DAE time integration for real-time applications in multi-body dynamics

Bernhard Burgermeister [a,*], Martin Arnold [a], Benjamin Esterl [b]

[a] *Martin–Luther–University Halle–Wittenberg, Department of Mathematics and Computer Science, Institute of Numerical Mathematics, D - 06099 Halle (Saale), Germany*

[b] *TESIS DYNAware GmbH, Baierbrunnerstr. 15, D-81379 München, Germany*

Abstract

The equations of motion of multi-body systems with kinematically closed loops are given by differential-algebraic equations (DAE). Real-time applications like hardware-in-the-loop testbeds or driving simulators require appropriate integration methods that can solve the non-linear constraints without exceeding an a priori fixed number of calculation steps.

Partitioned linear-implicit Euler methods can be extended by appropriate stabilization techniques to keep the error in the constraints limited for arbitrary time intervals. These methods need a fixed low number of operations for each time step. One simplified Newton step for the projection onto the constraint manifold can prevent the drift-off for arbitrary time intervals if the stepsize is sufficiently small.

Selected methods were successfully implemented in the commercial vehicle simulation package veDYNA to improve substantially the simulation capabilities for vehicle-trailer coupling.

Key words: Differential-algebraic equations, time integration, real-time, multi-body dynamics

1 Real-time simulation of multi-body systems

Computer simulations are an important tool for the design and optimization of mechanical systems like robots, cars and planes. Mechanical multi-body system models reproduce the dynamical behavior of mechanical systems in great detail [16].

* Corresponding author.
Email address: bernhard.burgermeister@mathematik.uni-halle.de (Bernhard Burgermeister).

A multi-body system (MBS) consists of a finite number of rigid or flexible bodies which are connected by coupling elements like joints, springs, dampers and actuators. The coupling elements are assumed to be massless, so that the mass of the mechanical system is concentrated in the bodies. Industrial simulation packages like SIMPACK [21] or ADAMS [18] provide an interface for modeling and numerical simulation of multi-body systems [23].

The equations of motion of a mechanical multi-body system are given by a second order system of ordinary differential equations (ODE)

$$M(q)\ddot{q} = f(q,\dot{q}) \qquad \Leftrightarrow \qquad \begin{cases} \dot{q} = u \\ \dot{u} = \tilde{f}(q,u) = M^{-1}(q) f(q,u). \end{cases} \tag{1}$$

If the mechanical system contains closed loops, additional algebraic constraints have to be introduced and the resulting equations of motion are given by a differential-algebraic equation (DAE) that may be written in first order form as [20]

$$\dot{q} = u, \tag{2a}$$
$$M(q)\dot{u} = f(q,u) - G^T(q)\lambda, \tag{2b}$$
$$0 = g(q) \tag{2c}$$

where q are the generalized coordinates, u the generalized velocities, $M(q)$ the positive definite mass matrix, $g(q) = 0$ the algebraic constraints, λ the Lagrange multipliers and $G(q) = \partial g(q)/\partial q$ the constraint matrix that is given by the Jacobian of $g(q)$. We suppose that $G(q)$ has full rank to exclude redundant constraints (*Grübler condition*) [9].

Real-time applications like simulators or hardware-in-the-loop test facilities have special demands on the utilized integration methods [6,22]. For hardware-in-the-loop tests a module of a larger system, for example the anti-skid system of a car, is embedded in a simulation environment that simulates the remaining parts of the car, the driver and the road. So the anti-skid system can be tested without time consuming and dangerous test runs with a real car.

The embedded hardware module has a fixed time scale, so the simulation has to be as fast as the mechanical system that is replaced by the simulation. Usually this is achieved by using a fixed small time stepsize (often in the range of 1.0ms) and exchanging data between the hardware module and the simulation at the beginning of each time step.

The complete calculation of one time-step has to be finished within this fixed period of time. So it is not possible to use iterative methods and techniques like automatic stepsize control which are commonly used in off-line simulation to improve the efficiency of the integration method.

The accuracy requirements on the time integration methods are quite low, because mechanical systems are usually designed to be stable or are embedded in stable control circuits that compensate errors in the time integration. Because of the required very small computing times only low order methods are used, in most cases the first order Euler method.

It is well known, that the classical *explicit* Euler method suffers from numerical instability in the application to stiff systems like vehicle models with stiff suspension elements or strongly damped components [19]. To avoid the iterative solution of nonlinear equations that is typical of implicit time integration methods for stiff systems, we consider in the present paper the *linear-implicit* Euler method [15,19] that may be interpreted as an implementation of the implicit Euler method with an a priori fixed number of Newton steps in the iterative solution of the arising systems of nonlinear equations.

In the unconstrained case the linear-implicit Euler method requires in each time step the solution of a system of linear equations $(I - hJ)x = b$ with h denoting the time stepsize and an approximation J of the Jacobian of the right hand side in (1). The choice of J is essential for the efficiency and for the numerical stability of the linear-implicit method: J should be a cheap approximation of the Jacobian containing all terms that cause stiffness in the system. On the other hand, all terms of the Jacobian that correspond to non-stiff components may be neglected (*partitioned* linear-implicit method). The partitioning may be based on physical considerations [19] or on numerical criteria [24,25].

In the present paper, we study the stability of the linear-implicit Euler method for different partitioning strategies and extend the analysis of low order linear-implicit methods to constrained systems (2).

The remaining part of the paper is organized as follows: In Section 2, the linear stability analysis based on Dahlquist's classical test equation $\dot{y} = \lambda y$ is extended to second order ODEs (1). Stability regions and the maximum stepsize h in the stiff case depend strongly on the structure of the Jacobian approximation J.

The main focus of the present work is on constrained systems (2) that are considered in more detail in Section 3. The key result of this section is a rigorous proof that one Newton step of a projection method is sufficient to guarantee that the residuum $\|g(q_n)\|$ in the constraints (2c) remains bounded on arbitrary long time intervals and the *drift-off effect* [3,15] is completely avoided for many problems.

Based on these theoretical investigations the partitioned linear-implicit methods were implemented in an industrial vehicle simulation package [7,8]. Section 4 summarizes test results for an academic test problem, for a classical numerical benchmark problem and for the application to the simulation of vehicle-trailer coupling in the industrial vehicle simulation package veDYNA.

