A modular modeling approach to simulate interactively multibody systems with Baumgarte/Uzawa formulation

Pierre Joli, Nicolas Seguy, Zhi-Qiang Feng

A Modular Modeling Approach to Simulate Interactively Multibody Systems With a Baumgarte/Uzawa Formulation

Pierre Joli
Nicolas Séguy
Laboratoire IBISC,
Université d'Evry-Val d'Essonne,
40 rue du Pelvoux,
91020 Evry, France

Zhi-Qiang Feng
Laboratoire de Mécanique
d'Evry, Université d'Evry-Val
d'Essonne, 40 rue du Pelvoux,
91020 Evry, France

In this paper, a modular modeling approach of multibody systems adapted to interactive simulation is presented. This work is based on the study of the stability of two differential algebraic equation solvers. The first one is based on the acceleration-based augmented Lagrangian formulation and the second one on the Baumgarte formulation. We show that these two solvers give the same results and have to satisfy the same criteria to stabilize the algebraic constraint acceleration error. For a modular modeling approach, we pro-pose to use the Baumgarte formulation and an iterative Uzawa algorithm to solve exter-nal constraint forces. This work is also the first step to validate the concept of two types of numerical components for object-oriented programming.

Keywords: DAE systems, numerical stability, modular modeling, interactive simulation

1 Introduction

Traditionally, a mechanical engineer in charge of the design of a mechanism with some expected behaviors starts with a draft version and then performs a kinematic, static, and dynamic analysis. According to the results, it refines the mechanism and restarts another cycle of design.

Introducing computer simulation accelerates this process considerably and for this reason a lot of software programs devoted to mechanical system simulation exist such as ADAMS, CATIA, SOLIDWORKS, etc. All these software programs have graphical user interfaces (GUIs) to help the design of mechanisms at the geometric level (to reinitialize the geometry and positions), sometimes at the static level (to reinitialize the forces and boundary conditions), but rarely at the dynamic level. It is not possible, for example, to modify the model during the simulation by eliminating, introducing, or modifying a part of the mechanism. It is necessary to follow the traditional cycle of modeling in these software programs, which involves a preprocessing phase (to define the geometry and the loading), a solution phase (to solve the equations of motion), and a postprocessing phase (to visualize and to analyze the results). The major reason for this shortcoming is that the dynamic model solved is not associated to an object-oriented design. Unlike conventional programming techniques, which require the developer to represent data and procedures separately, an object in C++ is a user-defined and self-contained entity composed of data (private or public) and procedures acting on data. So a developer can create an application as a collection of cooperating objects in which the behavior is completely defined [1,2].

A primary condition to create such objects consists of defining a dynamic model as a package of independent numerical subsystem components with their own system of coordinates and own numerical solver. A numerical subsystem component is associated with one rigid body or to a set of rigid bodies with no changing of topology. This is what we call the modular modeling, which is different from the centralized modeling in which the whole multibody system is defined by only one numerical component.

Taking the above considerations into account, the Lagrangian formulation offers the most general and versatile way to model multibody systems [3,4]. In this approach, the system is divided into several subsystems and glued together by introducing algebraic constraints and Lagrange multipliers. Because the modeling of each subsystem is independent of the topology of the system, it is possible to parallelize numerical tasks, which is the basic idea of numerous substructuring methods [5,6]. However, if the numerical task to glue the subsystems is centralized, then it is still necessary to build an admittance (or flexibility) matrix of all constraints to solve the constraint forces. In our modular approach, the numerical subsystems are connected between them by numerical joint components. Each numerical joint component computes external constraint forces only from output data of numerical subsystems. By taking advantage of object-oriented programming, an operator could interact easily with the virtual multibody system via a GUI or haptic devices. The concept of virtual reality in mechanical design could be one of the "leading-edge" technologies of the next decade [7,8]. As state of the arts of technologies in this field, we can cite the French PERF-RV platform in which a *Virtuose*™ 6D (Haption) has been connected to CATIA V5 (Dassault Systèmes) to manipulate digital mockups [9].

A numerical subsystem component has to be viewed as a "black box" where the input variables are positions, velocities, and accelerations at the beginning of the time step and the output variables are the same variables at the end of the time step. The accelerations are computed by a solver of ordinary differential equations (ODEs) or differential algebraic equations (DAEs) in case of internal constraints. The velocities and positions are calculated by a numerical integrator.

Many DAE solvers exist that we divide in two main classes:

- The first class of solvers calculates the Lagrange multipliers by a complete implicit formulation based on a Newton–Raphson procedure. These methods use an implicit numerical scheme and an incremental form (i.e., linearized form) of the DAE. At each time step of simulation, the displacements and the reaction forces are iteratively updated until the dynamic equilibrium equations, at the end of the time step, are satisfied. The numerical task is very high because at each iteration we have to invert a tangent matrix. In the case of multibody systems, this is a complex numerical task especially because of the nonlinearity induced by large rotations. Moreover, there are high frequencies of dynamic response

Contributed by the Design Engineering Division of ASME for publication in the JOURNAL OF COMPUTATIONAL AND NONLINEAR DYNAMICS. Manuscript received October 3, 2006; final manuscript received June 29, 2007; published online December 10, 2007. Review conducted by Aki Mikkola.

due to the algebraic constraints at the geometric level. An augmented Lagrangian formulation combined with the Hilbert–Hughes–Taylor algorithm (HHT) damps these undesired frequencies by numerical dissipation [10].

- The second class of solvers calculates the Lagrange multipliers by a partial implicit formulation because the dynamic equilibrium equations are solved from position and velocities estimated at each time step of simulation. The Lagrange multipliers are only implicitly dependent on the generalized accelerations. These methods are based on an explicit or prediction-correction numerical scheme (central difference, Newmark scheme, etc.), and on the transformation of the DAEs into ODEs by differentiating twice the geometric constraints [11]. The numerical task is very weak in comparison with the previous solvers. We can solve directly, simultaneously, or separately (by projection on the constraint directions) the generalized accelerations and the Lagrange multipliers. One well-known problem associated with this transformation is the drifting of geometric constraints due to accumulated errors from the time integration scheme. A method is necessary to stabilize the geometrical constraints if a long simulation time is required (Baumgarte stabilization [11]).

Of course other solvers exist, which avoid this nonexhaustive classification such as methods based on GGL stabilized index-2 formulation [12] or on the penalty formulation. For reasons of real-time simulation, we have focused our work on the study of the stability of the Baumgarte formulation [11] and the acceleration-based augmented Lagrangian formulation introduced by Bayo [13] belonging to the second class of solvers.

In the Bayo solver, the numerical convergence is very fast but it is not adapted to our design of modular modeling because the external constraint forces cannot be solved explicitly from the accelerations of the connected subsystems. We propose to use this efficient formulation only as a solver of internal constraints. Other formulations exist such as reducing the redundant coordinates and the number of internal algebraic equations of each subsystem with joint coordinates [14,15]. Another way is to eliminate Lagrange multipliers by partitioning dependent and independent coordinates [16]. The problem with these reductional methods is that singular positions or redundant constraints may lead to the inversion of an ill-conditioned matrix, which is not the case in the Bayo formulation. Using the pseudoinverse matrix by singular value decomposition may be an alternative solution [17].

In the Baumgarte solver, Lagrange multipliers and generalized accelerations are calculated directly (and noniteratively as in the Bayo formulation) by inversion of two matrices, the mass matrix and the impedance matrix of constraints. This solver is faster than the Bayo solver principally if the number of constraints is small (e.g., the size of the admittance matrix of constraints is small). However, it fails in the case of singular positions. Although the computation of the constraint forces is done independently of the accelerations, it is a partially modular approach because the construction of the admittance matrix of constraints requires a supervision of all the constraints by a centralized numerical task.

