MULTIBODY DYNAMICS 2007, ECCOMAS Thematic Conference
C.L. Bottasso, P. Masarati, L. Trainelli (eds.)
Milano, Italy, 25–28 June 2007

ANALYSIS OF LOAD PATTERNS IN RUBBER COMPONENTS FOR VEHICLES

Jerome Merel[†], Israël Wander[‡], Pierangelo Masarati[⋆], Marco Morandini[⋆]

[†]Numerical Simulation, Hutchinson Corporate Research Center
Rue Gustave Nourry - BP 31, 45120 Châlette sur Loing, France
web page: http://www.hutchinson.fr/

‡Apex Technologies
99bis, av. du Général Leclerc, 75014 Paris, France
e-mail: iw@apex-technologies.fr,
web page: http://www.apex-technologies.fr/

[⋆]Dipartimento di Ingegneria Aerospaziale, Politecnico di Milano
Via La Masa 34, 20156 Milano, Italy
e-mail: pierangelo.masarati@polimi.it, marco.morandini@polimi.it,
web page: http://www.aero.polimi.it/

Keywords: Structural Dynamics, Computational Mechanics, Visco-Elastic Constitutive Laws, Vehicle Dynamics.

Abstract. *This paper illustrates the application of the multibody formalism to the dynamic analysis of mechanical systems to determine the influence of rubber components on the dynamics of a vehicle suspension, and to verify the load levels in the components themselves.*

An essential advancement, in terms of consistency of the modeling and in reduction of the proneness to user errors when dealing with significantly anisotropic components and complex mechanisms, is represented by the possibility to eliminate, or at least reduce, any undue sensitivity of the model to implementation choices.

The paper presents a brief description of the developments of the software specifically originated from this work's requirements and examples of applications to realistic problems, including the entire suspension system of a conventional car.

1 INTRODUCTION

Multibody analysis represents a well-established analysis and design technique in the mechanical engineering field in a broad sense. Starting from early formulations, implementations and applications essentially targeted at rigid-body mechanism modeling [1], eventually it evolved into the thorough description of the dynamics of deformable systems, including the finite, arbitrarily large displacements and rotations that are typical of well-posed nonlinear finite elements [2, 3].

Hutchinson, as a world-wide supplier of high-technology rubber components for the automotive and aerospace industry, with a consolidated expertise in the detailed analysis of the rheological behavior of rubber components by means of nonlinear finite element analysis, is more and more often confronted with the need to provide detailed analysis of its products, integrated in the mechanical systems they are designed for. Thus multibody analysis represents the natural choice for a variety of applications ranging from engine mounts to wires, pipes and seals, from car suspension supports to helicopter elastomeric dampers.

1.1 Software Requirements

One of the requirements for a big company is the possibility to scale problem solving capabilities from centralized basic research centers to more specialized competence centers located as close as possible to the customers' needs, to address customers' specific problems based on the models developed by the research centers. This may result in a quite large number of licenses for very expensive software that for most uses exceeds by far the requirements. Moreover, due to the quite specific field of interest of Hutchinson, the company developed a strong background in specialized scientific software development, for the purpose of cultivating and preserving independent analysis capabilities.

As a consequence, to explore the opportunity of exploiting multibody analysis as a possible means to investigate and predict the behavior of the company's products in their operational environment, it appeared natural to choose an existing open source software, MBDyn. It is a free multibody analysis software developed at the Dipartimento di Ingegneria Aerospaziale of Politecnico di Milano [4, 5], for applications in the fields of structural dynamics and aeroservoelasticity of complex systems like helicopter rotors [6]. It is currently being evaluated for applications to the analysis of the mechanical systems where Hutchinson's products are typically involved.

MBDyn solves initial value problems in differential algebraic form. It deals efficiently with large rotations and displacements of components like rubber bushings, whose tangent principal directions can change drastically when strained, due to non linear characteristics and to finite changes of relative orientation. This may affect the coupling between the related components of the system. This change of orientation of the principal directions could be fully accounted for by non linear finite elements using a 3D volumic mesh of the bushing, but the size of such models could become impractical, for example, for multi-component simulations.

In many multibody packages, bushings are modeled as springs with non linear stiffness characteristics but with fixed principal directions, which may be appropriate for components whose relative orientations change by only a minimal amount. However, this prevents realistic changes of relative orientation in very flexible components, like rubber bushings in many automotive applications. These real-life orientation changes are enabled in MBDyn where no such restrictive assumptions are made on the 3D deformation and motion of the bushings. Moreover, the variety of junctions and the capability to mix rigid bodies with flexible parts like non linear beams and

linearly resonant parts is well suited for low frequency simulations of automotive suspensions.

2 SOFTWARE DESCRIPTION

To analyze generic mechanical systems, a fundamental requirement is the capability to efficiently model arbitrarily complex kinematic constraints. Another essential feature, in this specific analysis, is the availability of sophisticated deformable components that allow to model arbitrarily complex interactions between elements. For this purpose, MBDyn provides a library of deformable elements based on detailed kinematic modeling of finite displacements and rotations. Those components exploit a generic interface to arbitrary constitutive laws, which, if required, can fully participate in the computation of the Jacobian matrix to guarantee quadratic (or, occasionally, nearly-quadratic) convergence when solving Newton's iterations. Custom constitutive laws can be easily designed and implemented, either by exploiting the symbolic constitutive law element (a special constitutive law based on automatic differentiation of a symbolic expression, performed by means of the GiNaC library [7]), or by providing run-time loadable modules with the required methods. In this work, a family of specialized constitutive laws that model the behavior of rubber components has been developed.

The software structure is based on a clear separation between the connectivity and the constitutive laws. The connectivity takes care of applying the appropriate generalized force terms to the equations of motion, and to collect the kinematics of the straining for a specific deformable component from the configuration of the system, while the constitutive laws take care of the relationship between strains and stresses in the material frame. Constitutive laws are implemented using C++ *templates*; as a consequence, the same code can be used for 1, 3 and 6D constitutive laws, unless special requirements dictate a specialization for specific dimensionalities. 1D constitutive laws are typically used for lumped components, like rods; 3D constitutive laws are used for linear and angular strain components; 6D constitutive laws are uses for generic linear/angular strain components and for beam sections.

In the following, the term "linear strain component", according to recent literature [8], will be used to indicate deformable components that introduce internal forces dependent on the relative position of the connected nodes, without any reference or limitation to linear constitutive laws.

3 CONNECTIVITY

The formulation of the connectivity plays an important role in the modeling of deformable components. It is important to realize that a deformable component, like a displacement or rotation spring, represents a synthesis of a real mechanical component, like a bushing. As such, it ideally represents the result of an experiment, which relates forces and/or moments to the overall relative kinematics between the two ends of the structural component. The behavior of the component does not depend on the choice of the reference frame in which the constitutive behavior is expressed. However, the choice of the material reference frame may play an important role when significant relative displacements and rotations take place.

To handle this issue, two families of 3D deformable components has been developed. In the first case, called "attached", the material frame is considered attached to one of the nodes; this is the case, for example, of deformable components characterized by measuring forces and moments at one end of the component.

In the second case, called "invariant", the material frame is located halfway between the two reference points that represent the location of the deformable component with respect to each node, in terms of both position and orientation.

In the former case, exchanging the order of the nodes while using the same constitutive law changes the behavior of the component; in the latter case, this is no longer true, as the behavior does no longer depend on the ordering of the connectivity.

Since the measurement of the constitutive law parameters in the "invariant" case is typically not feasible, some manipulation may be required to project "attached" measurements into the "invariant" reference.