2 Partitioned linear-implicit Euler method

Frequently, vehicle models contain stiff springs or components with strong damping. Therefore, the equations of motion can contain stiff components and explicit time integration methods get ineffi cient. Explicit methods need the lowest numerical effort for each integration step, but are not stable for stiff systems or require very small step sizes [15, Section IV.2]. On the other hand, implicit methods can have excellent stability properties, but require the iterative solution of nonlinear equations at each time-step and hence the number of computation steps and the computing time are not known in advance.

The class of linear-implicit Runge-Kutta methods combines positive properties of explicit and implicit methods [15, Section IV.7]. They have good stability properties and need only the solution of *linear* systems of equations at each time step. We will focus on the linear-implicit Euler method

$$y_{n+1} = y_n + h(I - hJ)^{-1}\hat{f}(y_n) \quad \text{for} \quad \dot{y}(t) = \hat{f}(y(t))$$

since it requires the lowest numerical effort at each time step and is well suited for large classes of stiff problems.

The linear-implicit Euler method belongs to the class of *W-methods* [15, Section IV.7] which do not need the exact Jacobian matrix J of the right hand side $\hat{f}$ for convergence. This allows us to partition the equations of motion and hence improve the performance of the integration method.

The calculation of the Lagrange multipliers λ in (2) from the constraints (2c) will be the subject of the Section 3. If λ is given, the equations of motion (2) can be written as ordinary differential equation:

$$\begin{aligned} \dot{q} &= u, \\ \dot{u} &= \tilde{f}(q,u) = M^{-1}f(q,u) - M^{-1}G^T(q)\lambda. \end{aligned}$$

For this system, one step of the linear-implicit Euler method is given by

$$(I - hJ)\begin{pmatrix} \Delta q \\ \Delta u \end{pmatrix} = \begin{pmatrix} u \\ \tilde{f}(q,u) \end{pmatrix}, \qquad \begin{pmatrix} q_1 \\ u_1 \end{pmatrix} = \begin{pmatrix} q_0 \\ u_0 \end{pmatrix} + h\begin{pmatrix} \Delta q \\ \Delta u \end{pmatrix} \tag{3}$$

with the exact Jacobian

$$J = J_{\text{exact}} := \begin{pmatrix} 0 & I \\ \tilde{f}_q(q,u) & \tilde{f}_u(q,u) \end{pmatrix}.$$

Without knowledge of the structure of $\tilde{f}$ we can partition the system only if we omit the identity matrix in the upper right block of J. As the matrix J is important for the stability of the method for stiff systems we consider the linear second order test equation

$$\ddot{q} = \tilde{f}(q, \dot{q}) = -aq - b\dot{q}, \quad (q(t), a, b \in \mathbb{R}, a \geq 0, b \geq 0)$$

and transform this system to a first order system

$$\begin{aligned} \dot{q} &= u, \\ \dot{u} &= \tilde{f}(q, u) = -aq - bu. \end{aligned}$$

With the exact Jacobian matrix

$$J_{\text{exact}} = \begin{pmatrix} 0 & I \\ \tilde{f}_q & \tilde{f}_u \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -a & -b \end{pmatrix}$$

of the test problem the linear-implicit Euler method is stable for all $a, b \geq 0$ at any stepsize. This method requires, however, the evaluation of $\tilde{f}_q, \tilde{f}_u$ and the solution of a system of $2n_q$ linear equations in each time step.

If we set the upper right block to zero

$$J = J_1 := \begin{pmatrix} 0 & 0 \\ \tilde{f}_q & \tilde{f}_u \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ -a & -b \end{pmatrix},$$

we can calculate $\Delta q = u_0$ by the *explicit* Euler method and Δu by solving a system of linear equations of size n_q instead of $2n_q$. The explicit treatment of q has a severe effect on the stability of the linear-implicit Euler method. Now, the coefficients $a,\ b$ and the stepsize h are limited by $0 \leq hb$ and $0 \leq h^2 a \leq 2hb + 4$.

Comparison of the Taylor series expansion of the numerical solution shows that the stability of the linear-implicit Euler method with $J = J_{\text{exact}}$ can be restored if we modify the lower right block of the stabilization matrix:

$$J = J_2 := \begin{pmatrix} 0 & 0 \\ \tilde{f}_q & \tilde{f}_u + h\tilde{f}_q \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ -a & -b - ha \end{pmatrix}.$$

This matrix allows stable integration of the test equation for any $a, b \geq 0$ and any stepsize h.

Depending on the model this full stabilization is not always necessary. One important case are moderately oscillating system with low or no damping. It is well

known that the explicit Euler method is inappropriate for such systems. The stabilization matrix

$$J = J_3 := \begin{pmatrix} 0 & 0 \\ \tilde{f}_q & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ -a & 0 \end{pmatrix}$$

allows stable integration for $0 \leq hb \leq 2$ and $0 \leq h^2 a \leq 4 - 2hb$. The main advantage of this stabilization matrix is, that no full Jacobian matrix of $\tilde{f}$ has to be calculated. Only one directional derivative $\tilde{f}_q u$ is needed. This method is closely related to the half-implicit Euler method of Gipser [11].

If we plot the eigenvalues e_F of the right hand side for stable parameters a and b of the test equation we get the stability domains shown in Figure 1.

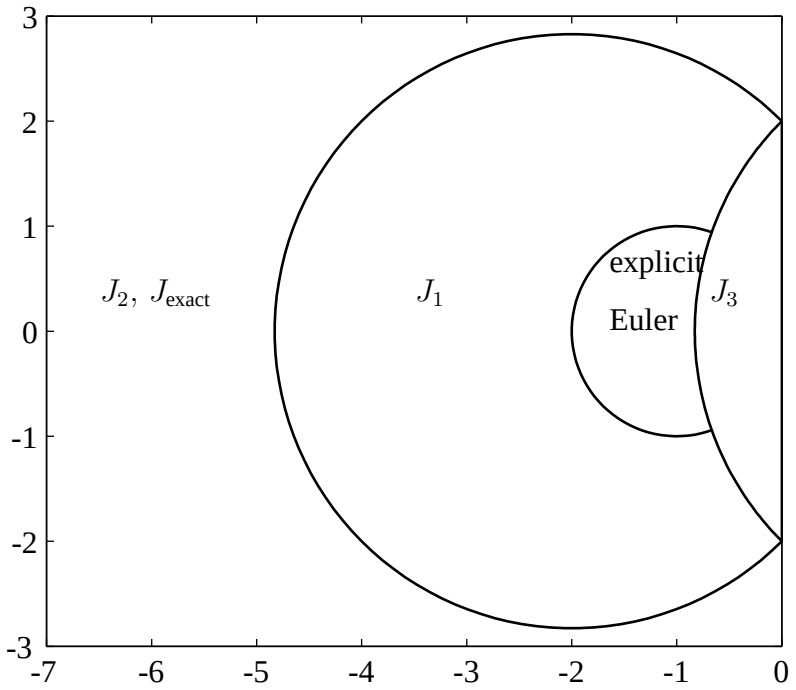

Fig. 1. Stability domain of the partitioned linear-implict Euler method for different matrices J.