A fully modular approach is proposed in this paper, which is organized as follows. In Sec. 2, we introduce notations and we recall the geometric approach of DAE system as described by Blajer [18]. In Sec. 3, we show that the two solvers presented just above give the right result of constraint acceleration augmented by an error quantity. An adapted criteria of stability are defined to control this error. In Sec. 3, we propose a Baumgarte formulation associated with the iterative Uzawa algorithm to give a straightforward modular approach of modeling. This method is quite similar to the acceleration-based augmented Lagrangian formulation but the constraint forces are calculated explicitly from the generalized accelerations. In Sec. 4, we define our modeling approach with the two concepts of numerical subsystem components and numerical joint component. Finally, we present two standard numerical examples in mechanics. The first one is the double pendulum in plane to test the feasibility of our algorithm and the dynamic behavior of a mechanism by mouse interaction. The second one is the slider-crank mechanism as a closed loop mechanism to test the robustness of our algorithm. The algorithm is implemented in the FER/MECH software, which is developed in C++ by the authors [2,19]. In FER/MECH, friendly GUIs have been developed, which enable users to create, modify, and manipulate a multibody system intuitively and easily in 3D space but not yet during the solution phase. One purpose of this work is to develop such an interface during the solution phase.

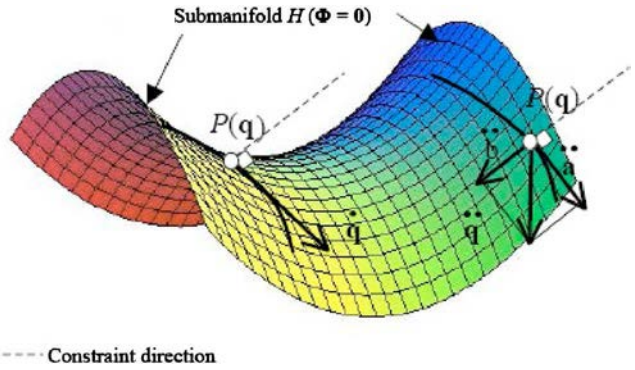

Fig. 1 Geometric interpretation of a constrained system

2 General Dynamic Model of a Multibody System With Holonomic Constraint

We will consider a multibody system characterized by n generalized coordinates $\mathbf{q}=[q_1 \cdots q_n]^T$, which are subject to m holonomic constraints $\mathbf{\Phi}=[\varphi_1 \cdots \varphi_m]^T$. The governing equations of the system at time t can be written in the following general matrix form:

$$\begin{cases} \mathbf{M}\ddot{\mathbf{q}} = \mathbf{F}(\mathbf{q},\dot{\mathbf{q}},t) + \mathbf{\Phi}_q^T\boldsymbol{\lambda} \\ \mathbf{\Phi}(\mathbf{q},t) = \mathbf{0} \end{cases} \tag{1}$$

where $\dot{\mathbf{q}},\ddot{\mathbf{q}}$ are, respectively, the generalized velocities and accelerations. $\mathbf{M}$ is the $n\times n$ symmetric positive-definite mass matrix of the system. $\boldsymbol{\lambda}=[\lambda_1 \ldots \lambda_m]^T$ denote the Lagrange multipliers. $\mathbf{\Phi}_q=\partial\mathbf{\Phi}/\partial\mathbf{q}$ is the $m\times n$ constraint Jacobian matrix. $\mathbf{\Phi}_q^T\boldsymbol{\lambda}$ are the generalized reaction forces due to the constraints. $\mathbf{F}(\mathbf{q},\dot{\mathbf{q}},t)$ represents other external and internal forces (Coriolis, gravity, motors, etc.).

Equation (1) is called a mixed system of DAEs. It can be deduced by different applications of the fundamental principle of mechanics (virtual work, Lagrange equations, Newton–Euler equations, etc.). The equations of motion and the constraint equations are, respectively, represented by the differential part and the algebraic part of this system. It is of interest to recall the geometric approach of such a system given by Blajer [18]: at each time t, the constrained motion of a multibody system can be represented as the motion of a generalized particle $P(\mathbf{q})$ in a submanifold H of an n-dimensional manifold E. The submanifold H is defined implicitly at each time by the equations $\mathbf{\Phi}(\mathbf{q},t)=\mathbf{0}$ and if all the holonomic constraints are independent, the dimension of H is $n-m$ (Fig. 1).

Equation (1) can be rewritten in a new form:

$$\begin{cases} \ddot{\mathbf{q}} = \ddot{\mathbf{a}} + \mathbf{M}^{-1}\mathbf{\Phi}_{\mathbf{q}}^T\boldsymbol{\lambda} \\ \mathbf{\Phi}(\mathbf{q},t) = \mathbf{0} \end{cases} \tag{2}$$

where $\ddot{\mathbf{a}}=\mathbf{M}^{-1}\mathbf{F}(\mathbf{q},\dot{\mathbf{q}},t)$ represents the tangential acceleration (Fig. 1).

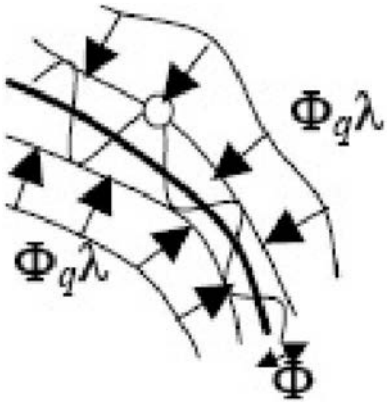

Fig. 2 Position error of particle *P*

The numerical solution of the DAE system is very sensitive to the integration truncation error and can lead to high numerical instability [10]. It stems from the ill conditioning of the leading matrix to solve simultaneously $\boldsymbol{\lambda}$ and $\ddot{\mathbf{q}}$ (DAE of index 3) [20]. Specific numerical methods have to be used carefully such as implicit Hilber–Hughes–Taylor algorithm with an incremental formulation of the system (2) [10].

For numerical instability purposes, instead of thinking about the constraints in terms of hard surfaces, it has been proposed to replace the *stiff* constraints by a strong force field $\boldsymbol{\Phi}_q^T\boldsymbol{\lambda}$ in the neighborhood of manifold $\boldsymbol{\Phi}(\mathbf{q},t)=\mathbf{0}$, directed toward the manifold H and proportional to the geometric constraint errors $\boldsymbol{\Phi}(\mathbf{q},t)$ and its derivatives (Fig. 2).

For reasons of real-time simulation, we consider in the following only predictor-corrector or explicit numerical schemes. This means we solve Eq. (2) at each time step of simulation from estimated values of $\mathbf{q}$ and $\dot{\mathbf{q}}$. So $\ddot{\mathbf{a}}$ and the Jacobian matrix $\boldsymbol{\Phi}_q$ are also estimated. $\mathbf{q}$, $\dot{\mathbf{q}}$, $\boldsymbol{\Phi}_q$ are constant during the time step of integration.

3 Different Adapted Approaches to Calculate the Constraint Forces

From the geometric constraint errors $\boldsymbol{\Phi}(\mathbf{q},t)$, we can define the acceleration constraint error $\ddot{\boldsymbol{\Phi}}$ by applying twice the total derivative operator with respect to time as follows:

$$\dot{\boldsymbol{\Phi}} = \boldsymbol{\Phi}_q\dot{\mathbf{q}} + \boldsymbol{\Phi}_t \quad \text{with } \boldsymbol{\Phi}_t = \frac{\partial\boldsymbol{\Phi}}{\partial t} \tag{3}$$

$$\ddot{\boldsymbol{\Phi}} = \boldsymbol{\Phi}_q\ddot{\mathbf{q}} + \dot{\boldsymbol{\Phi}}_q\dot{\mathbf{q}} + \dot{\boldsymbol{\Phi}}_t \tag{4}$$

In view of Eqs. (2) and (4) we have

$$\ddot{\boldsymbol{\Phi}} = \boldsymbol{\Phi}_q\ddot{\mathbf{a}} + \boldsymbol{\Phi}_q\mathbf{M}^{-1}\boldsymbol{\Phi}_q^T\boldsymbol{\lambda} + \dot{\boldsymbol{\Phi}}_q\dot{\mathbf{q}} + \dot{\boldsymbol{\Phi}}_t \tag{5}$$

or equivalently,

$$\mathbf{K}_\Phi\boldsymbol{\lambda} = \ddot{\boldsymbol{\Phi}} - \ddot{\boldsymbol{\Phi}}_f \tag{6}$$

with

$$\mathbf{K}_\Phi = \boldsymbol{\Phi}_q\mathbf{M}^{-1}\boldsymbol{\Phi}_q^T \quad \ddot{\boldsymbol{\Phi}}_f = \boldsymbol{\Phi}_q\ddot{\mathbf{a}} + \dot{\boldsymbol{\Phi}}_q\dot{\mathbf{q}} + \dot{\boldsymbol{\Phi}}_t \tag{7}$$

This system of equations represents the dynamic equilibrium equations following the constraint directions and points out the dependency between the constraint forces $\boldsymbol{\lambda}$ and the acceleration constraint errors $\ddot{\boldsymbol{\Phi}}$.