3.1 1D Components

One-dimensional components are rod-like deformable elements, whose straining is defined as a change in relative distance between two points in space. The distance between two points, optionally rigidly offset from the respective nodes by offsets $\boldsymbol{f}_1 = \boldsymbol{R}_1\tilde{\boldsymbol{f}}_1$, $\boldsymbol{f}_2 = \boldsymbol{R}_2\tilde{\boldsymbol{f}}_2$, is

$$l = \sqrt{\boldsymbol{l}^T\boldsymbol{l}}, \tag{1}$$

with

$$\boldsymbol{l} = \boldsymbol{x}_2 + \boldsymbol{f}_2 - \boldsymbol{x}_1 - \boldsymbol{f}_1. \tag{2}$$

The strain is defined as

$$\varepsilon = \frac{l}{l_0} - 1. \tag{3}$$

The force $f = f(\varepsilon, \dot{\varepsilon})$ exchanged between the two bodies is defined in terms of the strain and of its rate. Its contribution to the nodal equilibrium results from the application of the Virtual Work Principle (VWP), such that the virtual deformation work, defined as

$$\delta\mathcal{L} = \delta l^T f \tag{4}$$

is expanded to yield the internal force contribution to each independent kinematic parameter.

The perturbation of Eq. (1) yields[1]

$$\delta l = \frac{1}{l}\boldsymbol{l}^T\left(\delta\boldsymbol{x}_2 - \boldsymbol{f}_2 \times \boldsymbol{\theta}_{2\delta} - \delta\boldsymbol{x}_1 + \boldsymbol{f}_1 \times \boldsymbol{\theta}_{1\delta}\right), \tag{6}$$

so, after defining

$$\boldsymbol{F} = \frac{\boldsymbol{l}}{l} f, \tag{7}$$

the virtual internal work expands into

$$\boldsymbol{F}_1 = -\boldsymbol{F} \tag{8}$$
$$\boldsymbol{M}_1 = -\boldsymbol{f}_1 \times \boldsymbol{F} \tag{9}$$
$$\boldsymbol{F}_2 = \boldsymbol{F} \tag{10}$$
$$\boldsymbol{M}_2 = \boldsymbol{f}_2 \times \boldsymbol{F}, \tag{11}$$

that represent the contributions of the component to the force and moment equilibrium equations of the connected nodes.

[1] Here the notation $\boldsymbol{\theta}_{i\delta}$ is used to indicate the vector that describes an orientation perturbation, namely the vector resulting from the relationship

$$\delta\boldsymbol{R}_i = \boldsymbol{\theta}_{i\delta} \times \boldsymbol{R}_i. \tag{5}$$

3.2 3D Components: Attached Angular Spring

The angular spring is a component that introduces an internal moment as a consequence of a change in the relative orientation of two nodes. The relative orientation of the nodes is represented by matrix

$$\boldsymbol{R} = \boldsymbol{R}_1^T \boldsymbol{R}_2, \tag{12}$$

which can be conveniently expressed by a vector

$$\boldsymbol{\theta} = \mathrm{ax}\left(\exp^{-1}\left(\boldsymbol{R}\right)\right). \tag{13}$$

describing a rotation about its axis with the magnitude determined by the magnitude of the vector itself. The orientation vector is used as a measure of the angular strain of the component; the angular strain rate results from the differentiation of the relative orientation matrix itself, namely

$$\begin{aligned} \boldsymbol{\omega} \times \; &= \dot{\boldsymbol{R}}\boldsymbol{R}^T \\ &= \boldsymbol{R}_1^T \boldsymbol{\omega}_1 \times^T \boldsymbol{R}_1 + \boldsymbol{R}_1^T \boldsymbol{\omega}_2 \times \boldsymbol{R}_1 \\ &= \boldsymbol{R}_1^T \left(\boldsymbol{\omega}_2 - \boldsymbol{\omega}_1\right) \times \boldsymbol{R}_1, \end{aligned} \tag{14}$$

yielding

$$\boldsymbol{\omega} = \boldsymbol{R}_1^T \left(\boldsymbol{\omega}_2 - \boldsymbol{\omega}_1\right). \tag{15}$$

The moment $\bar{\boldsymbol{M}} = \bar{\boldsymbol{M}}\left(\boldsymbol{\theta}, \boldsymbol{\omega}\right)$ exchanged between the two nodes is defined in terms of the angular strain $\boldsymbol{\theta}$ and of its rate $\boldsymbol{\omega}$. Its contribution to the nodal equilibrium results from the VWP; the virtual deformation work

$$\delta\mathcal{L} = \boldsymbol{\theta}_\delta^T \bar{\boldsymbol{M}}, \tag{16}$$

after considering that the perturbation of the angular strain is

$$\boldsymbol{\theta}_\delta = \boldsymbol{R}_1^T \left(\boldsymbol{\theta}_{2\delta} - \boldsymbol{\theta}_{1\delta}\right), \tag{17}$$

results in the contributions

$$\boldsymbol{M}_1 = -\boldsymbol{M} \tag{18}$$

$$\boldsymbol{M}_2 = \boldsymbol{M}, \tag{19}$$

to the connected nodes moment equilibrium equations, where $\boldsymbol{M} = \boldsymbol{R}_1 \bar{\boldsymbol{M}}$.

3.3 3D Components: Invariant Angular Spring

The above described angular strain component expresses the internal moment in the reference frame attached to node 1; as a consequence, if an anisotropic constitutive law is used but the sequence of nodes 1 and 2 is reversed, a different behavior may appear as soon as the anisotropy of the constitutive law is exploited.

This behavior not only is expected; in many practical cases it may be even desirable, since it seems quite reasonable to set up experiments that allow to determine constitutive laws by measuring internal moments in a reference frame attached to one end of the deformable component.

However, it may be desirable to introduce a family of deformable components, which requires specifically determined constitutive laws, that is independent of the ordering of the connected nodes. This family is termed “invariant”. The rationale behind its design is that the

strain-reaction properties of a deformable component are intrinsic and do not depend on the reference frame they are referred to. What depends on the choice of the reference frame is the representation of those properties, namely, the way the deformable component kinematically interacts with its boundary.

A practical reason for introducing those components is that "attached" components may be error prone, since the behavior of the component itself depends on the ordering of the connected nodes, as illustrated in the buckling example of Section 3.5. The "invariant" form, on the contrary, behaves the same regardless of the ordering of the connected nodes, thus being much more user-friendly.

The basic idea, inspired by [8], consists in defining some non-dimensional coordinate ξ, ranging from 0 to 1, that allows to express the orientation of any intermediate frame between the two ends of the deformable component. The relative orientation at this coordinate can be expressed in terms of fractions of the overall relative orientation, assuming that the intermediate orientations are co-axial and thus can be summed; namely

$$\boldsymbol{\theta}\left(\xi\right) = \xi\boldsymbol{\theta}. \tag{20}$$

What is worth noticing is that the mid-span value of the coordinate, $\xi = 1/2$, defines an intermediate orientation $\tilde{\boldsymbol{\theta}}$ such that, by calling

$$\tilde{\boldsymbol{R}} = \exp\left(\tilde{\boldsymbol{\theta}} \times\right), \tag{21}$$

the relative orientation matrix can be expressed as

$$\boldsymbol{R} = \tilde{\boldsymbol{R}}\tilde{\boldsymbol{R}} = \tilde{\boldsymbol{R}}^2. \tag{22}$$