Though this scalar second order test equation cannot give a proof of the stability for larger systems [15, p. 116], it shows at least some necessary entries of the stabilization matrix J.

3 Linear-implicit methods for constrained mechanical systems

In implicit time integration methods, the consideration of constraints is straightforward [4]. Furthermore, there are implicit methods with excellent stability proper-

ties for stiff equations. However, for real-time applications implicit methods are not suitable since systems of nonlinear equations have to be solved iteratively in each time step.

The results of Section 2 show that similar stability properties may be achieved by linear-implicit methods without any iterative algorithms. For constrained systems, we cannot expect that a non-iterative method solves nonlinear constraints exactly.

In the present section, we discuss several ways to extend the linear-implicit Euler method of Section 2 to a non-iterative method for constrained systems and prove that the error in the constraints $g(q) = 0$ is bounded by $\|g(q_n)\| \leq Ch^r$ with $r \geq 2$ and a constant C that is independent of the length of the time interval.

3.1 Index reduction and drift-off effect

The equations of motion (2) are given by a differential-algebraic system of index 3. To avoid numerical problems which make the direct time integration of index-3 systems difficult, the system is transformed to a new system with lower index by differentiation of the constraints [4,9,15]:

$$0 = g(q), \tag{5a}$$

$$0 = \frac{\mathrm{d}g(q)}{\mathrm{d}t} = g_q(q)\dot{q} = G(q)u, \tag{5b}$$

$$0 = \frac{\mathrm{d}^2 g(q)}{\mathrm{d}t^2} = g_{qq}(q)(u,u) + G(q)\dot{u}. \tag{5c}$$

If we substitute in (2) the original constraints (5a) on position level by their second derivative (5c) on acceleration level we get the so called *index-1 formulation*. Then we can insert the differential equation (2b) into (5c) and calculate the Lagrange multipliers λ by solving the linear system of equations

$$\lambda = (G(q)M^{-1}G^T(q))^{-1}(g_{qq}(q)(u,u) + G(q)M^{-1}f(q,u)).$$

Accordingly we get the *index-2 formulation* by using the first derivative (5b) on velocity level of the constraints. Because of the special structure of the equations of motion we can insert the formula for the integration step of u_{n+1}, see (3), in (5b) and choose λ_n so that $G(q_{n+1})u_{n+1} = 0$ holds:

$$q_{n+1} = q_n + hu_n, \tag{6a}$$

$$\begin{pmatrix} M - hJ_u & G^T(q_n) \\ G(q_{n+1}) & 0 \end{pmatrix} \begin{pmatrix} u_{n+1} - u_n \\ \lambda_n \end{pmatrix} = h \begin{pmatrix} f + hJ_q u_n \\ -G(q_{n+1})u_n \end{pmatrix}. \tag{6b}$$

Here, J_q and J_u denote the lower blocks in the 2×2 block matrix J. The main advantages of this linear-implicit Euler method are that the velocity constraints (5b) are solved exactly and there is no need for the second derivative of $g(q)$. It may be considered as a drawback of this approach that the matrix of (6b) is no longer symmetric and the order of the calculation of q_{n+1} and u_{n+1} is fixed.

The numerical solution of an index reduced system can leave the manifold

$$\{q : g(q) = 0\}$$

defined by the original constraints (2c) because of discretization and round-off errors (*drift-off effect*). To avoid the drift, we have to apply additional stabilization techniques to keep the error in the constraints limited. Otherwise the error in the position constraints (5a) can grow linearly in time for methods that are based on the index-2 formulation (*index-2 methods*) or up to quadratically in time for methods that are based on the index-1 formulation (*index-1 methods*) [15, Theorem VII.2.1].

3.2 Baumgarte stabilization

The first method for stabilization of the constraints was proposed by Baumgarte [3]: he substituted the constraints on acceleration level (5c) by a linear combination of (5a)-(5c):

$$0 = \frac{\mathrm{d}^2}{\mathrm{d}t^2} g + 2\alpha \frac{\mathrm{d}}{\mathrm{d}t} g + \beta g = g_{qq}(q)(u,u) + G(q)\dot{u} + 2\alpha G(q)u + \beta g(q).$$

The scalar constants α and β are called *Baumgarte parameters*. These parameters have to be chosen so that the scalar differential equation $0 = \ddot{y} + 2\alpha\dot{y} + \beta y$ has an asymptotically stable solution $y(t) = c_1 e^{\tau_1 t} + c_2 e^{\tau_2 t}$. This is equivalent to the condition that $\tau_{1,2} = -\alpha \pm \sqrt{\alpha^2 - \beta}$ have both negative real part.

The quality of the stabilization depends on the parameters α and β. If we choose them too small, the stabilization is not sufficient. Too large parameters introduce additional stiffness to the system and make the numerical integration inefficient and sensitive to round-off errors [14].

To find the optimal parameters, we abbreviate the errors in the position and velocity constraints in the i-th step by $w_0 := g(q_i)$ and $v_0 := G(q_i)u_i$. Beginning with sufficiently small initial values w_0 and v_0 we get for the errors after one and two time steps:

$$
\begin{aligned}
w_1 &= w_0 + hv_0 + \mathcal{O}(h^2),\\
v_1 &= -h\beta w_0 + (1 - 2h\alpha)v_0 + \mathcal{O}(h^2),\\
w_2 &= (1 - h^2\beta)w_0 + 2h(1 - h\alpha)v_0 + \mathcal{O}(h^2),\\
v_2 &= 2h\beta(h\alpha - 1)w_0 + (4h\alpha(h\alpha - 1) + 1 - h^2\beta)v_0 + \mathcal{O}(h^2) + (\alpha + \beta)\mathcal{O}(h^3).
\end{aligned}
$$

For $\alpha = 1/h$ and $\beta = 1/h^2$ the influence of w_0 and v_0 on w_2 and v_2 is annihilated in first order. On the other hand the large Baumgarte parameter $\beta = 1/h^2$ introduces artificial stiffness to the system and adds a large error term to the velocity constraints. The expected errors in the position constraints are of magnitude $\mathcal{O}(h^2)$ and $\mathcal{O}(h)$ in the velocity constraints. To avoid problems with the additional stiffness, the Baumgarte parameters are usually chosen smaller: $0 < \alpha < 1/h$ and $0 < \beta < 1/h^2$, see [1]. Then the influence of w_i and v_i on w_{i+2} and v_{i+2} is damped by the factors $1 - h\alpha < 1$ and $1 - h^2\beta < 1$.