It is obvious that if the constraint reactions vanish ($\boldsymbol{\lambda}=\mathbf{0}$) then the acceleration constraint errors satisfied $\ddot{\boldsymbol{\Phi}}=\ddot{\boldsymbol{\Phi}}_f$ and so $\ddot{\boldsymbol{\Phi}}_f$ represents the free acceleration. Consequently, $\mathbf{K}_\Phi\boldsymbol{\lambda}$, which is the other part of the constraint acceleration error, represents the constrained acceleration.

If $\ddot{\boldsymbol{\Phi}}=\mathbf{0}$ then $\mathbf{K}_\Phi\boldsymbol{\lambda}_c=-\ddot{\boldsymbol{\Phi}}_f$ where $\boldsymbol{\lambda}_c$ represents the exact constraint forces to enforce the acceleration constraint errors to zero and $\mathbf{K}_\Phi$ represents the constraint admittance matrix.

From the previous considerations, we propose two equivalent formulations to solve the constraint forces $\boldsymbol{\lambda}$ and to establish the criteria of stability to control the solution.

3.1 Baumgarte Formulation. We consider in this subsection the following relationship:

$$\mathbf{K}_\Phi\boldsymbol{\lambda} = -\ddot{\boldsymbol{\Phi}}_f - k_v\dot{\boldsymbol{\Phi}} - k_p\boldsymbol{\Phi} \tag{8}$$

where k_p and k_v are, respectively, stiffness and viscous damping parameters. They can be considered as feedback gains defined by numerical considerations of precision and stability as we will see just below.

By substitution between Eqs. (6) and (8), we obtain the following linear differential equation of an oscillator:

$$\ddot{\boldsymbol{\Phi}} + 2\varepsilon\omega_0\dot{\boldsymbol{\Phi}} + \omega_0^2\boldsymbol{\Phi} = \mathbf{0} \quad \text{with } \omega_0 = \sqrt{k_p} \quad \text{and } \varepsilon = \frac{k_v}{2\omega_0} \tag{9}$$

ω_0 is the natural frequency of the oscillator and ε the viscuous damping parameter. If $\varepsilon^2 \ll 1$ and if we consider the initial velocity constraint error $\dot{\boldsymbol{\Phi}}_0$ then the analytical solution is

$$\boldsymbol{\Phi} = \frac{\dot{\boldsymbol{\Phi}}_0}{\omega_0}e^{-\varepsilon\omega_0 t}\sin(\omega_0 t) \tag{10}$$

$\boldsymbol{\Phi}$ is then bounded by $\dot{\boldsymbol{\Phi}}_0/\omega_0$ and tends to zero when $t\to\infty$.

The bigger the stiffness parameter is, the bigger the natural frequency is and the smaller the geometric constraint error is. The generalized particle P has an oscillatory motion in the neighborhood of the manifold H defined by $\boldsymbol{\Phi}(\mathbf{q},t)=\mathbf{0}$ (Fig. 2). If the geometric constraint errors are stabilized then the consequent error on the position of the generalized particle P does not affect the stability of the system. However, we need to define a criteria of constraint stabilization to respect this condition. Following the idea of the Shannon sampling theorem, We need at least four correct values of $\boldsymbol{\Phi}(\mathbf{q},t)$ during each oscillator's wave to describe correctly the geometric constraint error. If we consider ten values, we have the following criteria:

$$\Delta t = \frac{T}{10} \Leftrightarrow \omega_0 = \frac{0.2\pi}{\Delta t} \tag{11}$$

where Δt is the time step of the simulation and T the period of oscillator.

If the time step of simulation is fixed by the operator, the feedback gains have to satisfy the following criteria of constraint stabilization:

$$k_p = \frac{0.394}{\Delta t^2} \quad k_v = \frac{0.125}{\Delta t} \quad \text{with } \varepsilon = 0.1 \tag{12}$$

The initial DAE system (1) has been transformed into the following ODE system:

$$\begin{cases} \mathbf{M}\ddot{\mathbf{q}} = \mathbf{F}(\mathbf{q},\dot{\mathbf{q}},t) + \boldsymbol{\Phi}_\mathbf{q}^T\boldsymbol{\lambda} \\ \mathbf{K}_\Phi\boldsymbol{\lambda} = -\ddot{\tilde{\boldsymbol{\Phi}}}_f \quad \text{with } \ddot{\tilde{\boldsymbol{\Phi}}}_f = \ddot{\boldsymbol{\Phi}}_f + k_v\dot{\boldsymbol{\Phi}} + k_p\boldsymbol{\Phi} \end{cases} \tag{13}$$

It is easy to see that this formulation is similar to the Baumgarte formulation [11]:

$$\begin{cases} \mathbf{M}\ddot{\mathbf{q}} = \mathbf{F}(\mathbf{q},\dot{\mathbf{q}},t) + \boldsymbol{\Phi}_\mathbf{q}^T\boldsymbol{\lambda} \\ \ddot{\boldsymbol{\Phi}} + k_v\dot{\boldsymbol{\Phi}} + k_p\boldsymbol{\Phi} = \mathbf{0} \end{cases} \tag{14}$$

The matrix $\mathbf{K}_\Phi$ is definite positive and is singular only when the constraint equations are not independent (semipositive). In this

case, we could use a full pivoting strategy or pseudoinverse matrix ($\mathbf{K}_{\mathbf{\Phi}}^{+}$) to solve $\boldsymbol{\lambda}$. Whatever the strategy of computation used to solve $\boldsymbol{\lambda}$, it can be shown that it does not affect the uniqueness of the solution $\ddot{\mathbf{q}}$.

From the above considerations, the constrained acceleration $\mathbf{K}_{\mathbf{\Phi}}\boldsymbol{\lambda}$ is equal to $\mathbf{K}_{\mathbf{\Phi}}\boldsymbol{\lambda}_c - k_v\dot{\mathbf{\Phi}} - k_p\mathbf{\Phi}$ where $k_v\dot{\mathbf{\Phi}} + k_p\mathbf{\Phi}$ represents a measure of the error. The relative error can be defined as follows:

$$\Sigma = \frac{\left\| k_v\dot{\mathbf{\Phi}} + k_p\mathbf{\Phi} \right\|}{\left\| \mathbf{K}_{\mathbf{\Phi}}\boldsymbol{\lambda} \right\|} \tag{15}$$

The higher Σ is, the lower stability is of the constrained acceleration around the referenced value $\mathbf{K}_{\mathbf{\Phi}}\boldsymbol{\lambda}_c$. Σ can also be related to the curvature of the manifold $\mathbf{\Phi}(\mathbf{q},t)=\mathbf{0}$, the higher it is, the stronger the curvature is.

3.2 Acceleration-Based Augmented Lagrangian Formulation. In this section, we introduce augmented Lagrange multipliers $\boldsymbol{\lambda}$, which are implicitly computed from a method based on an iterative computation of the acceleration constraint error

$$\boldsymbol{\lambda}_{i+1} = \boldsymbol{\lambda}_i - \alpha\ddot{\mathbf{\Phi}}_{i+1} \quad (\alpha > 0) \tag{16}$$

At the first iteration, $\boldsymbol{\lambda}_0=\mathbf{0}$. Equation (6) imply the two following equations:

$$\boldsymbol{\lambda}_i = \mathbf{K}_{\Phi}^{+}(\ddot{\mathbf{\Phi}}_i - \ddot{\mathbf{\Phi}}_f) \tag{17}$$

$$\boldsymbol{\lambda}_{i+1} = \mathbf{K}_{\Phi}^{+}(\ddot{\mathbf{\Phi}}_{i+1} - \ddot{\mathbf{\Phi}}_f) \tag{18}$$

By substitution between Eqs. (17), (18), and (16) we obtain the following recursive formula:

$$\ddot{\mathbf{\Phi}}_{i+1} = (\mathbf{K}_{\Phi}^{+} + \alpha)^{-1}\mathbf{K}_{\mathbf{\Phi}}^{+}\ddot{\mathbf{\Phi}}_i = ((\mathbf{K}_{\mathbf{\Phi}}^{+} + \alpha)^{-1}\mathbf{K}_{\mathbf{\Phi}}^{+})^{i+1}\ddot{\mathbf{\Phi}}_f \tag{19}$$

It is noted from Eq. (17) that, if $\ddot{\mathbf{\Phi}}_{i+1}\to 0$, we have then $\boldsymbol{\lambda}_i \to -\mathbf{K}_{\mathbf{\Phi}}^{+}\ddot{\mathbf{\Phi}}_f$.