As a consequence, a perturbation of the relative orientation matrix yields

$$\begin{aligned}
\boldsymbol{\theta}_\delta \times \; &= \delta\boldsymbol{R}\boldsymbol{R}^T \\
&= \tilde{\boldsymbol{\theta}}_\delta \times \; + \tilde{\boldsymbol{R}}\tilde{\boldsymbol{\theta}}_\delta \times \tilde{\boldsymbol{R}}^T \\
&= \tilde{\boldsymbol{\theta}}_\delta \times \; + \left(\tilde{\boldsymbol{R}}\tilde{\boldsymbol{\theta}}_\delta\right) \times ,
\end{aligned} \tag{23}$$

so

$$\boldsymbol{\theta}_\delta = \left(\boldsymbol{I} + \tilde{\boldsymbol{R}}\right)\tilde{\boldsymbol{\theta}}_\delta, \tag{24}$$

and, according to Eq. (17),

$$\begin{aligned}
\tilde{\boldsymbol{\theta}}_\delta &= \left(\boldsymbol{I} + \tilde{\boldsymbol{R}}\right)^{-1}\boldsymbol{\theta}_\delta \\
&= \left(\boldsymbol{I} + \tilde{\boldsymbol{R}}\right)^{-1}\boldsymbol{R}_1^T\left(\boldsymbol{\theta}_{2\delta} - \boldsymbol{\theta}_{1\delta}\right).
\end{aligned} \tag{25}$$

Define now the orientation matrix $\hat{\boldsymbol{R}}$, which refers the intermediate reference frame with respect to the global frame:

$$\hat{\boldsymbol{R}} = \boldsymbol{R}_1\tilde{\boldsymbol{R}} \tag{26}$$

$$= \boldsymbol{R}_2\tilde{\boldsymbol{R}}^T. \tag{27}$$

The perturbation of matrix $\hat{\boldsymbol{R}}$ yields

$$\hat{\boldsymbol{\theta}}_\delta \times = \delta \hat{\boldsymbol{R}} \hat{\boldsymbol{R}}^T \tag{28}$$

$$= \boldsymbol{\theta}_{1\delta} \times + \boldsymbol{R}_1 \tilde{\boldsymbol{\theta}}_\delta \times \boldsymbol{R}_1^T \tag{29}$$

$$= \boldsymbol{\theta}_{2\delta} \times - \hat{\boldsymbol{R}} \tilde{\boldsymbol{\theta}}_\delta \times \hat{\boldsymbol{R}}^T, \tag{30}$$

resulting in

$$\hat{\boldsymbol{\theta}}_\delta = \boldsymbol{\theta}_{1\delta} + \boldsymbol{R}_1 \tilde{\boldsymbol{\theta}}_\delta \tag{31}$$

$$= \boldsymbol{\theta}_{2\delta} - \hat{\boldsymbol{R}} \tilde{\boldsymbol{\theta}}_\delta. \tag{32}$$

By exploiting Eq. (25), after some non-trivial matrix algebra manipulation, the two forms of $\hat{\boldsymbol{\theta}}_\delta$ reported above can be expressed as

$$\hat{\boldsymbol{\theta}}_\delta = \hat{\boldsymbol{I}} \boldsymbol{\theta}_{2\delta} + \hat{\boldsymbol{I}}^T \boldsymbol{\theta}_{1\delta}, \tag{33}$$

where

$$\hat{\boldsymbol{I}} = \hat{\boldsymbol{R}} \left(\boldsymbol{I} + \tilde{\boldsymbol{R}} \right)^{-1} \hat{\boldsymbol{R}}. \tag{34}$$

The relative velocity at the intermediate orientation $\tilde{\boldsymbol{R}}$ is

$$\bar{\boldsymbol{\omega}} = \tilde{\boldsymbol{R}}^T \boldsymbol{\omega}$$

$$= \hat{\boldsymbol{R}}^T \left(\boldsymbol{\omega}_2 - \boldsymbol{\omega}_1 \right). \tag{35}$$

The internal moment is now $\bar{\boldsymbol{M}} = \bar{\boldsymbol{M}} \left(\boldsymbol{\theta}, \bar{\boldsymbol{\omega}} \right)$; in fact, the angular strain does not depend on what intermediate orientation is considered, since it is coaxial to the relative orientation vector $\xi \boldsymbol{\theta}$ that defines the relative orientation matrix and, as such, $\tilde{\boldsymbol{R}}^T \boldsymbol{\theta} = \boldsymbol{\theta}$.

The contribution of the "invariant" angular strain component to the equilibrium of the connected nodes is again

$$\boldsymbol{M}_1 = -\boldsymbol{M} \tag{36}$$

$$\boldsymbol{M}_2 = \boldsymbol{M}, \tag{37}$$

but now $\boldsymbol{M} = \hat{\boldsymbol{R}} \bar{\boldsymbol{M}}$, and there is clearly no longer a bias towards any of the connected nodes.

3.4 3D Components: Attached Linear Spring

The linear spring introduces an internal moment based on the relative position of two points in space, optionally rigidly offset from the respective nodes by offsets analogous to those defined for the 1D components. The relative position, in the global reference frame, is

$$\boldsymbol{d} = \boldsymbol{x}_2 + \boldsymbol{f}_2 - \boldsymbol{x}_1 - \boldsymbol{f}_1, \tag{38}$$

which is analogous to the distance $\boldsymbol{l}$ defined in Eq. (2). In the reference frame of node 1, the relative position is

$$\tilde{\boldsymbol{d}} = \boldsymbol{R}_1^T \boldsymbol{d}. \tag{39}$$

Its time derivative represents the linear strain rate

$$\dot{\tilde{\boldsymbol{d}}} = \boldsymbol{R}_1^T \left(\dot{\boldsymbol{d}} - \boldsymbol{\omega}_1 \times \boldsymbol{d} \right). \tag{40}$$

By defining

$$\boldsymbol{d}_1 = \boldsymbol{x}_2 + \boldsymbol{f}_2 - \boldsymbol{x}_1, \tag{41}$$

Eq. (40) can be rewritten as

$$\dot{\tilde{\boldsymbol{d}}} = \boldsymbol{R}_1^T \left(\dot{\boldsymbol{d}}_1 - \boldsymbol{\omega}_1 \times \boldsymbol{d}_1 \right). \tag{42}$$

The internal force $\tilde{\boldsymbol{F}} = \tilde{\boldsymbol{F}} \left(\tilde{\boldsymbol{d}}, \dot{\tilde{\boldsymbol{d}}} \right)$ exchanged between the two nodes is defined in terms of the relative position $\tilde{\boldsymbol{d}}$ and of its rate $\dot{\tilde{\boldsymbol{d}}}$. Its contribution to the nodal equilibrium results from the VWP, where the deformation work

$$\delta \mathcal{L} = \delta \tilde{\boldsymbol{d}}^T \tilde{\boldsymbol{F}}, \tag{43}$$

after considering that the perturbation of the relative position is

$$\begin{aligned} \delta \tilde{\boldsymbol{d}} &= \boldsymbol{R}_1^T \left(\delta \boldsymbol{d} + \boldsymbol{d} \times \boldsymbol{\theta}_{1\delta} \right) \\ &= \boldsymbol{R}_1^T \left(\delta \boldsymbol{d}_1 + \boldsymbol{d}_1 \times \boldsymbol{\theta}_{1\delta} \right) \\ &= \boldsymbol{R}_1^T \left(\delta \boldsymbol{x}_2 - \boldsymbol{d}_2 \times \boldsymbol{\theta}_{2\delta} - \delta \boldsymbol{x}_1 + \boldsymbol{d}_1 \times \boldsymbol{\theta}_{1\delta} \right), \end{aligned} \tag{44}$$

with $\boldsymbol{d}_2 = \boldsymbol{f}_2$, results in the contributions

$$\boldsymbol{F}_1 = -\boldsymbol{F} \tag{45}$$

$$\boldsymbol{M}_1 = -\boldsymbol{d}_1 \times \boldsymbol{F} \tag{46}$$

$$\boldsymbol{F}_2 = \boldsymbol{F} \tag{47}$$

$$\boldsymbol{M}_2 = \boldsymbol{d}_2 \times \boldsymbol{F} \tag{48}$$

where $\boldsymbol{F} = \boldsymbol{R}_1 \tilde{\boldsymbol{F}}$. The above formulas are definitely biased towards node 1. In fact, the force $\tilde{\boldsymbol{F}}$ is projected in the global reference frame by $\boldsymbol{R}_1$. Moreover, while the arm $\boldsymbol{d}_2$ only depends on the orientation of node 2, the arm $\boldsymbol{d}_1$, defined in Eq. (41), depends on the position of both nodes and on the orientation of node 2 as well, but not on that of node 1.