Baumgarte stabilization can be applied to the index-2 formulation too [6,22]. Here, the position constraints (5a) are multiplied by one Baumgarte parameter α and added to the velocity constraints (5b):

$$
0 = \frac{\mathrm{d}}{\mathrm{d}t} g + \alpha g = G(q)u + \alpha g(q).
$$

The error in the position constraint after one time step is

$$
g(q_{n+1}) = g(q_n) + hG(q_n)u_n + \mathcal{O}(h^2) = (1 - h\alpha)g(q_n) + \mathcal{O}(h^2)
$$

which is independent of $g(q_n)$ in first order for $\alpha = 1/h$ and has the magnitude of $\mathcal{O}(h^2)$. Again this introduces a large error of size $\mathcal{O}(h)$ to the velocity constraints.

3.3 Iterative projection methods

A common method for the stabilization of constraints is the stabilization by projection. This projection can be added to almost any integration methods for differential-algebraic systems.

After each integration step of the index-reduced system there is usually a small error in the constraints $g(\tilde{q}_{n+1}) \neq 0$ with $\tilde{q}_{n+1}$ denoting the numerical solution after one step. To satisfy the constraints, the numerical solution can be projected to the manifold defined by $g(q) = 0$ by solving the following minimization problem [17]:

$$
\min_{q_{n+1} \in \mathbb{R}^{n_q}} \left\| q_{n+1} - \tilde{q}_{n+1} \right\|_M \quad \text{with} \quad 0 = g(q_{n+1}).
$$

The distance is measured in the M-Norm $\|x\|_M = \sqrt{x^T M x}$ which is induced by the positive definite mass matrix M. In the norm $\|\cdot\|_M$, the weighting factors for

displacements of heavy bodies are larger than the ones for displacements of light bodies.

To solve the minimization problem, the constraints are coupled to the target function by Lagrange parameters μ. That gives the necessary conditions for a minimum [15, Section VII.2]:

$$M(q_{n+1})(q_{n+1} - \tilde{q}_{n+1}) + G^T(q_{n+1})\mu = 0, \tag{7a}$$
$$g(q_{n+1}) = 0. \tag{7b}$$

Normally, this nonlinear system has to be solved by a Newton like method, but for real-time applications we must not use an iterative method, so we can only perform a limited number of Newton steps to reduce the error in the constraints.

The error in the velocity constraints can be corrected in a similar way by projection of the velocity $\tilde{u}_{n+1}$ on the manifold defi ned by $G(q_{n+1})u = 0$:

$$\min_{u_{n+1} \in \mathbb{R}^{n_u}} \|u_{n+1} - \tilde{u}_{n+1}\|_M \quad \text{with} \quad 0 = G(q_{n+1})u_{n+1}.$$

The necessary conditions for this minimization problem are given by the *linear* equation

$$\begin{aligned} M(q_{n+1})(u_{n+1} - \tilde{u}_{n+1}) + G^T(q_{n+1})\mu &= 0, \\ G(q_{n+1})u_{n+1} &= 0 \end{aligned}$$

which can be solved exactly without iterative methods.

The projection of the velocities u has to be done after the projection of the positions q because the velocity projection depends on the value of q_{n+1} and changes only u_{n+1} (*sequential position velocity projection*). If the individual calculations of the integration step and the projections are interleaved in an appropriate order, only very few additional evaluations of the multi-body formalisms are necessary [15, Section VII.2].

As an alternative to sequential position velocity projection, Gear et al. [10] proposed the simultaneous consideration of position and velocity constraints in the index-2 formulation (*Gear-Gupta-Leimkuhler formulation*):

$$M\dot{q} = Mu - G^T(q)\mu, \tag{9a}$$
$$M\dot{u} = f(q,u) - G^T(q)\lambda, \tag{9b}$$
$$0 = g(q), \tag{9c}$$
$$0 = G(q)u. \tag{9d}$$

This system is a DAE of index 2 that contains the position and velocity constraints. Additional Lagrange multipliers μ are introduced that vanish for the exact solution [10].

The explicit Euler step for (9a) replaces the first stage of method (6):

$$q_{n+1} = q_n + hu_n - hM^{-1}G^T\mu_n.$$

The Lagrange multipliers μ_n are determined implicitly by the position constraint (9c) at $t = t_{n+1}$. With $\tilde{q}_{n+1} = q_n + hu_n$ this nonlinear system is identical to (7a) for the projection of the position coordinates. After that, u_{n+1} can be calculated exactly like for the unstabilized index-2 method (6b), so that the velocity constraints (9d) are solved exactly at $t = t_{n+1}$.

3.4 Non-iterative projection methods

Practical experience shows, that for real-time applications only one step of the Newton iteration for the projection of the position coordinates is sufficient for stabilization. In this section we will give a rigorous proof that this strategy guarantees that the error in the constraints remains limited for arbitrary time intervals [5]:

To calculate the propagation of the error in the position constraints we have to examine the following steps:

$$(q_n, u_n) \xrightarrow{\text{integrator}} (\widetilde{q}_{n+1}, \widetilde{u}_{n+1}) \xrightarrow{q\text{-projection}} (q_{n+1}, \widetilde{u}_{n+1}) \xrightarrow{u\text{-projection}} (q_{n+1}, u_{n+1})$$

We can assume $G(q_n)u_n = 0$ because of the u-projection in the preceding time step $t_{n-1} \to t_n$. The error in the position constraints after one integration step with the explicit Euler method can be estimated by a Taylor series expansion:

$$\begin{aligned} g(\widetilde{q}_{n+1}) &= g(q_n + hu_n) = g(q_n) + hG(q_n)u_n + h^2R_{1,n} \\ &= g(q_n) + h^2R_{1,n} \end{aligned} \tag{10}$$

with the remainder term

$$R_{1,n} := \int_0^1 (1-\zeta)g_{qq}(q_n + \zeta hu_n)(u_n, u_n)\,\mathrm{d}\zeta.$$