The complete numerical algorithm can be summarized as follows:

$$\begin{cases} (\mathbf{M} + \mathbf{\Phi}_q^T\alpha\mathbf{\Phi}_q)\ddot{\mathbf{q}}_{i+1} = \mathbf{M}\ddot{\mathbf{a}} + \mathbf{\Phi}_q^T(\boldsymbol{\lambda}_i - \alpha(\dot{\mathbf{\Phi}}_q\dot{\mathbf{q}} + \dot{\mathbf{\Phi}}_t)) \\ \boldsymbol{\lambda}_0 = \mathbf{0} \\ \boldsymbol{\lambda}_i = \boldsymbol{\lambda}_{i-1} - \alpha\ddot{\mathbf{\Phi}}_i \end{cases} \tag{20}$$

Of course, the higher α is, the better the numerical convergence is. In theory, there is no upper limit to choose α except having a good numerical conditioning of the matrix $(\mathbf{M}+\mathbf{\Phi}_q^T\alpha\mathbf{\Phi}_q)$. In two or three iterations, the numerical convergence of $\boldsymbol{\lambda}$ to $\boldsymbol{\lambda}_c$ is obtained.

This formulation gives the same solution as the following ODE system:

$$\begin{cases} \mathbf{M}\ddot{\mathbf{q}} = \mathbf{F}(\mathbf{q},\dot{\mathbf{q}},t) + \mathbf{\Phi}_q^T\boldsymbol{\lambda} \\ \ddot{\mathbf{\Phi}} = \mathbf{0} \end{cases} \tag{21}$$

which is similar to the previous Baumgarte formulation without feedback gains to stabilize the geometric constraint errors. If we consider initial velocity constraint errors $\dot{\mathbf{\Phi}}_0$ then the analytical solution of the geometric constraint error is $\mathbf{\Phi}=\dot{\mathbf{\Phi}}_0 t$.

This is the well-known phenomena of constraint drifting, where the generalized particule is no longer stuck on the manifold H due to small perturbations on velocity constraint errors. This undesired displacement, after a long time of simulation, can lead to a large error on the positioning of the generalized particle P and to numerical instability. In order to prevent this problem, Bayo has proposed to change the iterative calculation of $\boldsymbol{\lambda}$ as follows [13]:

$$\boldsymbol{\lambda}_{i+1} = \boldsymbol{\lambda}_i - \alpha(\ddot{\mathbf{\Phi}}_{i+1} + k_v\dot{\mathbf{\Phi}} + k_p\mathbf{\Phi}) \tag{22}$$

Following the same reasoning as in Eq. (19), it is easy to prove that

$$\ddot{\mathbf{\Phi}}_{i+1} \to -\alpha(k_v\dot{\mathbf{\Phi}} + k_p\mathbf{\Phi}) \quad \text{and } \boldsymbol{\lambda}_i \to -\mathbf{K}_{\Phi}^{+}(\ddot{\mathbf{\Phi}}_0 + k_v\dot{\mathbf{\Phi}} + k_p\mathbf{\Phi}) \tag{23}$$

The solution converges to the same one as in the Baumgarte formulation in Eq. (19), so we can use the same criteria of stability [Eq. (12)]. Finally, in this last formulation, the system of equations becomes

$$\begin{cases} (\mathbf{M} + \mathbf{\Phi}_q^T\alpha\mathbf{\Phi}_q)\ddot{\mathbf{q}}_{i+1} = \mathbf{M}\ddot{\mathbf{a}} + \mathbf{\Phi}_q^T(\boldsymbol{\lambda}_i - \alpha(\dot{\mathbf{\Phi}}_q\dot{\mathbf{q}} + \dot{\mathbf{\Phi}}_t)) \\ \boldsymbol{\lambda}_0 = \mathbf{0} \\ \boldsymbol{\lambda}_i = \boldsymbol{\lambda}_{i-1} - \alpha(\ddot{\mathbf{\Phi}}_i + k_v\dot{\mathbf{\Phi}} + k_p\mathbf{\Phi}) \end{cases} \tag{24}$$

3.3 Baumgarte/Uzawa Formulation. We have examined two methods, which can be used easily with an explicit or prediction-correction numerical scheme to solve a DAE. We have proved that the acceleration-based augmented Lagrangian formulation converges to the same solution obtained in the Baumgarte formulation. For these two methods, we have defined the same criteria [Eq. (12)] to control the stability of the acceleration constraint error, which is only dependent on the time step of simulation. The advantage of the acceleration-based augmented Lagrangian formulation is that there is no problem of singularity; however, we have to successively invert the mass matrix $\mathbf{M}$ and the augmented mass matrix $(\mathbf{M}+\mathbf{\Phi}_q^T\alpha\mathbf{\Phi}_q)$ several times (two or three times according to Bayo). In the Baumgarte formulation, we have to successively invert the mass matrix $\mathbf{M}$ and the impedance matrix $\mathbf{K}_{\mathbf{\Phi}}$ only once. If there is no problem of singularity due to dependent constraints, the Baumgarte formulation seems to be more efficient. We did not discuss here the sparsity of the matrix $\mathbf{K}_{\mathbf{\Phi}}$ or $(\mathbf{M}+\mathbf{\Phi}_q^T\alpha\mathbf{\Phi}_q)$, which does seem no longer an argument because efficient methods exist to factorize such matrices as the sparse $\mathbf{LDL}^T$ factorization [4].

For these two methods, we have to invert a matrix $[(\mathbf{M}+\mathbf{\Phi}_q^T\alpha\mathbf{\Phi}_q)$ and $\mathbf{K}_{\Phi}]$ due to the coupling between the subsystems. This numerical task is not modular because the procedure involved acts on private data belonging to different subsystems.

For a straightforward modular approach, we propose to solve the equation $\mathbf{K}_{\mathbf{\Phi}}\boldsymbol{\lambda} = -\ddot{\tilde{\mathbf{\Phi}}}_f$ established in the Baumgarte formulation by the following Uzawa algorithm [21]:

$$\boldsymbol{\lambda}_{i+1} = \boldsymbol{\lambda}_i - \rho(\mathbf{K}_{\mathbf{\Phi}}\boldsymbol{\lambda}_i + \ddot{\tilde{\mathbf{\Phi}}}_f) \quad \text{with } \boldsymbol{\lambda}_0 = \mathbf{0} \quad \text{and } \rho > 0 \tag{25}$$

Because the diagonal terms K_{jj} of $\mathbf{K}_{\Phi}$ are positive and dominant, we have to satisfy the following convergence criteria:

$$\rho \leqslant \inf(1/K_{jj}) \tag{26}$$

Considering Eq. (6), Eq. (25) can be easily transformed as follows:

$$\boldsymbol{\lambda}_{i+1} = \boldsymbol{\lambda}_i - \rho\ddot{\tilde{\mathbf{\Phi}}}_i \quad \text{with } \ddot{\tilde{\mathbf{\Phi}}}_0 = \ddot{\tilde{\mathbf{\Phi}}}_f \quad \text{and } \ddot{\tilde{\mathbf{\Phi}}}_i = \ddot{\mathbf{\Phi}}_i + k_v\dot{\mathbf{\Phi}} + k_p\mathbf{\Phi} \tag{27}$$

In this formulation, the constraint reactions are only explicitly dependent on the acceleration constraint errors $\ddot{\mathbf{\Phi}}_i$. As we know the analytical relations of $\ddot{\mathbf{\Phi}}_i$, they can be directly calculated from acceleration computed at each iteration i and from the estimated position and velocity vectors $\mathbf{q},\dot{\mathbf{q}}$. So it is no longer necessary to build the admittance matrix of constraint $\mathbf{K}_{\Phi}$. A numerical tolerance of convergence can be defined as follows:

$$\boldsymbol{\lambda}\,\% = (\boldsymbol{\lambda}_{i+1} - \boldsymbol{\lambda}_i)/\boldsymbol{\lambda}_{i+1} \tag{28}$$

4 Modular Modeling Approach

In our design of modular modeling, the numerical joint component is defined as a "white box" (with public data and algebraic equations) connected to two numerical subsystem components considered as "black boxes" (with private data and DAE solver).