An "invariant" form of this component has been developed as well; however, its formulation is a bit more involved, and is not reported here for the sake of conciseness.

3.5 Connectivity-Related Buckling

Consider now a simple example, made of a node grounded by an isotropic angular spring of stiffness k_θ=1e+3 Nm/radian and by an anisotropic linear spring of stiffness k_1=1e+3 N/m in the x and z directions, and of stiffness k_2=1e+2 N/m in the y direction. The node is loaded by a force in the z direction, namely in a principal direction of maximal stiffness. When a perturbation in the orientation of the node is introduced, by means of a very small constant moment about the x axis, so that the node rotates by a small amount, and the constitutive law is attached to the floating node, the orientation in which the constitutive law is expressed changes as well. As the load grows, this small perturbation in the orientation of the constitutive law may become sufficient, so that the deformable component finds a lower energy equilibrium condition by twisting and bending about the softer principal axis, as shown in Figures 1 and 2.

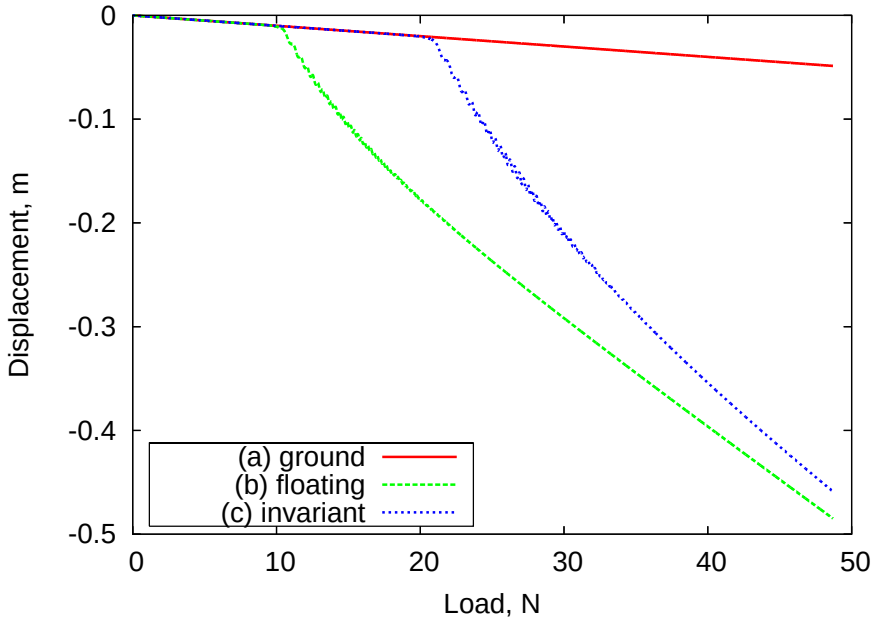

Figure 1: Vertical displacement of a node grounded by an isotropic angular spring and by an anisotropic linear spring, whose properties are "attached" to the ground (a), to the floating node (b), or referred to the intermediate orientation (c).

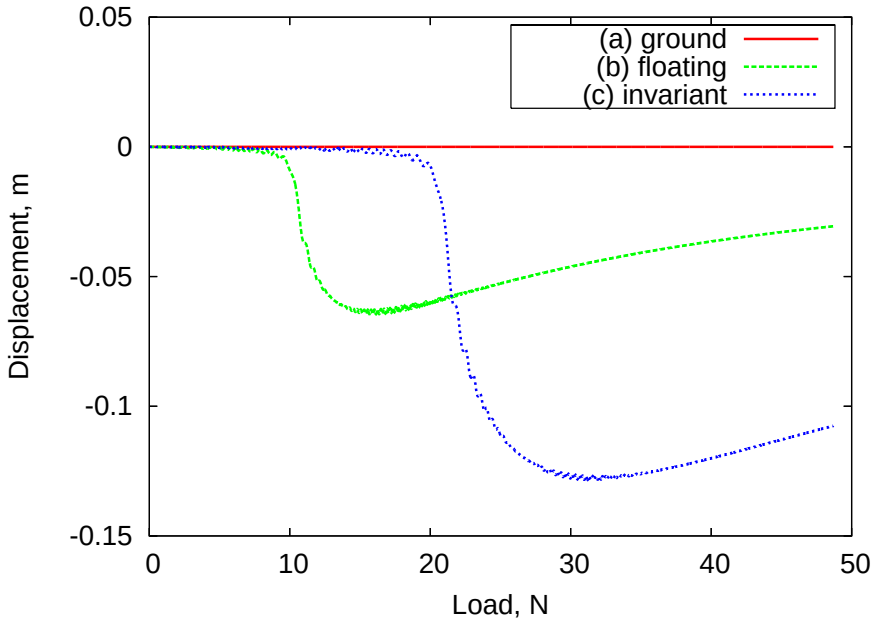

Figure 2: Lateral displacement of a node grounded by an isotropic angular spring and by an anisotropic linear spring, whose properties are "attached" to the ground (a), to the floating node (b), or referred to the intermediate orientation (c).

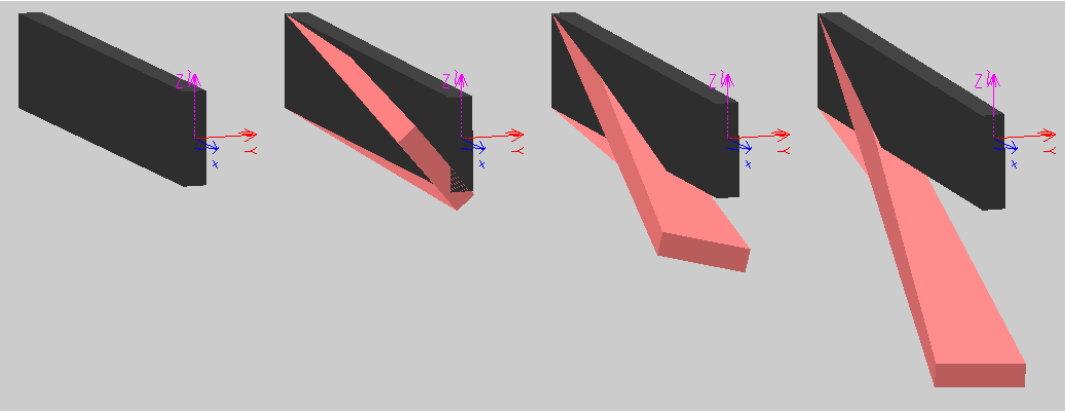

Figure 3: Sketch of the connectivity-related buckling at 0.0, 12.5, 25.0 and 50.0 N, left to right; the constitutive law of the lighter item is "attached" to the floating node.