The projection step requires the solution of the linear system of equations

$$\begin{pmatrix} M(q_n) & G^T(q_n) \\ G(q_n) & 0 \end{pmatrix} \begin{pmatrix} \Delta q_n \\ \Delta\mu_n \end{pmatrix} = \begin{pmatrix} 0 \\ g(\widetilde{q}_{n+1}) \end{pmatrix} \tag{11}$$

which we can write with the help of the inverse

$$\begin{pmatrix} M & G^T \\ G & 0 \end{pmatrix}^{-1} = \begin{pmatrix} * & M^{-1}G^T(GM^{-1}G^T)^{-1} \\ * & * \end{pmatrix}$$

of the so-called optimization matrix, see [13], as

$$\Delta q_n = P_n g(\widetilde{q}_{n+1}), \quad G(q_n)\Delta q_n = g(\widetilde{q}_{n+1}) \tag{12}$$

with

$$P_n := M^{-1}G^T(GM^{-1}G^T)^{-1}|_{q_n}.$$

By Taylor series expansion we can estimate the error in the position constraint for the new position coordinates $q_{n+1} = \widetilde{q}_{n+1} - \Delta q_n$:

$$\begin{aligned} g(q_{n+1}) &= g(\widetilde{q}_{n+1}) - G(\widetilde{q}_{n+1})\Delta q_n + R_{2,n}(\Delta q_n, \Delta q_n) \\ &= (G(q_n) - G(\widetilde{q}_{n+1}))\Delta q_n + R_{2,n}(\Delta q_n, \Delta q_n) \\ &= -hR_{3,n}(u_n, \Delta q_n) + R_{2,n}(\Delta q_n, \Delta q_n) \end{aligned}$$

with the remainder terms

$$\begin{aligned} R_{2,n}(y_1, y_2) &:= \int_0^1 (1-\zeta) g_{qq}(\widetilde{q}_{n+1} + \zeta \Delta q_n)(y_1, y_2)\, \mathrm{d}\zeta, \\ R_{3,n}(y_1, y_2) &:= \int_0^1 g_{qq}(\widetilde{q}_{n+1} - \zeta h u_n)(y_1, y_2)\, \mathrm{d}\zeta. \end{aligned}$$

Together with (12) and (10) this gives the quadratic form in $g(q_n)$

$$\begin{aligned} g(q_{n+1}) =& R_{2,n}(P_n g(q_n), P_n g(q_n)) \\ &- hR_{3,n}(u_n, P_n g(q_n)) + 2h^2 R_{2,n}(P_n R_{1,n}, P_n g(q_n)) \\ &- h^3 R_{3,n}(u_n, P_n R_{1,n}) + h^4 R_{2,n}(P_n R_{1,n}, P_n R_{1,n}). \end{aligned}$$

Then the error in the position constraints is limited by

$$\|g(q_{n+1})\| \leq C_{2,n}\|g(q_n)\|^2 + hC_{1,n}\|g(q_n)\| + h^3 C_{0,n}$$

with the abbreviations

$$\begin{aligned} A_{1,n} &:= \tfrac{1}{2} \max_{\xi \in \mathcal{U}_n} \|g_{qq}(\xi)(u_n, u_n)\| \\ A_{2,n} &:= \tfrac{1}{2} \max_{\xi \in \mathcal{U}_n} \|g_{qq}(\xi)((P_n(\cdot), P_n(\cdot))\| \\ A_{3,n} &:= \max_{\xi \in \mathcal{U}_n} \|g_{qq}(\xi)(u_n, P_n(\cdot))\| \\ C_{2,n} &:= A_2 \\ C_{1,n} &:= A_3 + hA_2A_1 \\ C_{0,n} &:= A_3A_1 + hA_2A_1^2 \end{aligned}$$

in a neighborhood $\mathcal{U}_n$ of the exact solution which contains q_n and the unprojected numerical solution $\tilde{q}_{n+1}$.

If the constants

$$C_0 := \max_{n=0,1,\ldots,N} C_{0,n}, \quad C_1 := \max_{n=0,1,\ldots,N} C_{1,n}, \quad C_2 := \max_{n=0,1,\ldots,N} C_{2,n}$$

fulfill the conditions

$$hC_1 < 1 \quad \text{and} \quad (1 - hC_1)^2 - 4h^3 C_0 C_2 > 0, \tag{13}$$

then $\|g(q_n)\|$ is limited by

$$\|g(q_n)\| \leq g_{\text{lim}} := \frac{1 - hC_1 - \sqrt{(1 - hC_1)^2 - 4h^3 C_0 C_2}}{2C_2} \leq Ch^3$$

with a suitable constant C that is independent of $n,\ n = 0, 1, \ldots, N$.

Proof. The scalar function $\Phi(g) = C_2 g^2 + hC_1 g + h^3 C_0$ has the attractive fixed point g_{lim} with $0 < \Phi'(g) < 1$ and $\Phi(g) < g$ for $0 \leq g \leq g_{\text{lim}}$. So any iteration $g_{n+1} = \Phi(g_n),\ (n = 0, 1, 2, \ldots)$ with $0 \leq g_0 < g_{\text{lim}}$ converges to g_{lim} with $g_n < g_{\text{lim}}$ for any n, see [5] for more details. □

In practical applications, the constants C_2, C_1 and C_0 remain bounded on arbitrary time intervals because velocities and forces are limited in real systems. For sufficiently small step sizes h, the conditions (13) are always satisfied. For real-time applications the stepsize h is fixed, so this gives a sufficient condition to guarantee that the error in the constraints remains bounded by Ch^3 if just *one* projection step is performed per time step. Similar results may be obtained for the sequential regularization methods of Ascher and Lin [2], see [5].

4 Numerical experiments

In the present section we summarize numerical results for three test problems of increasing complexity. The drift-off effect in the unstabilized time integration of an academic test problem underlines the need for stabilization methods in the long-term simulation of constrained systems, see Section 4.1. A detailed comparison of various stabilization methods applied to a simplified car axle model is given in Section 4.2. Finally, one of the stabilization methods was tested successfully in a developer version of an industrial vehicle simulation tool, see Section 4.3.

4.1 Circular track

To test the long-term stability of the constraints we created a test scenario where a point mass rotates in a plane around a fi xed center point, see Fig. 2. The position of the mass is given by its Cartesian coordinates $q := (x_1, x_2)^T$. The distance between the mass and the center is fi xed to 1 by an algebraic constraint $g(q) := x_1^2 + x_2^2 - 1$. A force F_{PI} tangential to the path controls the speed of the mass by a PI-controller and a force F_{R} orthogonal to the path pushes the mass away from the center in addition to the centrifugal force.