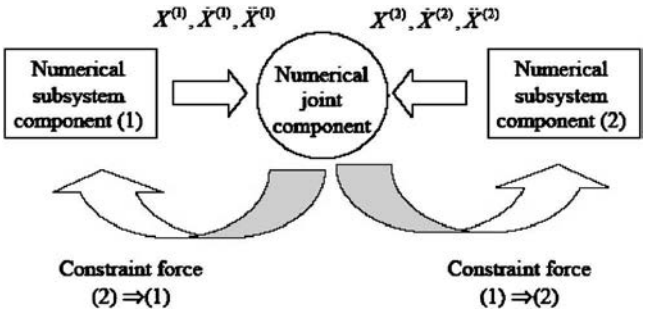

Fig. 3 Basic principle of the modular modeling

It gives the constraint active/reactive forces between these two numerical subsystem components, which are calculated only from the accelerations, velocities, and positions of connected points (Fig. 3).

So a multibody system could be divided into independent numerical subsystem components with its own:

- Mechanical parameters: mass matrix $\mathbf{M}$, numerical gains k_v, k_p of stabilization of the internal constraints
- Numerical explicit or prediction-correction scheme
- Formulation (Baumgarte or Bayo formulation, etc.)

Considering a pair of subsystem components only constrained by internal constraints, with the notations defined in Sec. 2, we have the following equations:

$$\text{Subsystem 1:} \quad \ddot{\mathbf{q}}^{(1)} = \ddot{\mathbf{a}}^{(1)} + \ddot{\mathbf{b}}^{(1)} \tag{29}$$

$$\text{Subsystem 2:} \quad \ddot{\mathbf{q}}^{(2)} = \ddot{\mathbf{a}}^{(2)} + \ddot{\mathbf{b}}^{(2)} \tag{30}$$

$\ddot{\mathbf{a}}^{(1)}$ (respectively, $\ddot{\mathbf{a}}^{(2)}$) represents the part of the accelerations of Subsystem 1 (respectively, Subsystem 2) calculated explicitly from estimated positions and velocities of Subsystem 1 and external forces (respectively, Subsystem 2).

$\ddot{\mathbf{b}}^{(1)}$ (respectively, $\ddot{\mathbf{b}}^{(2)}$) represents the part of the constrained accelerations of Subsystem 1 (respectively, Subsystem 2) implicitly dependent on $\ddot{\mathbf{q}}^{(1)}$ (respectively, $\ddot{\mathbf{q}}^{(2)}$) and calculated by an internal DAE solver.

If we consider now these two subsystems connected together by the following external constraints $\mathbf{\Phi}(\mathbf{q}^{(1)}, \mathbf{q}^{(2)}, t) = \mathbf{0}$ then we have

$$\dot{\mathbf{\Phi}} = \mathbf{\Phi}_{q^{(1)}} \dot{\mathbf{q}}^{(1)} + \mathbf{\Phi}_{q^{(2)}} \dot{\mathbf{q}}^{(2)} + \frac{\partial \mathbf{\Phi}}{\partial t} \tag{31}$$

The associated reaction forces calculated from the Baumgarte/Uzawa formulation is

$$\mathbf{\Phi}_q^T \boldsymbol{\lambda}_{i+1} = \begin{Bmatrix} \mathbf{\Phi}_{q^{(1)}}^T (\boldsymbol{\lambda}_i - \rho \ddot{\tilde{\mathbf{\Phi}}}_i) \\ \mathbf{\Phi}_{q^{(2)}}^T (\boldsymbol{\lambda}_i - \rho \ddot{\tilde{\mathbf{\Phi}}}_i) \end{Bmatrix} \tag{32}$$

Then Eqs. (29) and (30) with the constraint forces are solved iteratively as follows:

$$\text{Subsystem 1:} \quad \ddot{\mathbf{q}}_{i+1}^{(1)} = \ddot{\mathbf{a}}_{i+1}^{(1)} + \ddot{\mathbf{b}}^{(1)} \tag{33}$$

$$\text{Subsystem 2:} \quad \ddot{\mathbf{q}}_{i+1}^{(2)} = \ddot{\mathbf{a}}_{i+1}^{(2)} + \ddot{\mathbf{b}}^{(2)} \tag{34}$$

where

$$\ddot{\mathbf{a}}_{i+1}^{(k)} = \ddot{\mathbf{a}}_i^{(k)} + \mathbf{\Phi}_{q^{(k)}}^T (\boldsymbol{\lambda}_i - \rho \ddot{\mathbf{\Phi}}_i) \quad \text{with } k = 1,2 \tag{35}$$

Since $\ddot{\mathbf{b}}^{(1)}$ (respectively, $\ddot{\mathbf{b}}^{(2)}$) is computed by the internal DAE solver of Subsystem 1 (respectively, Subsystem 2), the two subsystems are completely uncoupled at each iteration.

The feedback gains k_v and k_p associated with the external constraints are calculated following the criteria of constraint stabilization defined in Eq. (12) and depend only on the time step of simulation.

The parameter ρ is formally evaluated by the criteria (26) but it is necessary to build the admittance matrix of constraints, which is not permitted in our modular modeling. In order to overcome this difficulty, we propose to consider ρ as the minimum masse associated with one external constraint among all the others:

$$\rho \leqslant m_\Phi^{(1)} + m_\Phi^{(2)} \quad \text{with } m_\Phi^{(k)} = \inf(\mathbf{\Phi}_{q^{(k)}} \mathbf{M}^{(k)} \mathbf{\Phi}_{q^{(k)}}^T) \quad k = 1,2 \tag{36}$$

In this way, ρ represents the mass or inertia of the two subsystems "viewed" from the joint and can be easily tuned by the operator.

For a system composed of n subsystem components and m joint components, the basic principle of the general algorithm during one time step of simulation can be described as follows:

(1) Predict (or estimate) the $2 \times n$ position and velocity vectors $\dot{\mathbf{q}}^{(n)}, \mathbf{q}^{(n)}$.
(2) Compute independently the n new acceleration vectors $\ddot{\mathbf{q}}^{(n)}$ inside each subsystem component (with internal algebraic constraints if there are any).
(3) Compute independently the m reaction forces inside each joint component from Eq. (32).
(4) Send the m reaction forces to the n subsystem components as external forces.
(5) Test the numerical convergence from Eq. (28) and return to Step (2) if the test is not satisfied.
(6) Correct $\dot{\mathbf{q}}^{(n)}, \mathbf{q}^{(n)}$ in the case of predictor-corrector numerical scheme.
(7) Start a new time step of simulation.

5 Numerical Results

In order to validate the proposed modular approach and highlight its performance, we present in this section the simulation of a double pendulum in plane and we compare the obtained numerical results with ones given by a centralized approach. Then we will simulate a slider-crank mechanism as an example of a kinematic closed loop system to prove the robustness and the flexibility of our modular approach.

In these examples, we use the Newmark implicit numerical scheme with the trapezoidal rule (α=0.25 and β=0.5). Consequently, the position and velocity vectors are calculated in two predictor-corrector steps as follows:

$$\text{Predictor step:} \quad \begin{cases} \mathbf{q} = \mathbf{q}_n + \dot{\mathbf{q}}_n \Delta t + \ddot{\mathbf{q}}_n \Delta t^2/4 \\ \dot{\mathbf{q}} = \dot{\mathbf{q}}_n \Delta t + \ddot{\mathbf{q}}_n \Delta t/2 \end{cases} \tag{37}$$

$$\text{Corrector step:} \quad \begin{cases} \mathbf{q}_{n+1} = \mathbf{q} + \ddot{\mathbf{q}} \Delta t^2/4 \\ \dot{\mathbf{q}}_{n+1} = \dot{\mathbf{q}} + \ddot{\mathbf{q}} \Delta t/2 \\ \ddot{\mathbf{q}}_{n+1} = \ddot{\mathbf{q}} \end{cases} \tag{38}$$

$\Delta t = t_{n+1} - t_n$ is the time step of simulation, and $\ddot{\mathbf{q}}$ is the acceleration vector computed from the governing equations (ODE or DAE).