This phenomenon is very well known for beams, and is called flexural-torsional buckling. This effect does not appear if the constant coefficients constitutive law is attached to the ground.

The point is that this kind of buckling should occur at the appropriate load level, which depends on the actual geometry and properties of the component that is here synthesized in lumped form, regardless of the way the lumped component is connected to the rest of the problem. This is where the "invariant" formulation of the linear and angular springs come into play. The three models presented are clearly different, and describe the behavior of different structural components. Only when the constitutive law is attached to the floating node the physically sound phenomenon of flexural-torsional buckling appears. This is also true when the "invariant" constitutive law is considered, although the phenomenon appears at a different load level (roughly twice), as expected, as the two problems describe different structural components. So the main advantage of the "invariant" constitutive law consists in freeing the operator from arbitrarily biasing the constitutive law of a component with respect to the connected nodes. It is clear that the same behavior could be obtained in all the three proposed cases by appropriately crafting the respective constitutive laws so that they correctly and consistently model the configuration dependent behavior of the actual component.

3.6 6D Components: Bushing

A real bushing may be represented by a combination of linear and angular springs, where the force and the moment depend on the relative displacement and rotation of the connected bodies. Right now, this type of element is present in MBDyn, but it is not fully implemented essentially because the full cross-coupling between linear and angular strains, and forces and moments is seldom required, mostly because the related constitutive data is not available.

3.7 6D Components: Beam

The software is based on an original finite volume implementation of nonlinear C^0 beams [9], which essentially consists in evaluating the equilibrium of finite portions of beams, cut at suitably chosen intermediate locations between the beam nodes. As a consequence, the formulation is very compact, and it only requires to evaluate the constitutive properties of selected beam sections as functions of appropriate measures of the linear and angular strains of the beam.

Without going into excessive detail, it is worth stressing that the beam elements allow to model arbitrarily large displacements and rotations, and to include a very accurate description

```
template <T, Tder>
class CL {
protected:
    T m_F;
    Tder m_FDE;
    Tder m_FDEP;
public:
    virtual ~CL(void) {};
    virtual void Update(const T& Eps, const T& EpsP) = 0;
    virtual void AfterConvergence(const Vector& X, const Vector& XP) {};
    const T& F(void) const { return m_F; };
    const Tder& FDE(void) const { return m_FDE; };
    const Tder& FDEP(void) const { return m_FDEP; };
};
```

Figure 4: Simplified form of the basic constitutive law C++ template class, that provides the basic interface (details omitted).

of the beam section kinematics and straining.

4 CONSTITUTIVE LAWS

Constitutive laws represent a critical aspect of the formulation. The linearization of the contributions of the deformable components to the equilibrium of the nodes is instrumental to the efficient solution of the problem by way of Newton iterations.

Each deformable component's internal force or moment is expressed in the form

$$\boldsymbol{f} = \boldsymbol{f}\left(\boldsymbol{x}, \dot{\boldsymbol{x}}\right) \tag{49}$$

where $\boldsymbol{f}$ and $\boldsymbol{x}$ have the same dimensionality (1, 3 or 6). The perturbation of the force yields

$$\delta\boldsymbol{f} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\delta\boldsymbol{x} + \frac{\partial \boldsymbol{f}}{\partial \dot{\boldsymbol{x}}}\delta\dot{\boldsymbol{x}} \tag{50}$$

so, in the most general case, it is the responsibility of the deformable component to take care of the perturbation of the strain and of its rate, while the computation of the value of the force and of its derivatives with respect to the strain and its rate is delegated to the constitutive law. The basic constitutive law C++ template class is illustrated in Fig. 4.

Its intended usage, as illustrated in Fig. 5, consists in calling the `Update()` method once for each iteration, which is supposed to update internal members containing the value of the force and of its derivatives, according to the dimensionality of the instance. As soon as convergence is reached, the `AfterConvergence()` method is called, which allows to deal with internal states updating and constitutive laws with memory.

In the simplest case, a linear, isotropic law is represented by two scalar constants k and r, such that the n-dimensional constitutive relation becomes

$$\boldsymbol{f}_n = k\boldsymbol{x}_n + r\dot{\boldsymbol{x}}_n \tag{51}$$

$$\frac{\partial \boldsymbol{f}_n}{\partial \boldsymbol{x}_n} = k\boldsymbol{I}_{n\times n} \tag{52}$$

$$\frac{\partial \boldsymbol{f}_n}{\partial \dot{\boldsymbol{x}}_n} = r\boldsymbol{I}_{n\times n}. \tag{53}$$

The use of C++ templates, in this very case, allowed to minimize code duplication, given the intrinsic simplicity and generality of the formulas, as illustrated in Fig. 6.

```
    // gather data
    CL<Vec3, Mat3x3>& cl;
    Vec3 Eps, EpsP;
    Vector X, XP;

    for (int i = 0; i < max_iterations; i++) {
        // after Eps, EpsP are prepared:
        cl.Update(Eps, EpsP);

        // when ready, use force and force derivatives...
        Residual.Add(cl.F());
        JacobianMatrix.Add(cl.FDE());
        JacobianMatrix.Add(cl.FDEP());
    }
    cl.AfterPredict(X, XP);
```

Figure 5: Usage of the constitutive law C++ template class (details omitted).

```
template <T, Tder>
class LinearViscoElasticIsotropicCL : public CL<T, Tder> {
private:
    double m_k;
    double m_r;

public:
    LinearViscoElasticIsotropicCL(const double& k, const double& r)
    : m_k(k), m_r(r) {
        // set the base class members, assuming Tder has a constructor
        // that builds a diagonal matrix out of a real
        CL<T, Tder>::m_FDE = m_k;
        CL<T, Tder>::m_FDEP = m_r;
    };
    void Update(const T& Eps, const T& EpsP) {
        CL<T, Tder>::m_F = Eps*m_k + EpsP*m_r;
    };
};

typedef LinearViscoElasticIsotropicCL<double, double>
    LinearViscoElasticIsotropicCL1D;
typedef LinearViscoElasticIsotropicCL<Vec3, Mat3x3>
    LinearViscoElasticIsotropicCL3D;
typedef LinearViscoElasticIsotropicCL<Vec6, Mat6x6>
    LinearViscoElasticIsotropicCL6D;
```

Figure 6: Linear visco-elastic isotropic constitutive law C++ template class; interface in Fig. 4 (details omitted).

```
template <T, Tder>
class LinearViscoElasticCL : public CL<T, Tder> {
public:
    LinearViscoElasticCL(const Tder& k, const Tder& r) {
        // directly set the base class members
        CL<T, Tder>::m_FDE = k;
        CL<T, Tder>::m_FDEP = r;
    };
    void Update(const T& Eps, const T& EpsP) {
        CL<T, Tder>::m_F = CL<T, Tder>::m_FDE*Eps
            + CL<T, Tder>::m_FDEP*EpsP;
    };
};

typedef LinearViscoElasticCL<double, double> LinearViscoElasticCL1D;
typedef LinearViscoElasticCL<Vec3, Mat3x3> LinearViscoElasticCL3D;
typedef LinearViscoElasticCL<Vec6, Mat6x6> LinearViscoElasticCL6D;
```

Figure 7: Linear visco-elastic anisotropic constitutive law C++ template class; interface in Fig. 4 (details omitted).