This test problem is a closed control circuit which compensates integration errors that change the speed of the mass, but emphasizes the error by the drift-off. Fig. 3 shows the drift for the unstabilized index-1 method, the unstabilized index-2 method and the fully stabilized index-1 method with projection of the velocities u and displacements q. As expected, the distance of the mass to the center diverges from 1 for the index-reduced methods without stabilization, and the results for the index-1 method diverge faster than the ones for the index-2 methods.

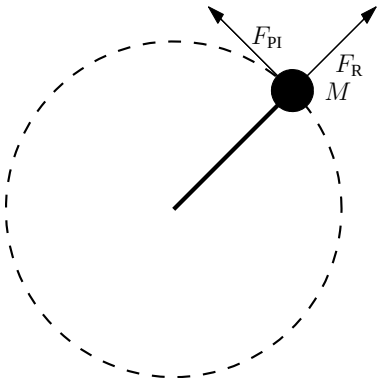

Fig. 2. Circular track.

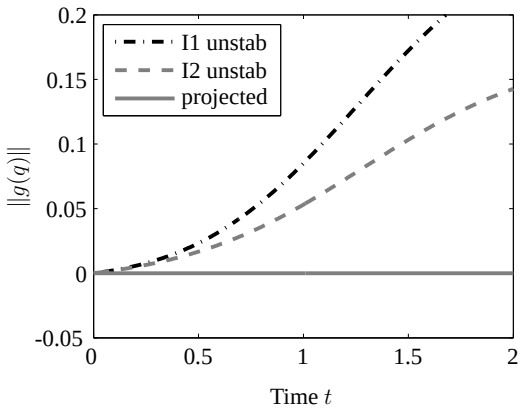

Fig. 3. Drift-off for circular track.

4.2 Car axle from IVP test set

The well known IVP test set at `http://pitagora.dm.uniba.it/~testset/` consists of several benchmark problems to test the accuracy of solvers for initial value problems. We selected the "car axle problem" to study the effects of Jacobian partitioning, see Section 2, and constraint stabilization, see Section 3, for a non-trivial but still rather simple multi-body system model. This benchmark represents a very simple model of a car axle on a bumpy road, see Fig. 4.

As the springs have a Hooke's constant of 10^4 we are outside the stability domain of the explicit Euler method for the stepsize $h = 1.0$ ms. Fig. 7 shows that the explicit

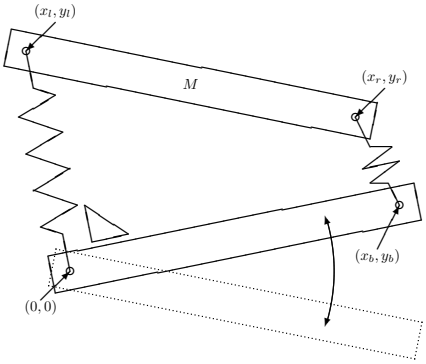

Fig. 4. Car axle from IVP test set.

Euler method is unstable for this problem independent of the method that is used for the stabilization of the constraints. If we use any of the partitioned Jacobians J_1, J_2, J_3 of Section 2 that contains the derivative $\tilde{f}_q$, stable integration of the car axle problem is possible.

The results of Fig. 5 illustrate again the drift-off effect with an unbounded growth of the error in the constraints for the unstabilized methods. For all stabilized methods the errors in the constraints remain bounded. Fig. 6 shows the maximum error $\|g(q)\|$ in the position constraints vs. stepsize h. For both unstabilized methods, the error is of size $\mathcal{O}(h)$ which we expected since the linear-implicit Euler method has order 1. Both Baumgarte methods have an error in the magnitude of $\mathcal{O}(h^2)$. For the projection methods (with only *one* Newton step per time step) we observe an error of size $\mathcal{O}(h^3)$ which coincides perfectly with the theoretical results of Section 3.

Fig. 8 shows the correlation between global error and stabilization method for selected methods. The global error is seen to be sensitive to the error in the constraints. Projection improves the global error by a factor of 10. For longer time intervals this difference grows considerably.

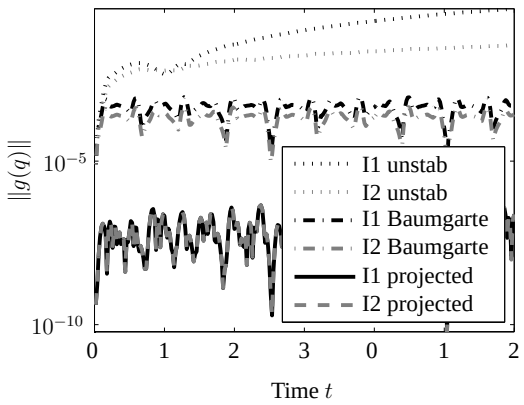

Fig. 5. Benchmark "Car axle": Drift, $h = 10.0\,\text{ms}$.

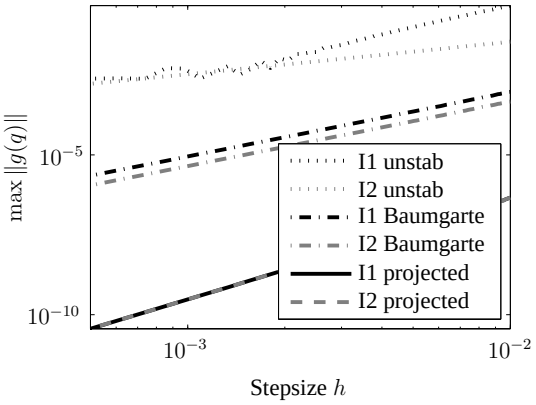

Fig. 6. Benchmark "Car axle": Maximum Drift vs. stepsize h.

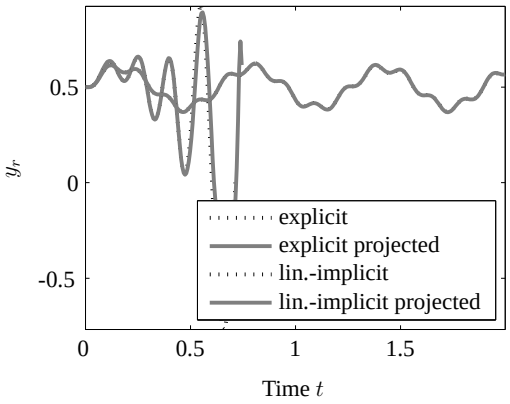

Fig. 7. Benchmark "Car axle": Explicit and (partitioned) linear-implicit Euler method.