5.1 Double Pendulum in Plane. The simplest way to model a double pendulum consists of using the relative joint coordinate vector $\mathbf{q}^T = \{\theta_1 \theta_2\}$, as shown in Figure 4.

This modeling leads to the following ODE system:

$$\mathbf{M}\ddot{\mathbf{q}} = \mathbf{F}(\mathbf{q}, \dot{\mathbf{q}}) \tag{39}$$

with

$$\mathbf{M} = \begin{bmatrix} m_1 l_1^2 + m_2 (l_1^2 + l_2^2) + 2 m_2 l_1 l_2 \cos\theta_2 & m_2 l_2^2 + m_2 l_1 l_2 \cos\theta_2 \\ m_2 l_2^2 + m_2 l_1 l_2 \cos\theta_2 & m_2 l_2^2 \end{bmatrix}$$

$$\mathbf{F}(\mathbf{q},\dot{\mathbf{q}}) = \begin{Bmatrix} (2\dot{\theta}_1\dot{\theta}_2 + \dot{\theta}_2^2) l_1 l_2 m_2 \sin\theta_2 - g((m_1+m_2) l_1 \cos\theta_1 + m_2 l_2 \cos(\theta_1+\theta_2)) \\ m_2 l_2^2 + m_2 l_1 l_2 \cos\theta_2 \end{Bmatrix}$$

The mass of the double pendulum is lumped on the connected nodes and the parameter g represents the acceleration quantity of the gravity force. This modeling is centralized because the mass matrix is global and $\ddot{\mathbf{q}}$ is computed directly from the system (39) by a unique solver. With such modeling, it is no longer possible to know the reaction forces between the two links of the double pendulum during the simulation.

Another way of modeling consists of using fully Cartesian coordinates [22], as shown in Fig. 5. $\mathbf{q}^{(1)^T}=\{x_{11}\ y_{11}\ x_{12}\ y_{11}\}$ (respectively, $\mathbf{q}^{(2)^T}=\{x_{21}\ y_{21}\ x_{22}\ y_{22}\}$ represents the fully Cartesian coordinate vector of the first link (respectively, of the second link). The links being assumed to be rigid, we must satisfy the following internal holonomic constraints:

$$\varphi^{(1)} = (x_{11}-x_{12})^2 + (y_{11}-y_{12})^2 - l_1^2 = 0$$

$$\varphi^{(2)} = (x_{21}-y_{22})^2 + (y_{21}-y_{22})^2 - l_2^2 = 0$$

With this type of coordinates, it is possible to have a modular modeling by considering each rigid link as a subsystem component and by considering two additive joint components, each one being defined by two external algebraic constraints as follows:

$$\text{Revolute joint 1} \Rightarrow \mathbf{\Phi}^{(1)} = \begin{Bmatrix} x_{11} \\ y_{11} \end{Bmatrix} = 0$$

$$\text{Revolute joint 2} \Rightarrow \mathbf{\Phi}^{(2)} = \begin{Bmatrix} x_{12}-x_{21} \\ y_{12}-y_{21} \end{Bmatrix} = 0$$

According to the notations defined in the Sec. 4, we have

$$\ddot{\mathbf{a}}_{i+1}^{(1)} = \ddot{\mathbf{a}}_i^{(1)} + \mathbf{\Phi}_{q^{(1)}}^{(1)^T}(\boldsymbol{\lambda}_i^{(1)} - \rho^{(1)}\ddot{\tilde{\mathbf{\Phi}}}_i^{(1)}) + \mathbf{\Phi}_{q^{(1)}}^{(2)^T}(\boldsymbol{\lambda}_i^{(2)} - \rho^{(2)}\ddot{\tilde{\mathbf{\Phi}}}_i^{(2)})$$

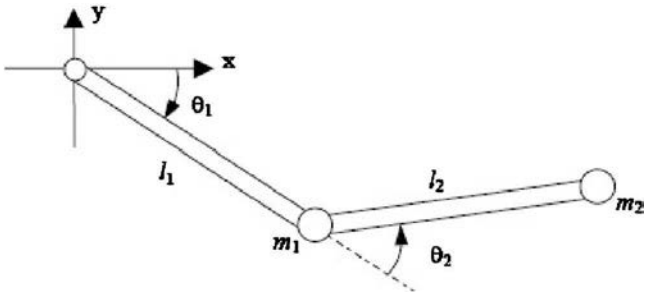

Fig. 4 Double pendulum modeled with relative joint coordinates

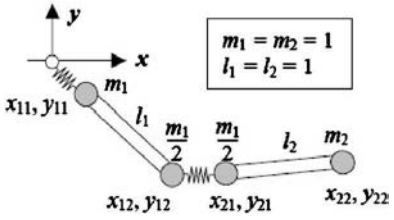

Fig. 5 Double pendulum modeled with fully Cartesian coordinates

$$\ddot{\mathbf{a}}_{i+1}^{(2)} = \ddot{\mathbf{a}}_i^{(2)} + \mathbf{\Phi}_{q^{(2)}}^{(2)^T}(\boldsymbol{\lambda}_i^{(2)} - \rho^{(2)}\ddot{\tilde{\mathbf{\Phi}}}_i^{(2)})$$

In this formulation, the reaction forces inside the revolute joints and the acceleration vectors $\ddot{\mathbf{a}}_{i+1}^{(1)}$, $\ddot{\mathbf{a}}_{i+1}^{(2)}$ are explicitly known at each iteration. The acceleration vector $\mathbf{b}^{(1)}$ (respectively, $\mathbf{b}^{(2)}$) is calculated from an internal DAE solver as an implicit (Bayo formulation) or explicit (Baumgarte formulation) function of $\ddot{\mathbf{a}}_{i+1}^{(1)}$ (respectively, $\ddot{\mathbf{a}}_{i+1}^{(2)}$). In our modular simulations, we have chosen the acceleration-based augmented formulation as the internal solver in each subsystem component.

The same scenario is applied in the two formulations (modular and centralized), which consists of releasing the double pendulum submitted only to the gravity force. The parameters l_1,l_2 are set to 1 m and m_1,m_2 to 1 kg. The initial conditions are similar between the two formulations:

$$\mathbf{q}^T = \{0\ -\pi/2\} \Leftrightarrow \begin{cases} \mathbf{q}^{(1)^T} = \{0\ 0\ 1\ 0\} \\ \mathbf{q}^{(2)^T} = \{1\ 0\ 1\ -1\} \end{cases}$$

and $\dot{\mathbf{q}}=0$, $\dot{\mathbf{q}}^{(1)}=0$, $\dot{\mathbf{q}}^{(2)}=0$. The total simulation time is 10 s and we use two different time steps, $\Delta t=10^{-2}$ s and $\Delta t=10^{-3}$ s in the modular simulation. The numerical precision of all the constraint forces is set to 1% and the solution parameter α is taken to be 10,000 for the internal solver of the two subsystem components. $\rho^{(1)}=1.5$ for the first revolute joint and $\rho^{(2)}=0.4$ for the second one.

Figure 6 shows two trajectories of the mass m_2 in the case of joint coordinates modeling, the first one with $\Delta t=10^{-2}$ s and the second one with $\Delta t=10^{-4}$ s. The numerical scheme used is the implicit Newmark with the trapezoidal rule. The two simulations are very different, which suggests strong nonlinearities and numerical difficulties to have a correct trajectory in such a system. The numerical results do not change really for $\Delta t<10^{-4}$ s, so we consider that the trajectory of the mass m_2 obtained with $\Delta t=10^{-4}$ s will be the referenced trajectory for comparison purposes.