However, a generic anisotropic linear visco-elastic constitutive law, characterized by constant stiffness and damping matrices $\boldsymbol{k}_{n\times n}$ and $\boldsymbol{r}_{n\times n}$,

$$\boldsymbol{f}_n = \boldsymbol{k}_{n\times n}\boldsymbol{x}_n + \boldsymbol{r}_{n\times n}\dot{\boldsymbol{x}}_n \tag{54}$$

$$\frac{\partial \boldsymbol{f}_n}{\partial \boldsymbol{x}_n} = \boldsymbol{k}_{n\times n} \tag{55}$$

$$\frac{\partial \boldsymbol{f}_n}{\partial \dot{\boldsymbol{x}}_n} = \boldsymbol{r}_{n\times n}. \tag{56}$$

presents similar formulation simplicity, exploiting the template capability, as illustrated in Fig. 7. The same scheme, although occasionally in a less straightforward manner, applies to most of the generically nonlinear constitutive laws implemented in MBDyn.

5 APPLICATION: CAR SUSPENSION MODEL

This section presents a brief description of the activities performed for the evaluation of MBDyn as a solution to improve Hutchinson's analysis capabilities in the field of mechanical and aerospace systems. Some benchmarks have been considered, including the modeling of assembly procedures for deformable components and vibrations of complex mechanical systems suspended by nonlinear bushings, along with the modeling of typical mechanical problems. Figure 8 illustrates a multibody model of a car, one of the most sophisticated problems that have been considered. The suspension configuration and their parameters do not describe a real system, but rather a collection of realistic components to benchmark MBDyn's features.

The model has been analyzed to determine the loads in selected rubber bushings when the car is subjected to experimentally measured accelerations at the wheels, as is common practice in suspension test rigs for car suspension testing [10].

For this analysis, particular care has been put in the kinematic and dynamic model of the car suspensions, where the bushings subjected to the most significant stresses are located. For this purpose, nonlinear, kinematically exact finite volume beam elements [9] are used to model the torsion bar in the front suspension, and the twist beam in the rear suspension. The beams required a considerable number of elements, owing to their peculiar geometry, as illustrated in Fig. 9. The rest of the problem is modeled by rigid bodies connected by appropriate holonomic

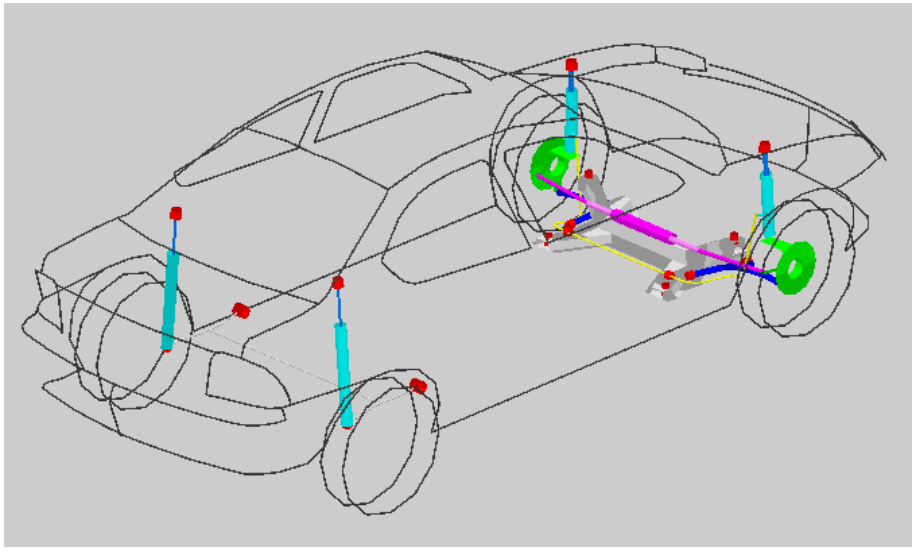

Figure 8: Car suspension model

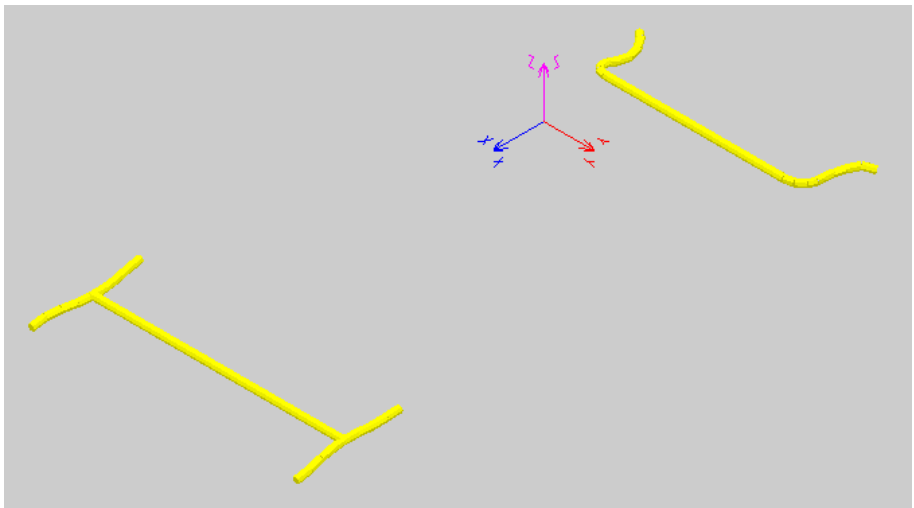

Figure 9: Detail of front torsion bar and rear twist beam.

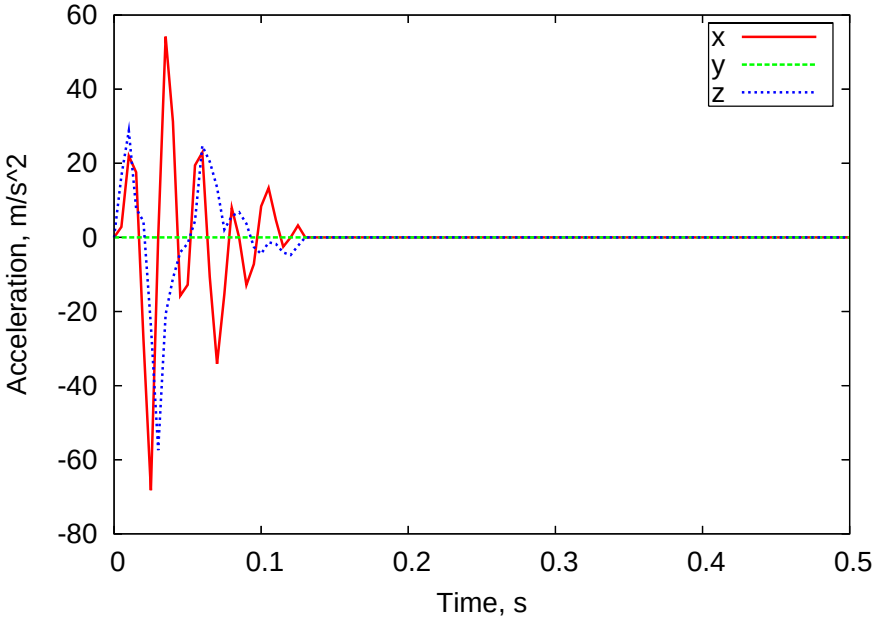

Figure 10: Car suspension model: imposed accelerations at front right wheel

constraints and by viscoelastic one- and three-dimensional components. Future developments will include a more detailed modeling of the overlay vehicle structure dynamics by means of component mode synthesis models of the chassis, to improve the fidelity of the analysis.

The problem is described by nearly 1300 equations, with about 100 structural nodes, nearly 60 nonlinear beam elements, and more than 80 joints. The time integration is performed by means of a second-order multistep A/L-stable integration scheme [11], that represents a good trade-off between accuracy and algorithmic dissipation. Time steps between 5.0 and 0.125 ms allow to capture the relevant dynamics without noteworthy convergence issues.

