrapp, and A. A. Shabana. Flexible multibody simulation and choice of shape functions. *Nonlinear Dynamics*, 20(4):361–380, 1999b.

A. A. Shabana. Substructure synthesis methods for dynamic analysis of multi–body systems. *Computers & Structures*, 20(4):737–744, 1985.

A. A. Shabana. Constrained motion of deformable bodies. *International Journal for Numerical Methods in Engineering*, 32(8):1813–1831, 1991.

A. A. Shabana. Finite element incremental approach and exact rigid body inertia. *Journal of Mechanical Design*, 118(2):171–178, 1996.

A. A. Shabana. Resonance conditions and deformable body co–ordinate systems. *Journal of Sound and Vibration*, 192(1):389–398, 1995.

A. A. Shabana. Flexible multibody dynamics: review of past and recent developments. *Multibody System Dynamics*, 1(2):189–222, 1997.

A. A. Shabana. *Dynamics of Multibody Systems*. Cambridge University Press, Cambridge, MA, 1998.

A. A. Shabana and R. A. Wehage. Variable degree of freedom component mode analysis of inertia–variant flexible mechanical systems. *Journal of Mechanisms, Transmissions and Automation in Design*, 105:370–378, 1983.

A. A. Shabana and R. Y. Yakoub. Three–dimensional absolute nodal coordinate formulation for beam elements: Theory. *Journal of Mechanical Design*, 123:606–613, 2001.

A. A. Shabana, Y. L. Hwang, and R. A. Wehage. Projection methods in flexible multibody dynamics. Part I: Kinematics. *International Journal for Numerical Methods in Engineering*, 35(10):1927–1939, 1992.

I. Sharf. Geometrically non–linear beam element for dynamics simulation of multibody systems. *International Journal for Numerical Methods in Engineering*, 39: 763–786, 1996.

P. Shi, J. McPhee, and G. Heppler. A deformation field for Euler–Bernoulli beams with applications to flexible multibody dynamics. *Multibody System Dynamics*, 5 (1):79–104, 2001.

J. C. Simó and L. Vu-Quoc. On the dynamics of flexible beams under large overall motions – The plane case: Part I. *Journal of Applied Mechanics*, 53:849–854, 1986a.

J. C. Simó and L. Vu-Quoc. On the dynamics of flexible beams under large overall motions – The plane case: Part II. *Journal of Applied Mechanics*, 53:855–863, 1986b.

J. C. Simó and L. Vu-Quoc. A finite strain beam formulation. The three–dimensional dynamic problem. Part I. *Computer Methods in Applied Mechanics and Engineering*, 49:253–271, 1986c.

J. C. Simó and L. Vu-Quoc. A three–dimensional finite strain rod model. Part II: Computational aspects. *Computer Methods in Applied Mechanics and Engineering*, 58:79–116, 1986d.

J. O. Song and E. J. Haug. Dynamic analysis of planar flexible mechanisms. *Computer Methods in Applied Mechanics and Engineering*, 24:359–381, 1980.

J. T. Sopanen and A. M. Mikkola. Description of elastic forces in absolute nodal coordinate formulation. *Nonlinear Dynamics*, 34(1–2):53–74, 2003.

W. Stelzle, A. Kecskeméthy, and M. Hiller. A comparative study of recursive methods. *Archive of Applied Mechanics*, 66:9–19, 1995.

H. Sugiyama, A. A. Shabana, M. A. Omar, and W. Loh. Development of nonlinear leaf spring model for multibody vehicle systems. *Computer Methods in Applied Mechanics and Engineering*, 195(50–51):6925–6941, 2006.

W. Sunada and S. Dubowsky. The application of finite element methods to the dynamic analysis of flexible spatial and coplanar linkage systems. *Journal of Mechanical Design*, 103:643–651, 1981.

W. Sunada and S. Dubowsky. On the dynamic analysis and behaviour of industrial robotic manipulators with elastic members. *Journal of Mechanical Design*, 105: 42–51, 1983.

R. E. Valembois, P. Fisette, and J. C. Samin. Comparison of various techniques for modelling flexible beams in multibody dynamics. *Nonlinear Dynamics*, 12(4):367–397, 1997.

T. Vampola and M. Valášek. Composite rigid body formalism for flexible multibody systems. *Multibody System Dynamics*, 18(3):413–433, 2007.

G. N. Vanderplaats. *Numerical Optimization Techniques for Engineering Design: with Application*. McGraw–Hill, New York, 1984.

S. Von Dombrowski. Analysis of large flexible body deformation in multibody systems using absolute coordinates. *Multibody System Dynamics*, 8(4):409–432, 2002.

N. Vukasovic. *Análisis dinámico de sistemas mecánicos con elementos flexibles en coordenadas naturales*. PhD thesis, University of Navarra, San Sebastián, Spain, 1990.

M. W. Walker and D. E. Orin. Efficient dynamic computer simulation of robotic mechanisms. *Journal of Dynamic Systems Measurement and Control*, 104(3):205–211, 1982.

O. Wallrapp. Standardization of flexible body modeling in multibody system codes, Part I: Definition of standard input data. *Mechanics of Structures and Machines*, 22 (3):283–304, 1994.

O. Wallrapp and S. Wiedemann. Simulation of deployment of a flexible solar array. *Multibody System Dynamics*, 7(1):101–125, 2002.

T. M. Wasfy and A. K. Noor. Computational strategies for flexible multibody systems. *Applied Mechanics Reviews*, 56(6):553–613, 2003.

R. A. Wehage and E. J. Haug. Generalized coordinate partitioning for dimension reduction in analysis of constrained dynamic systems. *Journal of Mechanical Design*, 104(1):247–255, 1982.

R. A. Wehage, A. A. Shabana, and Y. L. Hwang. Projection methods in flexible multibody dynamics. Part II: Dynamics. *International Journal for Numerical Methods in Engineering*, 35(10):1941–1966, 1992.

J. Wittenburg. *Dynamics of Systems of Rigid Bodies*. Teubner, Stuttgart, 1977.

S. C. Wu and E. J. Haug. Geometric non–linear substructuring for dynamics of flexible mechanical systems. *International Journal for Numerical Methods in Engineering*, 26(10):2211–2226, 1988.

R. Y. Yakoub and A. A. Shabana. Three dimensional absolute nodal coordinate formulation for beam elements: Implementation and applications. *Journal of Mechanical Design*, 123:614–621, 2001.

E. V. Zahariev. Nonlinear dynamics of rigid and flexible multibody systems. *Mechanics of Structures and Machines*, 28(1):105–136, 2000.

E. V. Zahariev. Relative finite element coordinates in multibody system simulation. *Multibody System Dynamics*, 7(1):51–77, 2002.

J. Znamenáček and M. Valášek. An efficient implementation of the recursive approach to flexible multibody dynamics. *Multibody System Dynamics*, 2(3):227–252, 1998.