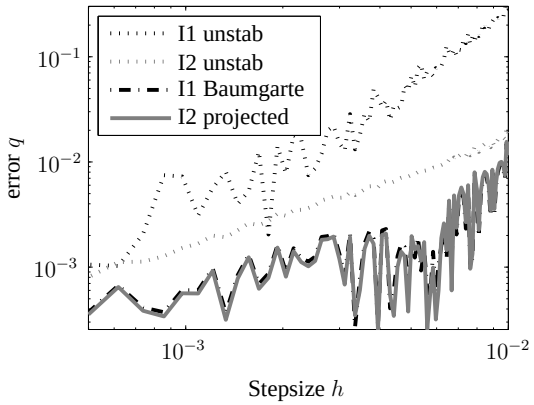

Fig. 8. Benchmark "Car axle": Global error vs. stepsize h.

4.3 Vehicle-trailer coupling in veDYNA

The industrial software package veDYNA [12] implements real-time simulation of full vehicle dynamics. Following the approach of Rill [19], the motion of vehicle and trailer is described by mechanical multi-body systems for the basic chassis extended by sophisticated axle models which can either be depicted by nonlinear elasto-kinematic look-up tables or by detailed multi-body models, including the respective control arms, drag links, subframes and bushings. Intrinsic vehicle dynamics properties are considered by specifi c component models for the drive train, the steering system and the tire dynamics. The choice of suitable generalized coordinates and velocities allows eliminating possible constraints in the respective equations of motion for vehicle and trailer and yields systems of ordinary differential equations formulated according to Jourdain's principle.

In the present section, we consider the simulation of vehicle-trailer coupling in veDYNA [7,8]. In former versions of veDYNA, the coupling between vehicle and trailer was implemented by the use of a spring-damper system to maintain the ODE structure of the equations of motion. This model has a limited accuracy and requires large spring and damper coeffi cients which make the system very stiff. In addition, the model accuracy with regard to the representation of the specifi c joint geometry and the computation of the coupling forces is limited.

To overcome these limitations, a new vehicle-trailer coupling was implemented in veDYNA which explicitly considers algebraic constraints between vehicle and trailer. Due to the modular structure of veDYNA, which is implemented by a couple of Matlab/Simulink-Blocks, the new coupling could be implemented in a separate hook module, see Fig. 9. To maintain the very effi cient integration scheme of veDYNA which needs only one evaluation of the equations of motion of the trailer and of the vehicle at each time-step, only a limited number of the integration methods described in Section 3 is suitable for implementation in veDYNA.

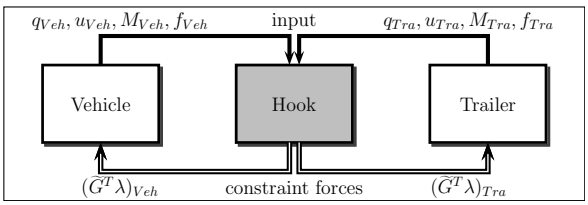

Fig. 9. Model structure for vehicle-trailer coupling in veDYNA.

The integration scheme of veDYNA uses a partitioned linear-implicit Euler method to solve the system of differential equations

$$\begin{aligned} \dot{q} &= Vu, \\ M\dot{u} &= f(q,u) \end{aligned} \tag{14}$$

where q are the position coordinates, u the generalized velocities, and f the vector of generalized forces and torques. At each time step, the new velocities u_{n+1} are given by the solution of the linear system

$$\left(M_n - h\frac{\partial f}{\partial u} - h^2\frac{\partial f}{\partial q}V_n\right)(u_{n+1} - u_n) = hf(q_n, u_n)$$

and afterwards the new positions q_{n+1} are given by

$$q_{n+1} = q_n + hV_{n+1}u_{n+1}.$$

Compared with the index-2 methods of Section 3 which start with the calculation of the new positions q_{n+1}, the calculation steps of the linear-implicit Euler method in veDYNA are in reverse order. Therefore no index-2 method can be used in veDYNA without rewriting of core functions.

Projection methods are incompatible with the modular structure of veDYNA where each block is evaluated only once per time step. There is only one stabilization method of Section 3 that is applicable in veDYNA: Baumgarte stabilization. This method can be implemented by extensions of the right hand sides of (14) without changes in the calling sequences and integration structure of veDYNA.

Several simulations with different vehicle-trailer combinations and different driving maneuvers were performed in veDYNA. Exemplarily, we show results from a heavy truck on a demanding handling course, further results are presented in [7,8]. Fig. 10 depicts the position difference between truck and trailer; defects for both the implementation via spring-damper element and the DAE coupling are shown. Both approaches yield long-term stability. The overall position error for the DAE solver is limited by 10^{-5} m, while an almost constant defect in the range of 10^{-1} m is obtained for the force coupling. The characteristic of the constraint forces (Fig. 11) shows that also the oscillations and the amplitude of the coupling forces are reduced to reasonable values.

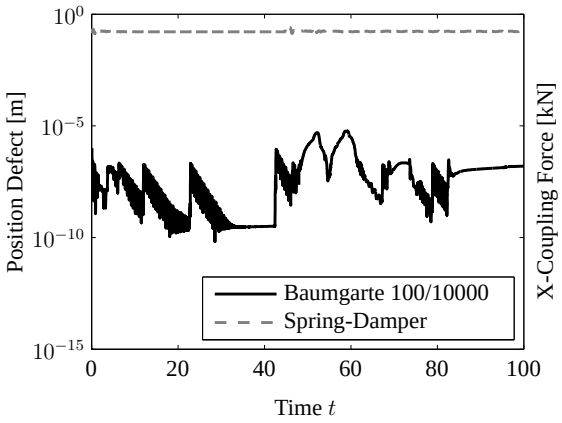

Fig. 10. Position defect between vehicle and trailer.

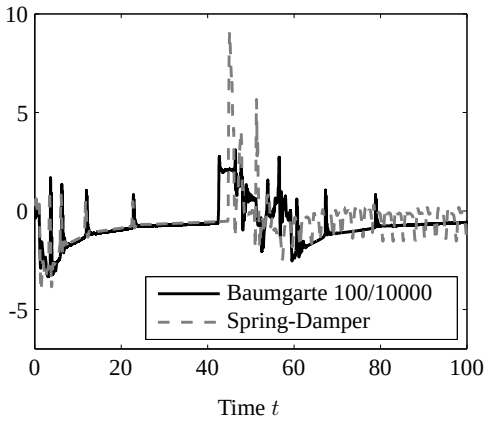

Fig. 11. Coupling forces between vehicle and trailer.