Figure 7 shows the trajectory of the mass m_2 given by the modular modeling (dashed line) with $\Delta t=10^{-2}$. We can notice that this trajectory is close to the referenced one and much better than the one obtained by the joint coordinate modeling for the same time step. Of course, the numerical results are even better when

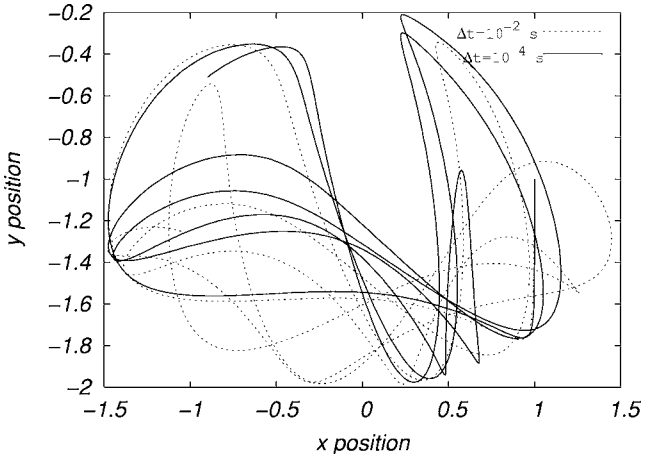

Fig. 6 Trajectory of the mass m_2 (centralized modeling)

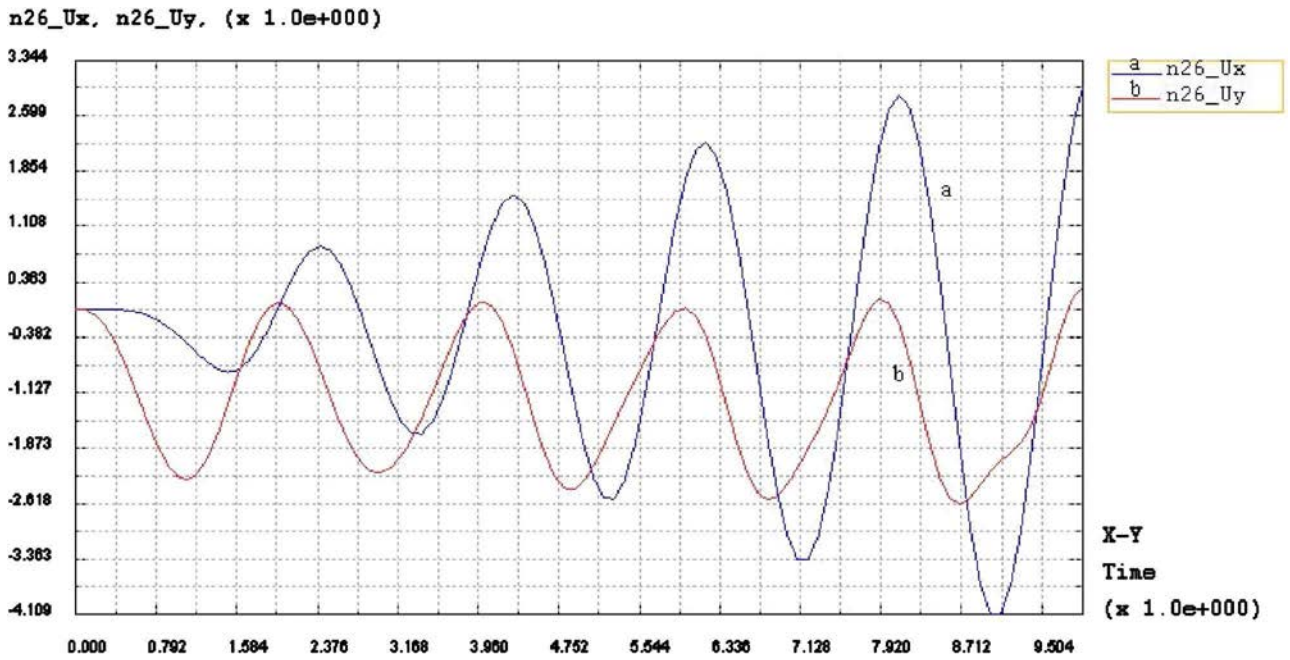

Fig. 7 Comparison between centralized and modular modeling

the time step is reduced to $\Delta t = 10^{-3}$, as we can see in Fig. 8.

CPU times to achieve the different simulations are given in Table 1. They have been obtained on a personal computer (PC) (Pentium 1400 Mhz). Even with a numerical tolerance $\lambda\%$ of only 10%, we obtain a good trajectory of the mass m_2 and the CPU time decreases consequently, as shown in the table. For sake of comparison, we give the CPU time obtained for the simulation of the referenced trajectory (centralized modeling with $\Delta t = 10^{-4}$ s), which is 1.54 s.

Figure 9 shows the graphical interface of FER/MECH software [2,19] in which the double pendulum is simulated in an interactive manner. It is worth noting that, in the deformation scenario, the operator can instantaneously delete the joint between the two rigid bodies by clicking on an interface button without stopping the solution process. The operator is thus incorporated into the solution phase.

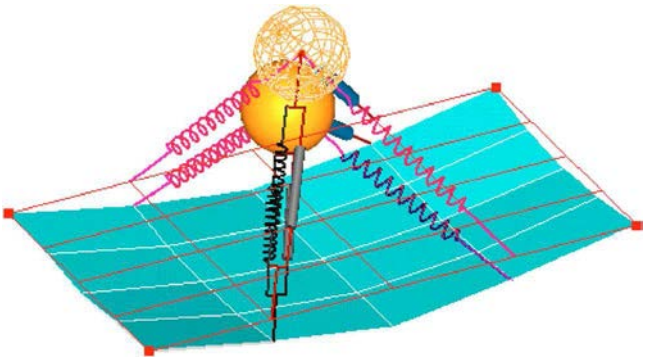

Fig. 8 Comparison between centralized and modular modeling

Table 1 CPU time (s) for the double pendulum in plane

$\lambda\% / \Delta t$	10^{-2} s	10^{-3} s
1%	5 s	47.7 s
10%	2.6 s	25.9 s

5.2 Slider-Crank Mechanism. In the modular approach, the knowledge of the topology of the system is not relevant. The system is decomposed into subsystems, which are directly positioned and oriented relative to the same global reference frame. It is no more difficult to simulate a kinematic open loop system such as a double pendulum, than to simulate a kinematic close loop system such as a slider-crank mechanism.

The slider-crank mechanism is considered in our modular modeling as the double pendulum studied previously in which the mass m_2 is constrained to slide along the x axis. So we must add a prismatic joint as follows:

$$\text{Prismatic joint } 3 \Rightarrow \mathbf{\Phi}^{(3)} = y_{22} = 0$$

Only the acceleration vector $\ddot{\mathbf{a}}_{i+1}^{(2)}$ is modified as follows:

$$\ddot{\mathbf{a}}_{i+1}^{(2)} = \ddot{\mathbf{a}}_{i}^{(2)} + \mathbf{\Phi}_{q^{(2)}}^{(2)T}(\boldsymbol{\lambda}_i^{(2)} - \rho^{(2)}\ddot{\mathbf{\Phi}}_i^{(2)}) + \mathbf{\Phi}_{q^{(2)}}^{(3)T}(\boldsymbol{\lambda}_i^{(3)} - \rho^{(3)}\ddot{\mathbf{\Phi}}_i^{(3)})$$

The scenario consists of releasing the slider-crank mechanism submitted only to the gravity force. The parameters l_1, l_2 are still

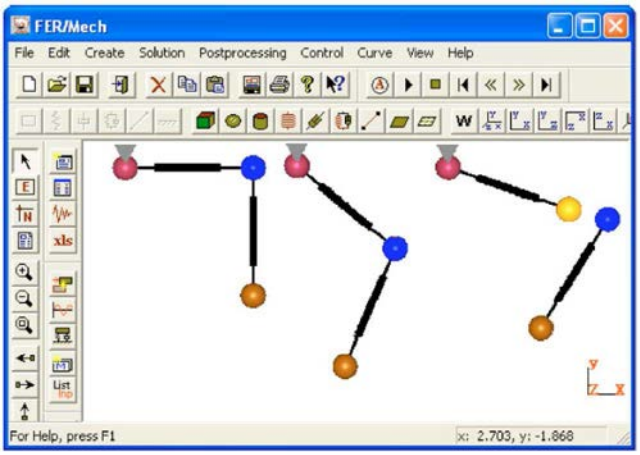

Fig. 9 Interactive simulation of the double pendulum with FER/MECH

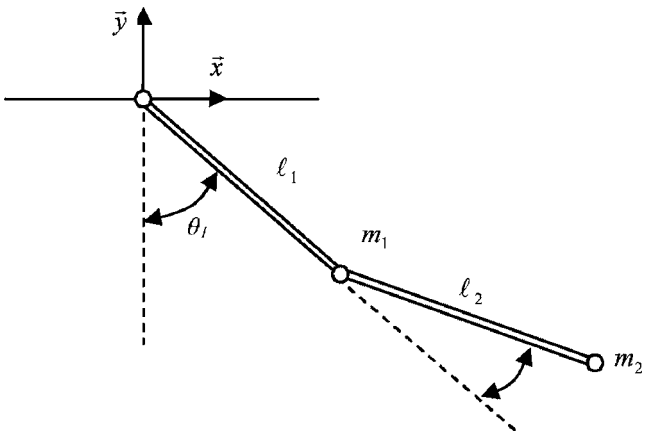

Fig. 10 Trajectory of the mass m_2: $\Delta t = 10^{-2}$ s

set to 1 m and m_1,m_2 to 1 kg. The initial conditions are defined by:

$$\mathbf{q}^T = \{\pi/4 - \pi/4\} \Leftrightarrow \begin{cases} \mathbf{q}^{(1)^T} = \{000.70710.7071\} \\ \mathbf{q}^{(2)^T} = \{0.70710.70711.71421.7142\} \end{cases}$$