The main load cases correspond to applying a prescribed motion pattern at one or more wheel axes, measuring the overall behavior of the vehicle for validation purposes, and specifically the behavior of the suspensions to determine realistic load patterns and to assess the adequateness of the components.

Figure 10 shows an example of imposed acceleration pattern at the front right wheel axle. The corresponding response in terms of vehicle motion and suspension components loads is illustrated in Figures 11–16.

In the current implementation of the car suspension model, the focus is on the modeling capabilities and on the ability to predict the correct dynamic interaction between the suspension components. Future activity will include the modeling of nonlinear anelastic constitutive laws specific for rubber components.

6 CONCLUSIONS

The paper discussed the implementation of deformable components in the general-purpose free multibody software MBDyn. Components of different dimensionality: 1D, 3D and 6D have been formulated and implemented to fully support the needs of synthetic deformable systems analysis. Issues related to the capability of handling arbitrary displacements and rotations, and significantly to the invariance of the component with respect to its connectivity to the rest of the model have been addressed. As an example of the complexity of the models that can be

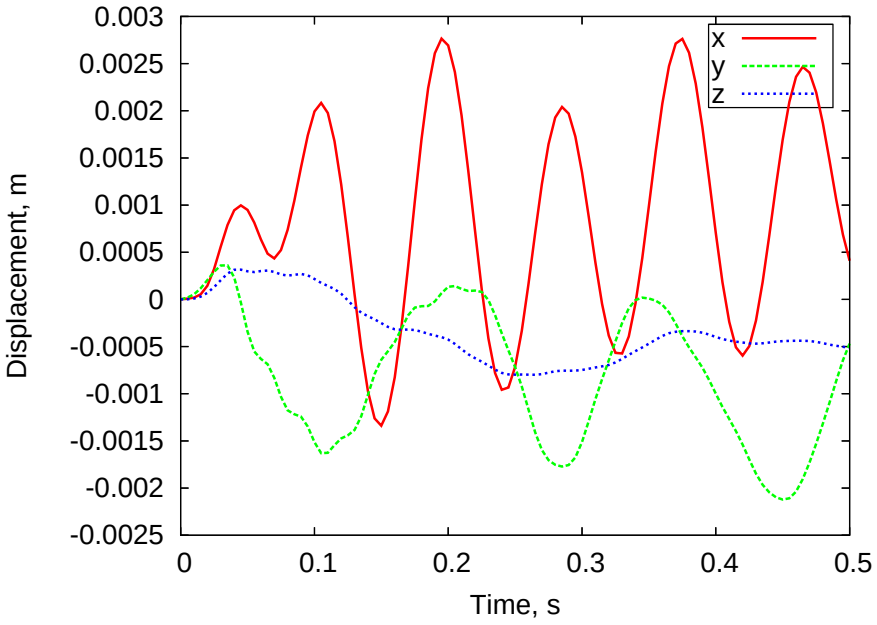

Figure 11: Car suspension model: car body CG motion

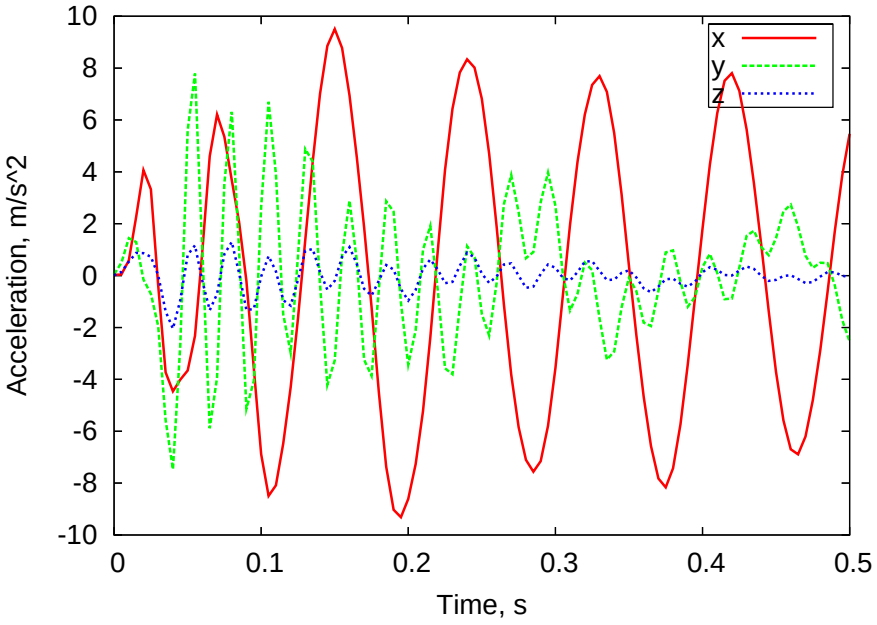

Figure 12: Car suspension model: car body CG accelerations

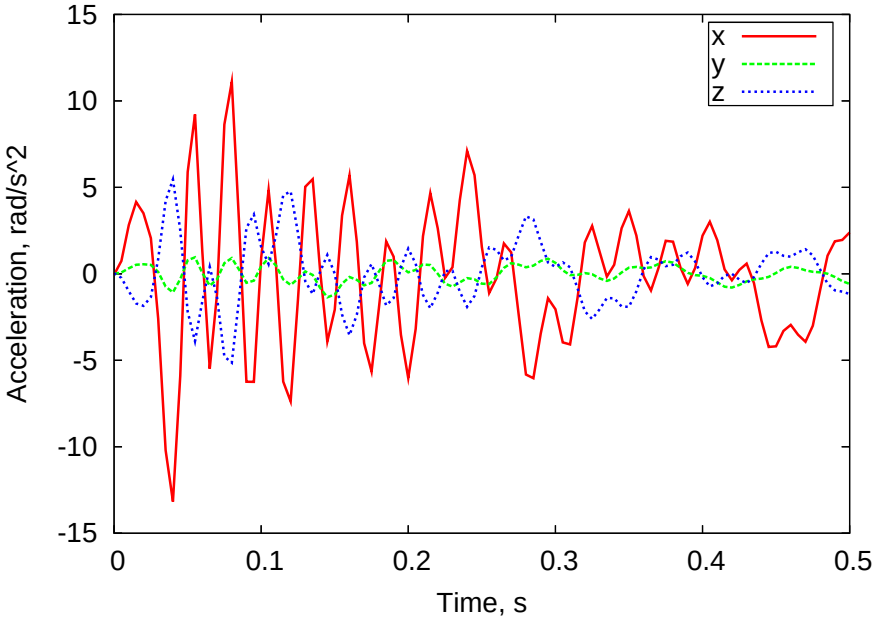

Figure 13: Car suspension model: car body angular accelerations

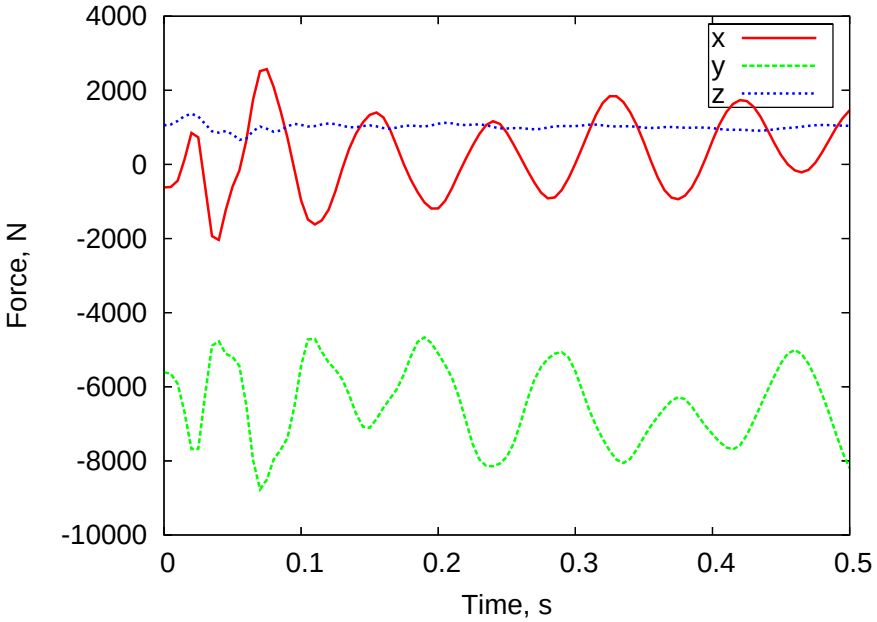

Figure 14: Car suspension model: front right shock absorber top bushing forces

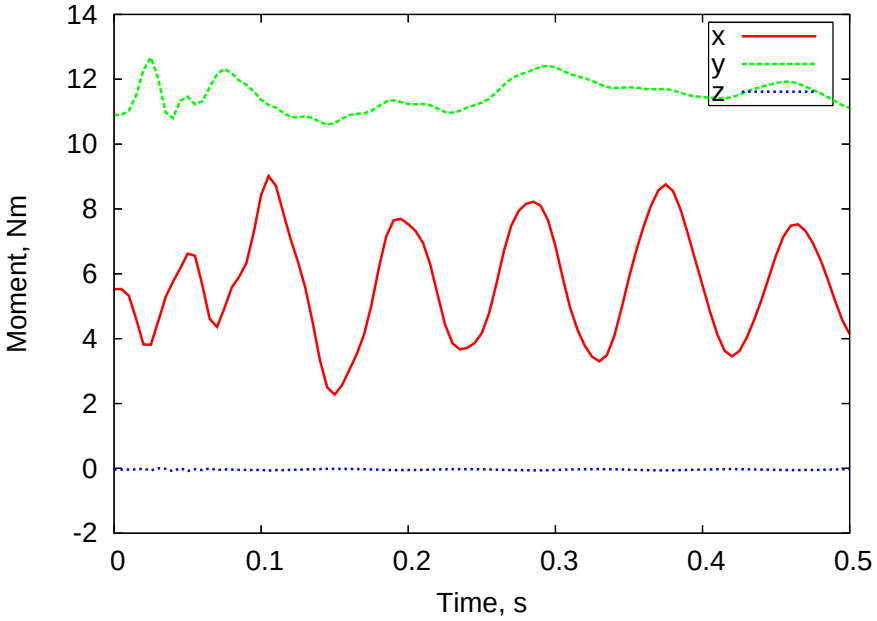

Figure 15: Car suspension model: front right shock absorber top bushing moments

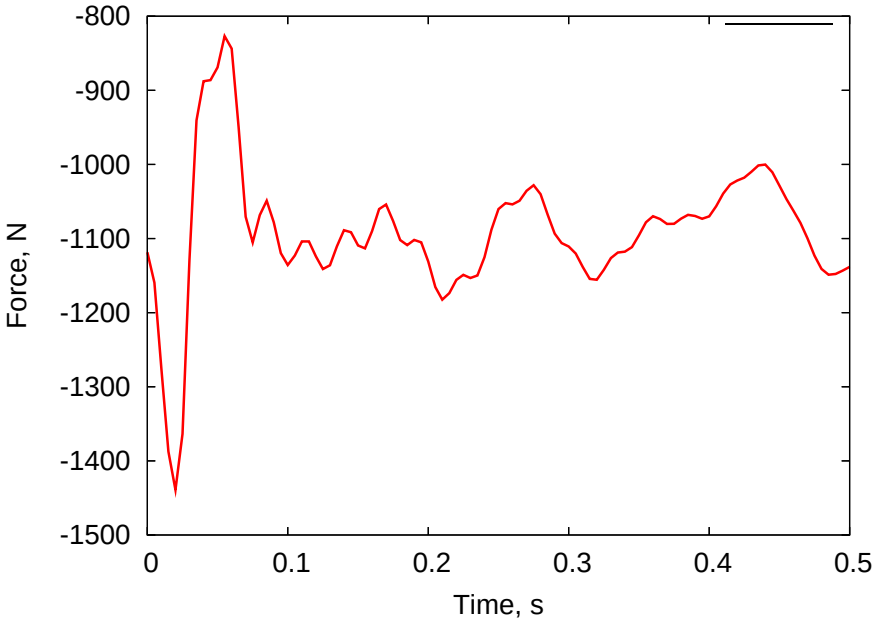

Figure 16: Car suspension model: front right shock absorber force

addressed, a complete vehicle suspension system has been modeled and analized under realistic load conditions. MBDyn is considered by Hutchinson as a viable tool for industrial exploitation.