The implementation of the Baumgarte stabilization showed that for veDYNA simulation with an integration step size of $h = 1.0\,\text{ms}$ the theoretically optimal Baumgarte parameters $\alpha = 1/h$ and $\beta = 1/h^2$ are too large. The coeffi cients had to be reduced to $\alpha = 100$ and $\beta = 10^4$ to avoid numerical instabilities due to the increased overall stiffness.

Moreover, also a modifi ed version of a sequential regularization method of order 1 was implemented in veDYNA [7] which yielded similar results. Because of the promising results of the Baumgarte stabilization, future versions of veDYNA will incorporate the new vehicle-trailer coupling by algebraic constraints.

The positive experience with vehicle-trailer coupling shows that a numerically stable time integration of constrained systems in real-time may be achieved by the methods of Section 3. In the present industrial application, the joint modelling by constraints is clearly superior to the penalty techniques that are typical of a classical ODE framework for time integration in real-time.

5 Summary

In real-time applications, the explicit Euler method for time integration may be considered as quasi-standard. It is well known, however, that the explicit Euler method suffers from numerical instability in the stiff case. Furthermore the method is restricted to unconstrained systems.

The stability problem is solved by the linear-implicit Euler method that may be interpreted as implicit Euler method with just one simplifi ed Newton step in the corrector iteration of each time step. The stability analysis for a linear test problem shows that the stability properties of the linear-implicit method depend strongly on the structure of the Jacobian approximation in use.

For constrained systems, explicit or linear-implicit Euler method are applied to an index reduced formulation of the equations of motion. The drift-off effect is avoided on arbitrary long time intervals by just one Newton step of a projection method in each time step. A comparison with the classical Baumgarte stabilization technique shows that the error in the constraints is smaller by a factor of h if the projection method is used.

The results are illustrated by numerical tests for two benchmarks and one large scale applied problem that was solved in a developer version of an industrial multi-body system simulation package.

References

[1] U.M. Ascher, H. Chin, and S. Reich. Stabilization of DAEs and invariant manifolds. *Numer. Math.*, 67:131–149, 1994.

[2] U.M. Ascher and P. Lin. Sequential Regularization Methods for Nonlinear Higher-Index DAEs. *SIAM J. Sci. Comput.*, 18(1):160–181, 1997.

[3] J. Baumgarte. Stabilization of constraints and integrals of motion in dynamical systems. *Computer Methods in Applied Mechanics and Engineering*, 1:1–16, 1972.

[4] K.E. Brenan, S.L. Campbell, and L.R. Petzold. *Numerical solution of initial–value problems in differential–algebraic equations*. SIAM, Philadelphia, 2nd edition, 1996.

[5] B. Burgermeister. Echtzeitfähige Zeitintegration von differentiell-algebraischen Systemen in der Mehrkörperdynamik. Master Thesis, Munich University of Technology, Department of Mathematics, 2003.

[6] A. Eichberger and W. Rulka. Process Save Reduction by Macro Joint Approach: The Key to Real Time and Efficient Vehicle Simulation. *Vehicle System Dynamics*, 41:401–413, 2004.

[7] B. Esterl. Echtzeitfähige Fahrzeug-Anhänger-Kopplung mit Algorithmen für differential-algebraische Systeme. Master Thesis, Munich University of Technology, Department of Mathematics, 2004.

[8] B. Esterl, T. Butz, B. Simeon, and B. Burgermeister. Real-time capable vehicle-trailer coupling by algorithms for differential-algebraic equations. Technical report, TU München, 2005, submitted for publication.

[9] C. Führer. *Differential-algebraische Gleichungssysteme in mechanischen Mehrkörpersystemen. Theorie, numerische Ansätze und Anwendungen*. PhD thesis, TU München, Mathematisches Institut und Institut für Informatik, 1988.

[10] C.W. Gear, G.K. Gupta, and B. Leimkuhler. Automatic integration of Euler–Lagrange equations with constraints. *J. Comp. Appl. Math.*, 12&13:77–90, 1985.

[11] M. Gipser. *Systemdynamik und Simulation*. Teubner–Verlag, Stuttgart Leipzig, 1999.

[12] TESIS DYNAware GmbH. *veDYNA User Manual*. München, 2004.

[13] G.H. Golub and Ch.F. van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore London, 2nd edition, 1989.

[14] H. Hahn and B. Simeon. Separation principle of mechanical system models including stabilized constraint relations. *Archive of Applied Mechanics*, 64:147–153, 1994.

[15] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations. II. Stiff and Differential-Algebraic Problems*. Springer–Verlag, Berlin Heidelberg New York, 2nd edition, 1996.

[16] E.J. Haug. *Computer Aided Kinematics and Dynamics of Mechanical Systems*, volume I. Allyn and Bacon, Boston, MA, 1989.

[17] Ch. Lubich, Ch. Engstler, U. Nowak, and U. Pöhle. Numerical integration of constrained mechanical systems using MEXX. *Mech. Struct. Mach.*, 23:473–495, 1995.

[18] N. Orlandea. *Development and Application of Node–Analogous Sparsity–Oriented Methods for Simulation of Mechanical Dynamic Systems*. PhD thesis, University of Michigan, 1973.

[19] G. Rill. *Simulation von Kraftfahrzeugen*. Fundamentals and Advances in the Engineering Sciences. Vieweg, Braunschweig Wiesbaden, 1994.

[20] R.E. Roberson and R. Schwertassek. *Dynamics of Multibody Systems*. Springer–Verlag, Berlin Heidelberg New York, 1988.

[21] W. Rulka. SIMPACK – A computer program for simulation of large–motion multibody systems. In W.O. Schiehlen, editor, *Multibody Systems Handbook*. Springer–Verlag, Berlin Heidelberg New York, 1990.

[22] W. Rulka and E. Pankiewicz. MBS Approach to Generate Equations of Motions for HiL-Simulations in Vehicle Dynamics. *Multibody System Dynamics*, 14:367–386, 2005.

[23] W.O. Schiehlen, editor. *Multibody Systems Handbook*. Springer–Verlag, Berlin Heidelberg New York, 1990.

[24] A. Schiela and F. Bornemann. Sparsing in Real Time Simulation. *Z. Angew. Math. Mech.*, 83:637–647, 2003.

[25] R. Weiner, M. Arnold, P. Rentrop, and K. Strehmel. Partitioning strategies in Runge–Kutta type methods. *IMA J. Numer. Anal.*, 13:303–319, 1993.