Figure 10 shows two curves representing the time dependency of the variable x_{22}, the first one when $\Delta t = 10^{-2}$ and the second one when $\Delta t = 10^{-3}$. As we can see, the two curves are very close and they represent a regular periodic motion in which the mechanical energy of the system is conserved. The maximal constraint error relative to the prismatic joint is

$$\max(y_{22}) = 2.3\ 10^{-7} \quad \text{with } \Delta t = 10^{-3} \text{ s}$$

$$\max(y_{22}) = 5.1\ 10^{-4} \quad \text{with } \Delta t = 10^{-2} \text{ s}$$

The CPU time to achieve the solution is given in the Table 2.

6 Conclusion

We have developed an efficient modular modeling approach of multibody systems using two numerical methods. The first one is based on the acceleration-based augmented Lagrangian formulation with Baumgarte stabilization, which is used to solve a DAE system inside each numerical subsystem component. However, it is possible to use other DAE solvers mentioned in the Introduction. The second one is based on the Uzawa algorithm with Baumgarte stabilization to solve the external constraint forces. We have shown that these two methods converge to the same result and have to satisfy the same criteria to stabilize the acceleration constraint error.

The robustness of the simulation depends on the relative error defined in Eq. (15), which is calculated from the velocity and geometric constraint violations estimated at each time step of simulation. This error is never null and depends on the following factors:

- The precision $O(\Delta t^n)$ of the numerical scheme used to integrate the DAE systems
- The measure Σ of curvature of the manifold H defined in Eq. (15)
- Sudden changes of holonomic constraints (impact, release of contact)

Table 2 CPU time (s) for the slider-crank mechanism

$\lambda\% / \Delta t$	10^{-2} s	10^{-3} s
1%	7.0 s	70.4 s
10%	3.0 s	30.1 s

All these factors contribute to create jumps of velocity that a formulation of DAE systems based on acceleration at discrete time cannot control without leading to serious errors such as an unstable increase of energy during the numerical simulation [23].

In future work, we will propose to adapt a first order formulation of the DAE based on velocity at discrete time and apply it to impact problems [24].

The results presented in this paper are obtained by using the same numerical scheme for the two subsystem components of the double pendulum. Moreover, we assume that the increment time step is constant all along the simulation. In future works, more complex systems will be investigated, in order to show the main interest of this modular method, which is that each component of the system can be considered as a "black box" with its own DAE solver, its own numerical scheme, and its own time step. Flexible multibody systems with changing topology could be studied.

References

[1] Dixit, D. S., Shanbhag, S. H., Mudur, S. P., Isaac, K., and Chinchalkar, S., 1999, "Object-Oriented Design of an Interactive Mechanism Simulation System-Clodion," Comput. Graphics, **23**, pp. 85–94.

[2] Feng, Z. Q., Joli, P., and Séguy, N., 2004, "FER/Mech—A Software With Interactive Graphics for Dynamic Analysis of Multibody System," Adv. Eng. Software, **35**, pp. 1–8.

[3] Schiehlen, W., 2001, "Multibody System Dynamics: Roots and Perspectives," Multibody Syst. Dyn., **1**, pp. 149–188.

[4] Baraff, D., 1996, "Linear-Time Dynamics Using Lagrange Multipliers," *Siggraph 96*, New Orleans, LA, Aug. 4–9.

[5] Wang, J., Ma, Z. D., and Hulbert, G. M., 2003, "A Gluing Algorithm for Distributed Simulation of Multibody Systems," Nonlinear Dyn., **34**(1–2), pp. 159–188.

[6] Tseng, F. C., Ma, Z. D., and Hulbert, G. M., 2003, "Efficient Numerical Solution of Constrained Multibody Dynamics Systems," Comput. Methods Appl. Mech. Eng., **192**, pp. 439–472.

[7] Sauer, J., and Schmer, E., 1998, "A Constraint-Based Approach to Rigid Body Dynamics for Virtual Reality Applications," *ACM Symposium on Virtual Reality Software and Technology*, Taipei, Taiwan, Nov. 2–5.

[8] Gao, S., Wan, H., and Peng, Q., 2000, "An Approach to Solid Modeling in a Semi-Immersive Virtual Environment," Comput. Graphics, **24**, pp. 191–202.

[9] http://www.inria.fr/rapportsactivite/RA2004/siames2

[10] Fahrat, C., Crivelli, L., and Gradin, M., 1995, "Implicit Time Integration of a Class of Constrained Hybrid Formulation. I. Spectral Stability Theory," *34th AIAA Adaptative Structure Forum*, 125, pp. 71–104.

[11] Baumgarte, J. W., 1972, "Stabilization of Constraints and Integrals of Motion in Dynamical Systems," Comput. Methods Appl. Mech. Eng., **1**, pp. 1–16.

[12] Gear, C. W., Leimkuhler, B., and Gupta, G. K., 1985, "Automatic Integration of Euler Lagrange Equations With Constraints," J. Comput. Appl. Math., **12**, pp. 77–90.

[13] Bayo, E., and Avello, A., 1994, "Singularity-Free Augmented Lagrangian Algorithms for Constrained Multibody Dynamics," Nonlinear Dyn., **5**, pp. 209–231.

[14] Shabana, A., 1994, *Computational Dynamics*, Wiley, New York.

[15] Schiehlen, W., 1990, *Multibody System Handbook*, Springer, Berlin.

[16] Wehage, R., and Haug, E., 1982, "Generalized Coordinate Partitioning for Dimension Reduction in Analysis of Constrained Dynamic Systems," ASME J. Mech. Des., **104**, pp. 245–255.

[17] Minh Tran, D., 1991, "Equations of Motion Multibody Systems in the ESA-MIDAS Software," *International Conference on Spacecraft Structures and Mechanical Testing ESTEC*, Noordwijk, Sweden, Apr. 24–26.

[18] Blajer, W., 2002, "Augmented Lagrangian Formulation: Geometrical Interpretation and Application to Systems With Singularities and Redundancy," Multibody Syst. Dyn., **8**, pp. 141–159.

[19] Feng, Z.-Q., http://gmfe16.cemif.univ-evry.fr:8080/~feng/FerMech.html

[20] Brenan, K. E., Campbell, S. L., and Petzold, L. R., 1989, *The Numerical Solution of Initial Value Problems in Differential-Algebraic Equations*, Elsevier Science, New York.

[21] Uzawa, H., Anow, K., and Hurwicz, L., 1958, *Studies in Linear and Non linear Programming*, Stanford University Press, Stanford.

[22] Nikravesh, P. E., and Attia, H. A., 1994, "Construction of the Equations of Motion for Multibody Dynamics Using Point and Joint Coordinates," *Computer-Aided Analysis of Rigid and Flexible Mechanical System*, *NATO ASI, Series E: Applied Sciences* Vol. 268, Kluwer Academic, Dordrecht, pp. 31–60.

[23] Laursen, T. A., 2002, *Computational Contact and Impact Mechanics: Fundamentals of Modeling Interfacial Phenomena in Nonlinear Finite Element Analysis*, Springer, New York.

[24] Feng, Z.-Q., Joli, P., Cros, J.-M., and Magnain, B., 2005, "The Bi-Potential Method Applied to the Modeling of Dynamic Problems With Friction," Comput. Mech., **36**, pp. 375–383.