7 ACKNOWLEDGMENTS

The authors acknowledge the support of Hutchinson SA R&D Centre and particularly of Dr. Daniel Benoualid in developing free multibody software.

REFERENCES

[1] Werner Schiehlen. *Multibody Systems Handbook*. Springer-Verlag, Berlin, 1990.

[2] Ahmed A. Shabana. *Dynamics of Multibody Systems*. Cambridge University Press, Cambridge, MA, 1998.

[3] Michel Geradin and Alberto Cardona. *Flexible Multibody Dynamics: a Finite Element Approach*. John Wiley & Sons, Chichester, 2001.

[4] Pierangelo Masarati, Marco Morandini, Giuseppe Quaranta, and Paolo Mantegazza. Open-source multibody analysis software. In *Multibody Dynamics 2003, International Conference on Advances in Computational Multibody Dynamics*, Lisboa, Portugal, July 1–4 2003.

[5] Pierangelo Masarati, Marco Morandini, Giuseppe Quaranta, and Paolo Mantegazza. Computational aspects and recent improvements in the open-source multibody analysis software "MBDyn". In *Multibody Dynamics 2005, ECCOMAS Thematic Conference*, Madrid, Spain, June 21–24 2005.

[6] Pierangelo Masarati, David J. Piatak, Giuseppe Quaranta, Jeffrey D. Singleton, and Jinwei Shen. Soft-inplane tiltrotor aeromechanics investigation using two multibody analyses. In 31st *European Rotorcraft Forum*, Firenze, Italy, 13–15 September 2005.

[7] `http://www.ginac.de/`.

[8] Teodoro Merlini and Marco Morandini. The helicoidal modeling in computational finite elasticity. part II: multiplicative interpolation. *International journal of solids and structures*, 41(18–19):5383–5409, 2004.

[9] Gian Luca Ghiringhelli, Pierangelo Masarati, and Paolo Mantegazza. A multi-body implementation of finite volume beams. *AIAA Journal*, 38(1):131–138, January 2000.

[10] M. Speckert, K. Dreßler, and H. Mauch. MBS simulation of a hexapod based suspension test rig. In *NAFEMS Seminar: Virtual Testing Simulation Methods as Integrated Part of an Efficient Product Development*, Wiesbaden, Germany, May 10–11 2006.

[11] Pierangelo Masarati, Massimiliano Lanz, and Paolo Mantegazza. Multistep integration of ordinary, stiff and differential-algebraic problems for multibody dynamics applications. In *XVI Congresso Nazionale AIDAA*, pages 71.1–10, Palermo, 24–28 Settembre 2001.